UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

MODELOS DE REDES NEURONALES PROFUNDAS PARA LA PREDICCIÓN DE SECUENCIAS DE ÁREAS DE INTERÉS, DATOS DE MIRADA E INDICADORES MENTALES, REGISTRADOS PARA USUARIOS DE UN SITIO WEB.

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN GESTIÓN DE OPERACIONES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INDUSTRIAL

FRANCISCO JAVIER DÍAZ GUERRA

PROFESOR GUÍA:
ÁNGEL JIMÉNEZ MOLINA

MIEMBROS DE LA COMISIÓN:
RICHARD WEBER HAAS
CHARLES THRAVES CORTES-MONROY

SANTIAGO DE CHILE
2019

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN GESTIÓN DE OPERACIONES
POR: FRANCISCO JAVIER DÍAZ GUERRA
FECHA: 2019
PROF. GUÍA: ÁNGEL JIMÉNEZ MOLINA

MODELOS DE REDES NEURONALES PROFUNDAS PARA LA PREDICCIÓN DE SECUENCIAS DE ÁREAS DE INTERÉS, DATOS DE MIRADA E INDICADORES MENTALES, REGISTRADOS PARA USUARIOS DE UN SITIO WEB.

Este trabajo busca generar tres modelos de predicción: fijaciones en áreas de interés que los usuarios visitan durante su navegación en un sitio web específico, fijaciones realizadas en las diferentes coordenadas del sitio, e indicadores mentales, bajo condiciones ambientales controladas, de iluminación y tarea asignada. Se basa en la experimentación llevada a cabo por Cristian Retamal [88] en el "Estudio del comportamiento de la carga cognitiva de usuarios que navegan en un sitio web", bajo el contexto del proyecto Fondecyt "A Cognitive Resource-Aware Mobile Service Framework to Support Human-Computer-Interactions in Ubiquitous Computing Environments", liderado por el Profesor Ángel Jiménez.

La predicción del primer modelo utiliza señales fisio-psicológicas: Eye tracker, sensor de respuesta electrodermal (GSR), Sensor de Temperatura de la piel (ST) y Fotopletismógrafo (PPG), para estudiar las transiciones entre zonas agrupadas semánticamente, basado en ventanas de tiempo de 5 segundos (parámetro a elección), pronosticando secuencias de áreas de interés, y aproximándose a una predicción en tiempo real.

El modelo se basa en tres etapas, "Predicción de intención de visita de cada área de interés", es decir, que zonas elige visitar cada usuario en cada ventana de tiempo, la "Asignación de estado mental" del usuario en cada ventana de tiempo, a partir de variables fisio-psicológicas y "Algoritmo de generación de caminos visuales", etapa final en la cual se consideran los resultados anteriores para entregar la predicción se secuencias para una ventana de tiempo. Para llevar a cabo estas etapas, se utilizan respectivamente los métodos de "Clasificación multi-etiqueta", "Hard clustering" y "Redes neuronales recurrentes".

El segundo modelo utiliza datos de mirada (fijaciones, sacadas y puntos indefinidos) registrados por el EyeTracker para generar densidades de probabilidad en las zonas de la página que son atendidas por el usuario. Utiliza redes neuronales recurrentes con mezclas de densidades gaussianas para generar secuencias de probabilidades de movimientos oculares registrados en una ventana de tiempo. Este modelo se basa en una arquitectura encoder decoder, donde los datos de mirada son codificados por una red neuronal recurrente bidireccional y decodificados en los parámetros de las distribuciones gaussianas a ser mezcladas.

El tercer modelo utiliza cluster de K-mean para categorizar las ventanas de tiempo con los diferentes índices mentales. Posteriormente se utilizan métodos de clasificación de secuencias para las señales EEG, para la predicción de cada índice mental, utilizando las señales fisiológicas del usuario, usando una arquitectura encoder-decoder.

Todos los modelos se evalúan con validación cruzada de diez conjuntos de entrenamiento-prueba, utilizando diferentes medidas de distancia para cada caso.

Dedicado a Mis padres: Francisco y Patricia *que sin ellos no habría llegado a ser lo que soy*
y a mi hermano Christian *compañero de prácticamente toda la vida.*

Agradecimientos

Quisiera agradecer al profesor Ángel Jiménez por otorgarme la oportunidad de trabajar en este proyecto de Tesis y el recibimiento a poder trabajar en Wesstalb. Al mismo tiempo a los integrantes de dicho laboratorio por la calidez y cordialidad que entregaron en cada momento y la ayuda brindada para el avance de esta tesis.

Agradecer enormemente a mis padres quienes me han entregado todo lo necesario para seguir adelante, a lo largo de toda mi vida y hoy en día poder reflejar todo mi esfuerzo en la culminación de mis estudios universitarios.

Agradecer a mi hermano Christian por estar siempre apoyándome y molestándome lo suficiente como para mantenerme concentrado a pesar de todo y en la búsqueda de nuevos logros.

Finalmente agradecer a mis amigos por el apoyo entregado y por que los buenos momentos junto a ellos siempre fueron una instancia para recobrar energías y seguir trabajando en lo que me gusta. En especial a mis amigos que han estado desde siempre, aquellos que sin nombrarlos saben que les estoy haciendo referencia.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Introducción

En la actualidad, es evidente que el uso de tecnologías de información se ha hecho común en las vidas de muchas personas. En particular, la cantidad de usuarios de internet a nivel mundial corresponde al 55,1 % de la población, siendo Europa y Norte América, las zonas de mayor penetración. Sorprende la tendencia al aumento de usuarios, donde desde el año 2000 a la fecha se presenta un crecimiento de 1066 %.

En Chile, para el año 2018 se registra que el 87,5 % de los hogares posee acceso a internet, con lo cual un 77,5 % de la población (total de 14.108.392 personas) posee acceso a la enorme cantidad de información presente en esta red. De hecho, para el año 2018 se registra un total de 1240 millones de sitios web accesibles en el mundo.

A pesar de la enorme variedad de contenido existente en los diferentes sitios web, muchas veces la navegación por parte de los usuarios implica la búsqueda de información relevante para resolver alguna tarea que lleven a cabo, en un ambiente donde se mezcla con información irrelevante para ellos. Debido a lo anterior, es que se han creado diferentes herramientas acerca del diseño de sitios web que permitan una mejor navegación.

El presente trabajo se motiva en lo expuesto anteriormente para generar modelos matemáticos que permitan entender los comportamientos de los usuarios al enfrentarse a un determinado sitio web. En particular, se busca comprender los movimientos oculares llevados a cabo por una persona al navegar en el sitio, además de diferentes indicadores fisio-psicológicos que permitan conocer sus condiciones en diferentes periodos de tiempo. Esto se relaciona a las áreas de estudio “interacción humano computador” y “computer vision”.

Respecto a esta última área, se ha estudiado la generación de mapas de saliencia en imágenes, es decir, conocer las partes de mayor probabilidad de ser vistas por los usuarios, y extendido esto a escenarios más complejos, cómo lo son los videos y la realidad virtual. Además, en el último tiempo se han generado modelos que permiten comprender los patrones de movimiento oculares denominados “scanpaths”. Sin embargo, estos modelos no han sido aplicados en ambientes de sitios web durante la navegación de los usuarios en tiempo real, donde el desafío principal corresponde a su carácter dinámicos y particulares para cada usuario, dados por su actuar de navegación.

En este trabajo se utilizan herramientas de “machine learning”, para modelar e implementar lo expuesto anteriormente, buscando un modelo de predicción de los lugares dentro de la página web en los que el usuario enfocará su mirada durante su navegación.

1.1. Organización del contenido

El contenido de la presente tesis de desarrolla bajo el siguiente esquema: en el capítulo 2 se presenta la contextualización en cuanto a los conceptos manejados para el tratamiento y uso de datos de mirada, y se define el problema abordado de manera general. Posteriormente el capítulo 3 expone el estado del arte en tres principales áreas, los "modelos de predicción de scanpath", el "uso de señales fisio-psicológicas" y los "modelos de predicción de secuencias en otros contextos".

En el capítulo 4 se entrega un marco teórico en el que se explican diferentes modelos y conceptos usados en este trabajo. En particular, se entrega información de los métodos de clasificación multi-etiqueta, métodos de generación de clusters y redes neuronales, relacionados al área de machine learning, usados como herramientas para la generación de modelos en esta investigación.

Luego de la contextualización, en el capítulo 5 se aborda el problema, explicando los conceptos relacionados al presente problema y ampliando la definición del problema entregada en el capítulo 2 mediante la esquematización de la solución a desarrollar. Posteriormente en el capítulo 6 se muestra la implementación de los modelos planteado en el capitulo anterior, entregando resultados, extensiones y conclusiones acerca de estos.

Finalmente, en el capítulo 7 se entregan las conclusiones finales del trabajo, en donde se explican las ventajas y desventajas de las soluciones implementadas, las principales contribuciones del presente trabajo y los desafíos futuros que se desprenden de la investigación realizada.

Capítulo 2

Contextualización y Definición del problema

El presente capítulo pretende entregar al lector la información necesaria para el correcto entendimiento del trabajo abordado. Para ello, en la sección de contextualización se definen los términos técnicos utilizados comúnmente en la literatura relativa al tema. Posteriormente se explica el problema a trabajar, explicando la fuente de datos, estructura y definición.

2.1. Contextualización

En el campo del seguimiento ocular (en inglés, Eye Tracking), se definen los siguientes términos:

Fijación: Es el evento más reportado en los estudios de seguimiento ocular, relativo al movimiento, y corresponde al estado en el que los ojos se mantienen estáticos durante un periodo de tiempo, por ejemplo cuando la mirada se detiene sobre alguna palabra al leer. Generalmente se considera que la posición en la cual se desarrolla una fijación corresponde al lugar en el cual el usuario mantiene su atención.

En la realidad los ojos no se mantienen estáticos por un periodo de tiempo determinado de la fijación, presentándose tres tipos de micro-movimientos: tremor, micro-sacadas, y drift. Por lo tanto, las fijaciones son detectadas como una agrupación de puntos de la mirada, tanto por cercanía en la posición como cercanía temporal. Típicamente una fijación tiene una duración de 100 - 300 milisegundos. En la imagen 2.1 las fijaciones corresponden a los círculos enumerados de 1-4.

Sacada: En inglés Saccade, corresponde al movimiento realizado entre una fijación y otra. Estos son movimientos balísticos teniendo una duración de entre 30 y 80 milisegundos. En la mayor parte del tiempo en que se desarrolla este movimiento la persona se encuentra sin visión.

La sacada no necesariamente se realiza en linea recta (mínimo camino entre dos fijaciones) sino que puede tomar diferentes formas en su trayectoria. Además, el movimiento no se detiene

directamente en la posición de la fijación final, sino que puede sobrepasarla y oscilar antes de alcanzarlo. En la figura 2.1 se representan con flechas de color rojo.

Smooth Pursuit: Corresponde al movimiento en el cual una persona realiza el seguimiento con la mirada de algún objeto en movimiento. A diferencia de la sacada es un movimiento más lento que requiere de un estímulo para ser llevado a cabo (objeto en movimiento). La sacada en cambio puede realizarse en una superficie sin estímulo presente.

Scanpath: Corresponde a la ruta que siguen los eventos oculomotores (fijaciones y sacadas) en un espacio en un determinado tiempo. Estos caminos requieren de un inicio y un final. El espacio es frecuentemente tratado en dos coordenadas x e y, por lo cual un scanpath de largo L puede representarse como la secuencia de coordenadas $S_{\mathrm{i}} = (x_{\mathrm{i}}, y_{\mathrm{i}}, t_{\mathrm{i}}), \mathrm{i} = 1...L$. Por otro lado, los scanpath tienen diferentes formas de representación, tales como las cadenas de componentes semánticas, vectores euclidianos y mapas de atención. En la figura 2.1 se representa con la secuencia de fijaciones y sacadas (círculos y flechas de color rojo).

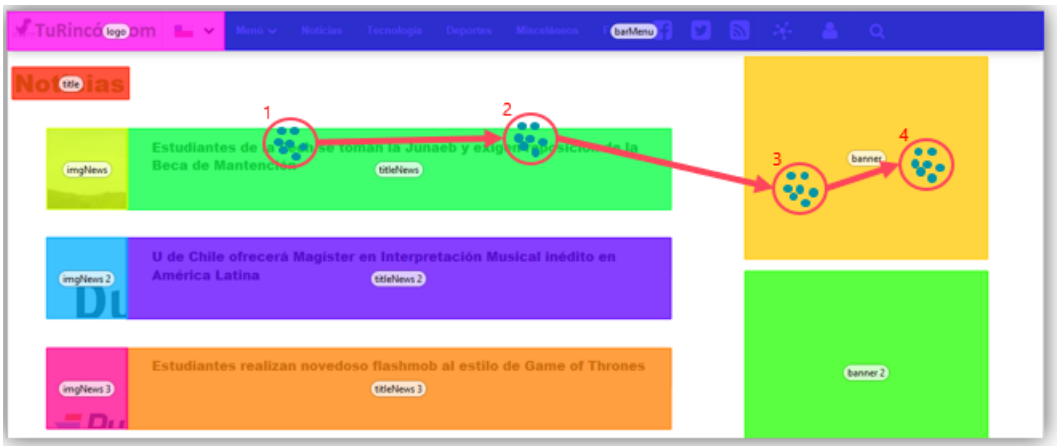


Figura 2.1: Ejemplos eventos oculares.

2.1.1. Áreas de interés

En esta sección se define lo que corresponde a **Áreas de interés (AOI)** o también llamadas **regiones de interés (ROI)** o **Zonas**. En términos simples, corresponde a las regiones dentro del estímulo a las cuales el investigador busca conocer los comportamientos del usuario, por ejemplo, saber la fecuencia de visita en los avisos publicitarios dentro de la página web.

La necesidad de definir las áreas de interés correctamente es que esto permite posteriormente la detección de eventos importantes, como lo son por ejemplo, las transiciones, detenciones dentro del área (en inglés dwell) y los alcances (hit). Al mismo tiempo, la representación del estímulo en estas áreas de interés facilitan el manejo de datos, ya sea con cadenas de caracteres, matrices de transición, gráficos, entre otros.

La definición de áreas de interés se debe realizar de acuerdo a las hipótesis que se desean estudiar, y en el caso de que las áreas de interés sean creadas posterior a la toma de datos, se pueden generar hipótesis adicionales. Además, es necesario considerar una serie de buenas prácticas, tales como considerar el tamaño de las áreas de interés (que sean comparables), el

espacio de separación entre ellas y el posible traslape que puede existir entre dos o más zonas (esto provocaría comportamientos similares en ambas AOI's traslapadas).

Existen variados tipos de área de interés, sin embargo, para este trabajo es necesario conocer las siguientes definiciones:

Whitespace: Corresponde a las zonas del estímulo que no se definen como algún área de interés. No necesariamente corresponden a zonas desocupadas. Por ejemplo, en un experimento acerca de los productos de una góndola, en la cual las pastas son definidas como áreas de interés, algunas zonas como lo son las salsas u otros productos, corresponden whitespaces.

AOI estáticos: AOI en las cuales se especifica la forma y tamaño sin tener variación a través del tiempo.

AOI dinámicos: AOI que presentan cambios en la forma y/o tamaño a través del tiempo. Por ejemplo, las AOI dinámicas se presentan en estímulos con movimiento como lo son los vídeos. Actualmente existen múltiples herramientas para la definición de estos casos.

AOI Semánticas: Se definen áreas de interés del estímulo de manera significativa, pudiendo existir diferencia en forma y/o tamaño respecto a otras áreas de interés. Por ejemplo, definir las diferentes partes en la estructura de una noticia presentada en un periódico (título, copete, cuerpo de la noticia, fotografía y epígrafe).

Gridded AOI: Las AOI en forma de grilla corresponden a una división espacial del estímulo en una grilla. Las AOI pueden presentar problemas en la interpretación debido a que parte de la semántica del estímulo es compartido entre diferentes cuadrantes de la grilla.

A continuación se definen los principales eventos relacionados a las áreas de interés.

AOI Hit: Corresponde al estado en el que los datos crudos o las fijaciones poseen sus coordenadas dentro del AOI definido. Generalmente se considera que un AOI no posee un hit si no se cumple con que la fijación dure un mínimo de tiempo dentro de la zona.

AOI Dwell: Se define como la visita a un AOI desde la entrada a la salida. Al igual que las fijaciones, posee un punto de inicio, final, dispersión, etc; sin embargo, el tiempo que toma en desarrollarse es mayor. Desde la entrada a un área de interés se desarrollan variadas fijaciones en su interior y finalmente una fijación en el exterior.

Transiciones: Corresponde a movimientos realizados entre dos AOI. Es similar a las sacadas, ya que posee amplitud, duración y otras características, sin embargo, son entidades más largas pudiendo estar compuesta de más de una sacada, y al realizarse pueden requerir de fijaciones en áreas que no cubiertas por las AOI definidas. No se considera como transición a la salida y entrada nuevamente a la misma área de interés.

Retorno: También conocido como revisita, corresponde a la transición a un área de interés ya visitada.

2.2. Definición del problema

El problema abordado se basa en la experimentación realizada por Cristian Retamal en su tesis "Estudio del comportamiento de la carga cognitiva de usuarios que navegan en un sitio web" [88], en la cual se pide a un conjunto de participantes navegar libremente por un sitio web. Se poseen datos acerca de la navegación web de los participantes de dicho estudio, los que serán usados en este trabajo.

Se tiene registro de las señales capturadas por el EyeTracker, incluyendo señales del comportamiento ocular de la persona y registro de eventos (clic, despliegue de elementos dinámicos, scroll, etc.), además el video correspondiente a cada experiencia realizada. Al mismo tiempo, se cuenta con registros de electroencefalografía, actividad electro-dérmica, actividad sistema cardiovascular y temperatura corporal de los participantes durante todo el desarrollo del experimento.

En dicho estudio se comprueba la hipótesis de que es posible medir la carga cognitiva para actividades de navegación web frente a un computador mediante señales psico-fisiológicas y que, además, se produce una baja en la carga mental del usuario en momentos de transición entre el análisis de un elemento web y otro. Esto se realiza mediante modelos de clasificación de carga cognitiva basados en etiquetas logradas a partir de los datos de tamaño de dilatación pupilar.

Lo que se espera en este estudio es generar un modelo de predicción de fijaciones realizadas por algún usuario que navega en un sitio web específico, bajo condiciones ambientales controladas, tales como la iluminación y la tarea asignada, a través del tiempo. De igual manera, se pretende categorizar y predecir diferentes índices de estado mental de cada usuario en un determinado intervalo de tiempo.

El modelo busca integrar la interacción del usuario con el sitio web en cada momento y al mismo tiempo la medición de sus señales psico-fisiológicas nombradas anteriormente. Además, se espera que la predicción considere horizontes temporales indefinidos, que dependan de la cantidad de tiempo que el usuario permanezca en interacción con la página web que puede variar entre diferentes usuarios.

Dentro de las motivaciones para realizar este trabajo se encuentran los posibles usos que se le puede dar a este tipo de modelos. El pronosticar las áreas de interés que un usuario escogerá durante su navegación en un sitio web permite entregar información de manera menos invasiva, optimizar el diseño o los contenidos del sitio, entregar información dinámicamente, o mejorar la transmisión de datos en zonas que serán objeto de atención de los usuarios, entre otros posibles usos.

Otra motivación corresponde a que el resolver este problema entregará nuevos enfoques para la predicción de áreas de interés en ambientes dinámicos, y al mismo tiempo permitirá la integración de los estados mentales de cada usuario en la predicción, lo cual no se ha dado a conocer dentro de la literatura hasta el momento es que es llevado a cabo este trabajo.

En un principio, para el caso de la predicción de fijaciones, se buscaba un modelo de probabilidades de transición desde un punto a otro dentro de la página web. Sin embargo,

debido a la gran posibilidad de escenarios existentes, y la limitación en datos de entrenamiento que se poseen, se establecen dos escenarios a ser estudiados: La "predicción a nivel semántico de AOI's" y la "predicción píxel a píxel".

A continuación se formalizan los problemas de predicción de fijaciones en ambos escenarios y el modelo de predicción de índices de estados mentales.

2.2.1. Modelo de predicción de fijaciones

Primer escenario: predicción a nivel semántico de AOI's

Primero se estudian las transiciones entre zonas de la página web (la cual es única durante toda la experimentación realizada en el estudio original), es decir, aglomeraciones de puntos de manera semántica, áreas de interés que serán detalladas más adelante. En este primer escenario se busca la predicción de **secuencias relativas a dwell time**.

Dado los siguientes conjuntos, el problema se define formalmente:

$u \in U$: conjunto de usuarios.

$c \in C$: conjunto de componentes de la página web (Áreas de interés).

Sea n_c^u el número de visitas que un usuario u realiza en una componente c durante una navegación en el sitio web. Se entenderá como visita cada vez que se registre un Dwell en la componente definida, dada la naturaleza semántica del sitio, permitiendo, por ejemplo, que una visita a una componente que posea texto esté compuestas por múltiples fijaciones (inherentes al proceso de lectura).

Cada visita a una componente c quedará definida por un tiempo de entrada Te y un tiempo de salida Ts. Así el conjunto $L_c^u = \{(Te_j, Ts_j)\}_{j=1}^{n_c^u}$ corresponde a todas las visitas realizadas por el usuario u a la componente c.

Durante la navegación completa realizada por el usuario u se tendrá el conjunto de secuencias $Sec_u = \Phi(\{L_c^u\}_{c=1}^{|C|})$, donde la función Φ entrega la secuencia ordenada de las áreas de interés visitadas por el usuario. Por ejemplo, si la pagina web consta de las áreas de interés A,B,..,Z, una posible secuencia corresponde a que el usuario realice el patrón ABCDAC, donde las letras se encuentran ordenadas temporalmente.

Sea V_{sec}^u el conjunto de variables relacionadas a la navegación realizada por el usuario u y VF_{sec}^u el conjunto de variables de las características psico-fisiológicas que presenta. Se agrega el subíndice *sec* representando un conjunto de variables usado para el caso de secuencias temporales, diferenciándolas de las variables del segundo escenario. Estas variables serán definidas formalmente más adelante [1].

Entonces, se busca una función Γ_{sec}, del conjunto de funciones de clasificadores, que minimice el error entre las predicciones realizadas en la secuencia de AOI visitadas por los usuarios y las secuencias llevadas a cabo en la realidad. Sea m alguna medida de distancia

[1] ver sección 6.3.4

entre secuencias (como por ejemplo la distancia de Levenshtein [2]), las que serán discutidas más adelante, entonces se busca minimizar:

$$\text{mín} \sum_{u=1}^{|U|} m(\Gamma_{sec}(V^u_{sec}, VF^u_{sec}), Sec_u)$$

Segundo escenario: predicción píxel a píxel

En el segundo escenario, se pretende generar una predicción de manera más detallada, considerando la predicción de **scanpaths**. Se puede observar, al comparar los resultados con el primer escenario, si el hecho de entregar información de manera más detallada permite mejores resultados, lo que permitiría entender de mejor manera las variables involucradas en la elección de movimientos oculares realizados por los usuarios.

Su definición es similar al caso anterior, pero en vez de utilizar el conjunto de componentes C, se establecen conjuntos de puntos p de coordenadas en la página web de tamaño (x_{max}, y_{max}) $P = \{\ p = (x, y)\ | x \in x_{max}, y \in y_{max}\}$.

El conjunto L^u_p corresponde a los puntos visitados por el usuario u y se definen de manera similar al escenario anterior.

El conjunto de Scanpaths para un usuario u, $Scan_u$, se define como la función ordenada de puntos p visitados, $Scan_u = \Phi(\{L^u_p\}^{|P|}_{p=1})$ y se busca minimizar:

$$\text{mín} \sum_{u=1}^{|U|} m(\Gamma_{scan}(V^u_{scan}, VF^u_{scan}), Scan_u)$$

2.2.2. Predicción de indicadores mentales

Dado un usuario u, su indicador de algún estado mental dentro del intervalo de tiempo (t_o, t_f) estará dado por una función que depende de su señal de encefalograma registrada en dicha ventana de tiempo. De esta manera, cada ventana de tiempo v será categorizado por una clase IM_v del indicador mental a ser considerado (los cuales serán detallados en la sección 6.5).

La función Γ_{IM} corresponde a un clasificador que entrega los resultados de una estimación del indicador mental real IM_v, dado por los datos registrados antes de la ventana de tiempo correspondiente VF^u_{-v}, generando para cada ventana de tiempo la estimación $\hat{IM}_v$. Así, se busca minimizar la diferencia entre las clases predichas para cada ventana de tiempo y los indicadores mentales reales, donde V_u corresponde al conjunto de ventanas de tiempo definidas para un usuario u y m corresponde a alguna medida de diferenciación, por ejemplo, la métrica de accuracy:

$$\text{mín} \sum_{u=1}^{|U|} \sum_{v=1}^{|V_u|} m(\Gamma_{IM}(VF^u_{-v}), IM^u_v)$$

[2] ver sección 6.3.6

Capítulo 3

Estado del Arte

En la actualidad existen diferentes enfoques para la resolución del problema de predicción de scanpaths, en variados ambientes. Cada uno presenta fortalezas que lo hacen robusto antes situaciones que cumplan las características de aplicación. Sin embargo, estos modelos poseen dificultades para su directa aplicación en sitios web.

Los modelos de predicción de scanpath encontrados en la literatura no incorporan variables fisio-psicológicas, por lo tanto, en el presente capítulo, además de explicar brevemente los estudios relacionados con la predicción de scanpaths, se expondrá acerca de estudios relacionados a las señales fisio-psicológicas a considerar en este trabajo. Al mismo tiempo, se entregará información acerca de estudios desarrollados sobre la predicción de secuencias en otros campos, no necesariamente relacionados a movimiento ocular o atención visual.

3.1. Modelos de predicción de Scanpaths

La predicción de scanpath se basa en la predicción de fijaciones secuenciales, las cuales generan un camino seguido por la mirada del usuario. Existen diferentes métodos para lograr esta compleja tarea.

En primer lugar se consideran los denominados mapas de saliencia, en inglés, **saliency maps**, una parte importante en el área de computer vision durante los últimos 30 años. Estos son mapas ordenados topográficamente que representa la prominencia visual de una escena visual correspondiente.

En la realidad, los estímulos visuales presentan gran cantidad de información. Lo que estos mapas entregan es un tratamiento de los estímulos visuales que permiten entender de manera más sencilla las partes de mayor o menor importancia en la selección de puntos de atención. En lo que se diferencian los métodos de saliency maps es en la información utilizada para su generación y los algoritmos implementados.

El estudio realizado en [63] se basa en este enfoque, proponiendo una red neuronal convolucional profunda, capaz de predecir la fijaciones y segmentar objetos en diferentes imágenes. En [55] también desarrollan un enfoque de transformación desde características presentes en

diferentes imágenes naturales, a mapas de saliencia, buscando diferentes métodos de predicción de fijaciones utilizando una menor cantidad de características presentes en las imágenes.

En [97] proponen una forma efectiva y simple de modelar la atención visual. En este caso, las imágenes son convolucionadas con una plantilla y post-procesadas para entregar el resultado. Además de lo anterior, existen registros de implementaciones con redes neuronales recurrentes u otros métodos, que sin embargo, no se dará mayor detalle.

A modo de ejemplo acerca de los resultados arrojados por estos métodos, se presenta la imagen 3.1 (véase [97]). A la izquierda se ve un imagen real y a la derecha la implementación de diferentes métodos para la generación de sus mapas correspondientes.

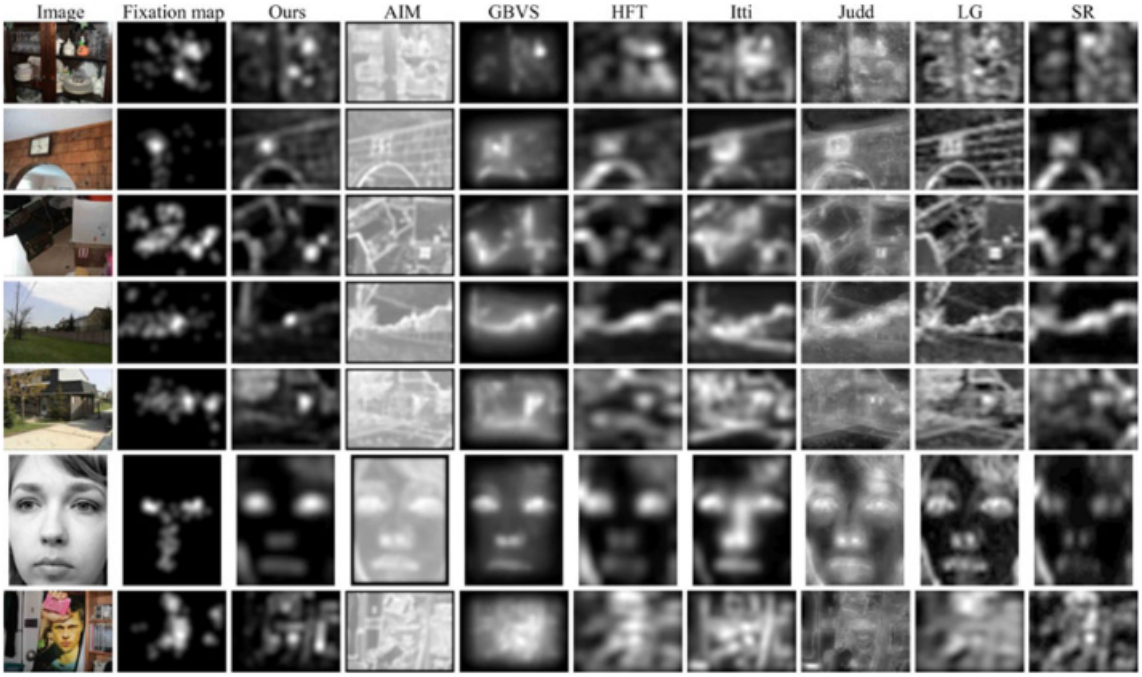


Figura 3.1: Visualización modelo saliency map.

Los resultados entregados por estos mapas pueden ser usados para la generación de scanpaths a partir de selección secuencial de las zonas de mayor prominencia. En general se considera que los mapas generados corresponden a una distribución de probabilidad acerca de las zonas de mayor atención de los usuarios, y a partir de esto, se aplican algoritmos greedy, sampleos de naive u otros métodos dinámicos.

Por otro lado, es posible encontrar en la literatura otros métodos de generación de scanpaths. En [68] se estima una secuencia de posiciones de la mirada de un usuario en una imagen. El modelo se basa en tres componentes: características de bajo nivel (probabilidades de transición entre áreas de la imagen según diferencia de las características), posición espacial y contenido semántico. Para ello emplean modelo de **Markov oculto (HMM)** y distribución de probabilidad Levy Flight.

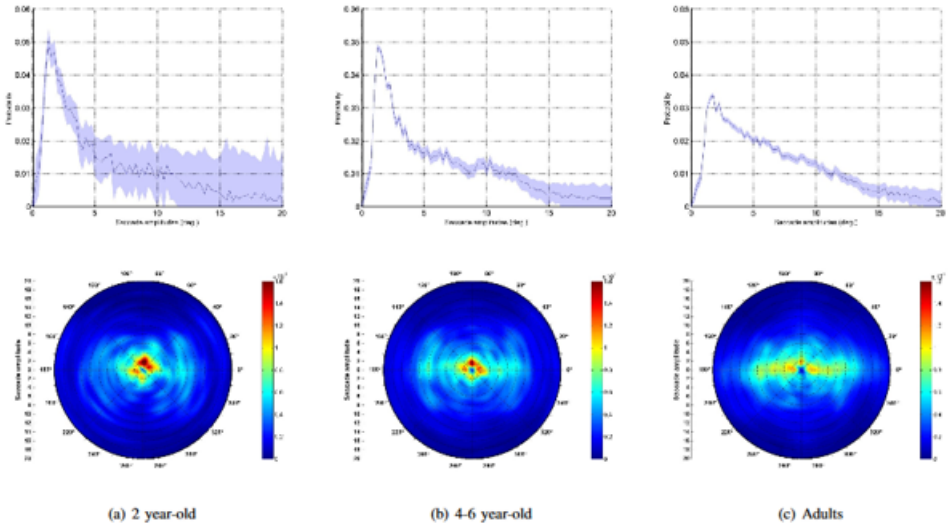
En [52] utilizan registros de datos de visualización de imágenes y un proceso de **decisión markoviano basado en least-squares policy iteration (LSPI)** para aprender acerca

de las políticas de exploración. Utilizando diferentes parámetros en cada transición, es posible entender como varía la importancia de cada característica a lo largo de la exploración realizada por el usuario. De esta forma, se obtienen predicciones de scanpath con un número determinado de pasos a partir del entrenamiento y evaluación a partir de los datos disponibles. Los scanpath predichos son comparados con los scanpath reales.

A pesar que los dos últimos modelos mencionados permiten la predicción de scanpaths en imágenes, estos enfoques no entregan resultados en ambientes dinámicos como la navegación web, y al mismo tiempo, los resultados no contemplan características propias de cada usuario en tiempo real.

Otro grupo de modelos utilizados para la predicción de scanpath son los denominados **modelos de sacada**. En [103], se realizan predicciones de scanpaths en imágenes naturales, integrando tres factores: respuesta sensorial, memoria visual, resolución periférica. Para ello calculan tres filtros multi-banda, relacionado a los tres factores, los cuales son combinados en un único filtro para crear los mapas de respuesta. Calculando la información perceptual residual, la cual varía de manera dinámica según los movimientos hechos por los usuarios, se seleccionan las siguientes fijaciones.

En [73] se realiza otro modelo de sacada en el cual se consideran tres rangos etarios de los usuarios (véase figura 3.2). Este modelo considera la interacción entre la forma en que los usuarios observan la información visual y el estado mental de cada uno. Esta caracterización permite considerar características tales como las capacidades de la persona para realizar ciertos movimientos, sus intereses o estados cognitivos, permitiendo predecir, además de los puntos en los que un usuario prestará atención, el cómo realizarán los movimientos en la escena.



(a) 2 year-old (b) 4-6 year-old (c) Adults

Figura 3.2: Distribuciones en las sacadas realizadas por los diferentes rangos etarios de los usuarios.

En el estudio, se hace alusión a que los niños más pequeños tienen patrones de movimiento de menor rango que los adultos debidos a las diferencias en las capacidades físicas entre usuarios. Al mismo tiempo, considerar la distribuciones de probabilidad en las sacadas de los

usuarios entrega información acerca de cómo las personas interactúan con diferentes estímulos visuales, lo que será comentado más adelante. Estos modelos son considerados a lo largo del trabajo, principalmente por el enfoque de diferenciación en las distribuciones de sacadas a través de los usuarios.

Un estudio reciente [10] lleva a cabo predicciones de mapas de saliencia en videos, cuadro a cuadro. Para ello utilizan métodos de redes recurrentes con densidad de mezcla. Este estudio servirá como base para la exploración de resultados del modelo desarrollado en el segundo escenario, descrito más adelante. Sin embargo, como el estímulo corresponde a una secuencia de imágenes sin interacción del usuario, no tiene sentido realizar una comparación de los resultados obtenidos en dicho estudio con los resultados de la presente tesis.

Como se verá más adelante, el modelo desarrollado para el segundo escenario, a diferencia del estudio antes mencionado, integra características del usuario, en un ambiente dinámico con interacción, a diferencia de los videos, los cuales no se ven afectados por el comportamiento de los usuarios.

Finalmente, en la literatura se pueden encontrar estudios que usan páginas web como estímulos visuales. En [83] se estudia la influencia de las características de la página web en los puntos en los que los usuarios dirigen su atención. Se establece un scanpath ideal a ser seguido por los usuarios y experimentalmente se observan los datos reales. De esta manera se genera un modelo que entrega información acerca de cómo los elementos presentes en la página web afectan los puntos de atención de los usuarios, permitiendo modificar dicha página para lograr que los usuarios sigan el scanpath ideal. Sin embargo, en este estudio no se generan modelos de predicción de scanpaths.

3.2. Estudios de señales fisio-psicológicas

En el desarrollo de este trabajo se pretende utilizar señales fisio-psicológicas para la predicción de movimientos oculares. Es por esto que, a continuación, se definen brevemente las señales a considerar y se exponen los principales estudios acerca de cada una de ellas en relación a conceptos considerados.

Electrocardiografía (ECG)

El corazón presenta actividad eléctrica que produce corrientes que se irradian a través del tejido circundante a la piel. El electrocardiógrafo conecta tres o más electrodos a la piel, detectando dichas corrientes eléctricas. En la figura 3.3 se muestra un ejemplo de la configuración de un electrocardiógrafo (estándar Lead II para la posición de electrodos) y la señal obtenida.

Figura 3.3: Configuración de electrodos ECG Lead II y ejemplo de forma de señal obtenida con esta configuración.

Las corrientes se transforman en formas de onda que representan el ciclo de despolarización y repolarización del corazón. La despolarización del miocardio ocurre cuando una onda de estimulación pasa a través del corazón y estimula al músculo del corazón a contraerse. La repolarización es el retorno al estado de reposo y da como resultado la relajación.

Un ECG muestra la secuencia precisa de eventos eléctricos que ocurren en las células cardíacas a lo largo del ciclo cardíaco. Este proceso corresponde a los eventos que ocurren en el corazón desde un latido hasta el siguiente y está compuesto por dos etapas principales: diástole, durante la cual el corazón se llena de sangre y sístole, cuando el corazón bombea toda esa sangre al cuerpo.

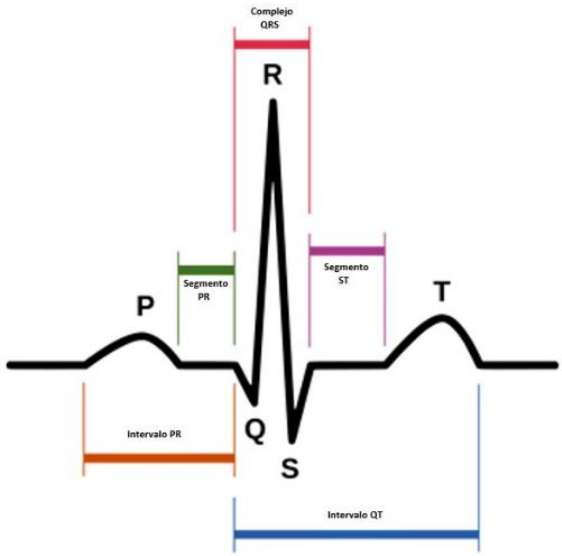


Figura 3.4: Ciclo cardiaco ideal.

En la figura 3.4 se muestra un ciclo cardiaco ideal, compuesto por [93]:

Intervalo PR: Mide el tiempo en que un impulso eléctrico viaja desde el miocardio

auricular adyacente al nódulo SA al miocardio ventricular adyacente a las fibras de la red de Purkinje. Normalmente dura de 0.10 a 0.21 segundos. Varía con la frecuencia cardíaca, siendo más corto cuando predomina el componente simpático. Tiende a aumentar con la edad, presentándose duraciones en la infancia de 0.10 a 0.12 segundos; adolescencia, 0.12 a 0.16 segundos; edad adulta, 0.14 a 0.21 segundos.

Segmento PR: El segmento PR es el segmento plano, generalmente isoeléctrico, entre el final de la onda P y el inicio del complejo QRS.

Intervalo QT: mide la duración de la activación eléctrica y la recuperación del miocardio ventricular. Varía inversamente con la frecuencia cardíaca. Para garantizar la recuperación completa de un ciclo cardíaco antes de que comience el próximo ciclo, la duración de la recuperación disminuye a medida que aumenta la velocidad de activación. La normalidad del intervalo QT solo puede ser determinada corrigiendo la frecuencia cardíaca.

Segmento ST: representa el período durante el cual el miocardio ventricular procede a través de las dos fases preliminares de la repolarización. Estos son las fases consideradas como repolarización temprana. Su longitud está influenciada por factores que alteran la duración de la activación ventricular.

Complejo QRS: la duración del complejo QRS se denomina intervalo QRS, y normalmente oscila entre 0.07 y 0.11 segundos. Tiende a ser un poco más larga en los hombres que en las mujeres. Se mide el intervalo QRS desde el comienzo de la primera onda Q o R que aparece hasta el final de la última onda R, S que aparece.

En un ECG normal se producen ondas R de amplitud 2mV aproximadamente, el cual debe ser amplificado por 2500. De la señal se pueden sacar varias medidas:

Periodo cardíaco: tiempo en milisegundos entre dos ciclos cardiacos adyacentes. En general se mide el tiempo entre dos ondas R consecutivas, dado que es la de mayor amplitud. Esta medida es reciproca al ritmo cardíaco (HR), que es la cantidad de latidos por minuto.

Variación del ritmo cardíaco (HRV): oscilación en el intervalo temporal entre dos beats consecutivos del corazón. Se refiere tanto a la variación del ritmo cardíaco instantáneo como de los intervalos RR. Se mide de distintas maneras, las que incluyen análisis en el dominio del tiempo y de la frecuencia.

Arritmia del seno respiratorio (RSA): el ritmo cardíaco aumenta durante la inspiración y decrece durante la exhalación, por lo que esta medida se utiliza como un índice del control automático del sistema nervioso en el corazón.

Intervalo entre beats (IBI).

Se ha demostrado que para tareas que involucran la manipulación de información, existe un aumento en el ritmo cardíaco [23].

Respecto a medidas directamente relacionadas al ECG, en [44] realizan un estudio centrado en la percepción visual y las tareas enfocadas en la velocidad cognitiva que aprovechan las capacidades comunes en las aplicaciones de computación ubicua.

En dicho estudio, de los datos recolectados en los diferentes sensores psico-fisiológicos, se muestra que medidas relacionadas al ECG presentan mayor precisión entre los niveles de carga cognitiva, entregando precisión superior al 80 % al combinar con otros sensores. Además, al proporcionar un método en tiempo real, es posible generalizar los resultados para evaluar la carga cognitiva en tareas que se encuentran comúnmente en sistemas de computación ubicua y situaciones de atención dividida.

Electroencefalografía (EEG)

Existen medidas relacionadas al cerebro, esto se refiere a registrar actividad eléctrica presente dentro del cerebro o muy cercano a la superficie del cuero cabelludo [37]. De esta manera, utilizando electrodos en el cuero cabelludo, el dispositivo llamado electroencefalógrafo mide la tensión resultante de los cambios de flujo de corriente iónica dentro de las neuronas del cerebro producidos por la actividad sináptica cerebral.

Dependiendo del modelo de electroencefalógrafo utilizado, se dispone de diferentes distribuciones de electrodos. Durante este trabajo se utiliza el sistema denominado 10-20 (ver figura 3.5). Los nombres de los electrodos se componen de una letra, que indica la región del cerebro (F = lóbulo frontal, T = lóbulo temporal, C = centro, O lóbulo occipital) y un número, que indica la posición (par = ubicación derecha de la cabeza, impar = ubicación izquierda de la cabeza).

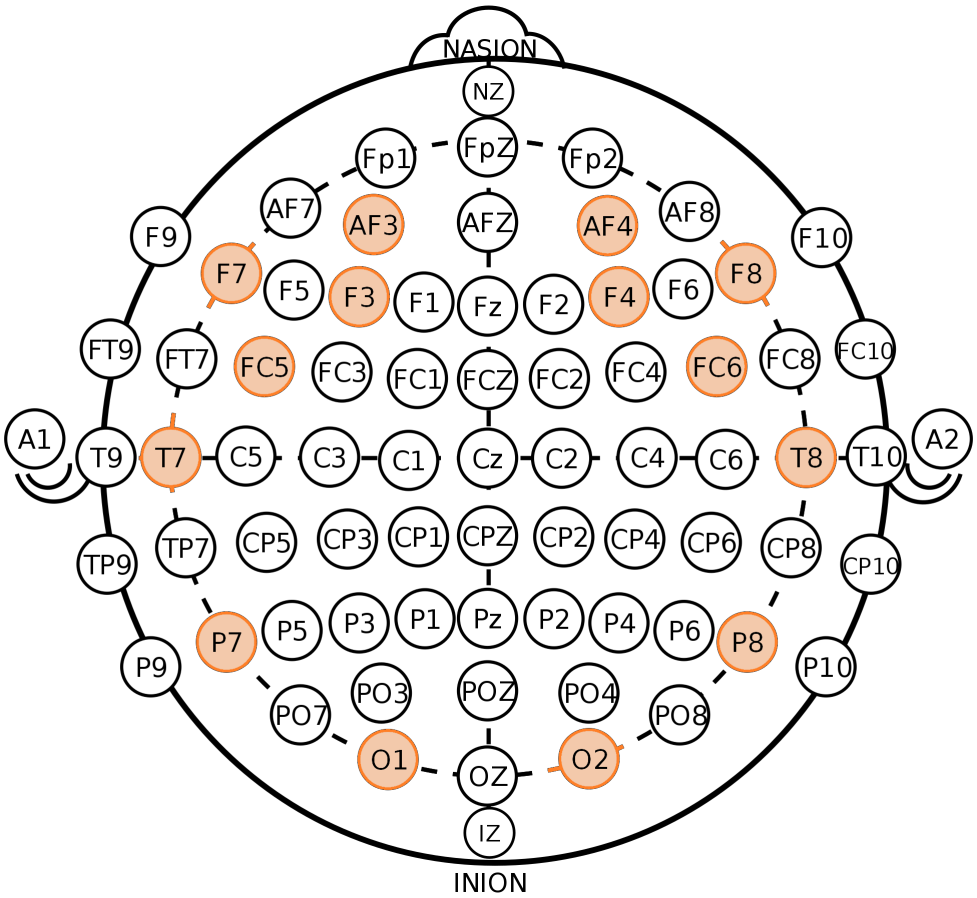


Figura 3.5: Sistema 10-20 de electrodos. Se destacan los electrodos registrados en este trabajo.

Según lo expuesto en [48] el cerebro genera actividad continuamente, incluso en ausencia de algún estímulo, por ejemplo, al dormir. Para aprovechar la actividad cerebral que impulsa nuestro comportamiento, pensamientos, motivaciones y emociones, se utilizan los análisis de frecuencias, recomendados en experimentos de tiempo de prueba limitados y bajo un análisis que no se trate del momento preciso la actividad relacionada con el estímulo, sino más bien del estado general del usuario. Estos análisis son útiles en estudios de estados cognitivo-afectivos, es decir, al medir EEG mientras los usuarios atienden el contenido de los medios, navegan por sitios web o interfaces de software, por ejemplo.

La señal cruda obtenida del EEG es una mezcla de varias frecuencias, que se consideradas un reflejo de cierto estado cognitivo, afectivo o atencional. Así, se definen 5 bandas de frecuencia, descritas brevemente a continuación:

Banda Delta (1-4 Hz): presente sólo en estados de sueño profundo no-REM. Asociada con la profundidad del sueño (a mayor ritmo delta, mayor es la profundidad del sueño). En general esta banda de frecuencias se encuentra más fuerte en el hemisferio derecho del cerebro, y sus fuentes más típicas están localizadas en el tálamo.

Banda Theta (4-8 Hz): se correlaciona a la dificultad de operaciones mentales, por ejemplo, durante periodos de atención focalizada o de captación de información. Puede ser obtenida desde toda la corteza, generada mediante una amplia red que involucra las zonas prefrontal, central, parietal y temporal cortices, y se vuelve más prominente al aumentar la dificultad de la tarea. Generalmente se asocia con procesos cerebrales subyacentes a la carga de trabajo mental o la memoria de trabajo.

Banda Alfa (8-12 Hz): tiene varias correlaciones con funciones sensoriales, motoras y de memoria. Esta banda aumenta los niveles durante relajación mental y física con los ojos cerrados y se reduce o suprime durante actividades mentales o corporales con los ojos abiertos. La supresión de la banda alfa constituye una señal de estados de actividad mental y compromiso con la tarea, interpretándose como que el cerebro está recolectando información desde varios sentidos, coordinando recursos de atención y enfocándose en lo que interesa en el momento. La banda alfa es generada en sitio posteriores corticales, incluyendo la región occipital, parietal y temporal posterior del cerebro.

Banda Beta (12-25 Hz): aumenta sus niveles con pensamientos ansiosos o activos y con estados de concentración activa. También, se observa un aumento en esta banda cuando se planea la ejecución de movimientos, cuando se requiere alguna acción que involucre movimientos finos de dedos y atención focalizada. Además, se establece que existe un sistema de neuronas espejo coordinado por estas frecuencias. Esta banda se genera en las regiones posterior y frontal del cerebro.

Banda Gamma (>25 Hz): no se conoce mucho ni sobre dónde se genera ni sobre su función en el cerebro. Se discute si su propósito es acerca del proceso cognitivo o un subproducto de otros procesos neuronales como movimientos oculares y micro sacadas.

El análisis de bandas de frecuencias ha sido utilizado para comprender cuales son las componentes que influyen tanto en la carga cognitiva, como en el movimiento ocular de los usuarios.

Existe evidencia de que al aumentar la carga cognitiva, se suprime la banda alfa en el lóbulo parietal y que aumenta la banda theta en el lóbulo frontal [37, 6, 90, 59]. Sin embargo, en [58] el autor indica que, a pesar de que las oscilaciones de las bandas alfa y theta reflejan cambios en la carga cognitiva y desempeño de la memoria, es importante definir la banda alfa y theta para cada sujeto a partir del peak de la frecuencia de su banda alfa (IAF por sus siglas en inglés: Individual Alpha Frequency). Es decir, las frecuencias de corte no son las mismas para cada persona. A pesar de esto, existe literatura que utiliza las bandas "estándar" del EEG para clasificar carga cognitiva con buenos resultados [107, 5, 109, 3].

Respuesta electro-dérmica (GSR)

Según [26] Galvanic Skin Response (GSR), también conocida como Electrodermal activity (EDA), es una medida en la conductividad de la piel humana. La conductancia eléctrica de la piel se mide a través de dos electrodos unidos a la mano. Estas medidas proveen indicadores en los cambios del sistema nervioso simpático y últimamente han sido utilizadas para la medición fisio-psicológica de las personas. La conductividad varia según los cambios de humedad de la piel, relacionado con los cambios en la sudoración del usuario.

La señal GSR consiste principalmente de dos componentes [49]:

Componente tónica o Skin conductance level (SCL): presenta variaciones lentas y cambia constantemente dentro del individuo, dependiendo de su hidratación, sequedad de la piel o regulación autonómica. Además, puede diferir entre los individuos.

Componente fásica o Skin Conductance Response (SCR): acompaña a los cambios tónicos alterándose más rápido. Es sensible a eventos específicos de estímulo que aparecen emocionalmente. Estas explosiones ocurren entre 1 y 5 segundos después de la aparición de los estímulos emocionales. Esta componente es más aceptada a la hora de explicar los cambios psico-fisiológicos del individuo.

Para la detección de estos dos componentes existen variados métodos. En [12, 13] se presentan dos métodos de descomposición de la señal cruda GSR usada en el toolbox "Ledalab". Ambas consideran que la señal presenta componentes tónica y fásica, y se diferencian en la manera de calcular cada una de ellas.

En [42] se presenta un tercer método basado en la optimización convexa. El modelo describe la conductancia de la piel como la suma de tres términos: la componente fásica, la tónica y un error de predicción del modelo Gaussiano (ruido blanco).

Existen estudios en variados escenarios en los cuales se utiliza la señal GSR como indicador de los cambios emocionales o cognitivos de los participantes:

En [76] se detectan emociones usando la conductancia de la piel y la actividad muscular.En [92], se mide la conductancia de la piel detectando condiciones de estrés y diferenciándola de condiciones de carga cognitiva. Además, en [94] también se encuentran correlaciones entre esta señal y la carga cognitiva.

En [36] se encuentra un efecto débil entre GSR y la carga cognitiva. En otro estudio [47], observan que la conductancia de la piel aumenta con la dificultad de la tarea desarrollada

por el usuario. En el análisis de la señal GSR durante las etapas de vuelo en un simulador [104] se encuentra un aumento en la respuesta durante el despegue y el aterrizaje, siendo las etapas de demandas cognitivas más altas en los pilotos.

En [65] investigan las técnicas de clasificación (K-means, GMM, SVM y árbol de decisiones) que pueden usarse para determinar automáticamente períodos de estrés agudo dependiendo de la señal GSR y/o el habla de una persona. La señal GSR varía de persona a persona y de un día para otro para la misma persona, por lo que se incluyen otras medidas para obtener un rendimiento más confiable.

En el estudio mostrado en [75] predicen el desempeño de los trabajadores en una tarea bajo condiciones estresantes y potencialmente de alto riesgo. Se muestra que la señal GSR permite predecir qué sujetos se desempeñarán bien bajo condiciones de alta tensión.

En el estudio [80] examinan las características temporales y espectrales del GSR frente a diferentes niveles de dificultad de múltiples tareas. Muestran la importancia de las características exploradas, especialmente las espectrales, en la medición de carga de trabajo cognitiva. Este estudio será de utilidad a la hora de generar features con la señal GSR.

En el estudio [79] se evalúan características de las señales GSR y datos Pupilográfios en la clasificación del nivel de carga cognitiva, experimentando en tareas con cuatro niveles de dificultad. Se muestra que las características estudiadas discriminan razonablemente los niveles de carga y los resultados mejoran al combinar ambas señales.

Temperatura de la piel (ST)

Según [61] la temperatura de la piel (en inglés ST) también se relaciona ante cambios en el estado mental de las personas, disminuyendo ante aumentos en la carga cognitiva y aumentando en respuesta a emociones positivas. En dicho estudio, utilizan medidas de la temperatura corporal junto a otras señales fisio-psicológicas para cuantificar la carga cognitiva de pacientes con accidente cerebro vascular.

En [102] se propone usar temperatura de la piel en zona facial mas otras medidas fisiopsicológicas para evaluar la sensibilidad a los cambios en esfuerzos mentales. Se demuestra que al combinar con estas medidas la sensibilidad de los esfuerzos mentales mejora.

La experimentación realizada en [82] desarrolla una medida fisiológica para la carga de trabajo mental usando el cambio de la temperatura de la piel del rostro humano. Se observa una correlación significativa entre el cambio de la temperatura de la piel de la nariz y la carga de trabajo en ambos experimentos realizados.

Fotopletismograma (PPG)

El Fotopletismograma (Photoplethysmography, PPG) también se conoce en la literatura como Blood Volume Pulse (BVP). Esta señal ha sido usada en otros estudios para la evaluación de la carga cognitiva de los usuarios. De esta señal, al igual que para el ECG, es posible obtener el ritmo cardíaco (HR) del usuario.

En [86] desarrollan un clasificador del nivel de carga cognitiva del usuario. Para ello utilizan

diferentes señales, entre ellas PPG. Se concluye que el sistema es capás de detectar estados de carga cognitiva en ventanas de datos de un minuto.

En [39] se capturan señales psico-fisiológicas de BVP y conductividad de la piel, utilizándolas para la generación de características que permitan generar modelos para clasificar estados emocionales y cognitivos de las personas. Las señales son utilizadas como SCR y HR.

En el estudio realizado en [50] utilizan la señal PPG para el cálculo de las variables ritmo cardiaco promedio y desviación estándar y tres bandas de frecuencia. Aquí desarrollan un clasificador de carga cognitiva para dos niveles, alto y bajo.

Por otro lado, en [110] utilizan la señal BVP de una manera más directa, sin la necesidad de cálculo de ritmo cardiaco. En este estudio se investiga el uso de BVP para la clasificación de niveles de carga cognitiva. Se encuentra que características como peak o max BVP fueron significativas para la identificación de niveles de carga cognitiva. Este enfoque será utilizado en el trabajo aquí expuesto.

Medidas basadas en comportamiento ocular (EyeTracker)

Según [26] la principal medida relacionada con la carga cognitiva corresponde al tamaño de la pupila. La respuesta pupilográfica ha sido estudiada en diferentes investigaciones. Por ejemplo en [11] se encontró que la dilatación pupilar de los alumnos era un indicador confiable de la carga cognitiva en el desarrollo de las tareas.

En otros casos, se ha estudiado las respuestas pupilares en otros estímulos, como lo son vídeos [60, 64, 28] , demostrando que la selección de características es un proceso fundamental para la clasificación de los niveles de carga de trabajo.

El tamaño de la pupila se ve afectado por otros factores externos a la carga cognitiva. Estos pueden ser los estados emocionales o la iluminación [105, 62]. En un resiente estudio [85] se presenta un enfoque para la medición de carga cognitiva independientemente de la luminosidad de la tarea desarrollada. Existen variados estudios relacionados a como afecta la iluminación en la predicción de la carga cognitiva. Sin embargo, dado que durante este trabajo se utiliza un experimento con iluminación controlada, no se incluyen dichos estudios en el actual estado del arte.

Por otro lado, existen otras medidas extraídas a partir de los datos obtenidos por el Eye Tracker. Medidas como la frecuencia, duración e intervalo de pestañeo [79, 2, 98, 101]; frecuencia y duración de las fijaciones [31, 67, 100]; distancia y velocidad de las sacadas [34, 33] pueden ser encontradas en otros estudios, mostrando utilidad en la generación de clasificadores para los niveles de carga cognitiva de los usuarios.

3.3. Predicción de secuencias en otros escenarios

En la literatura existen estudios relacionados a la predicción de secuencias en escenarios diferentes a lo que es la navegación en páginas web o a la predicción de movimiento ocular. En esta sección se nombrarán algunos estudios, dado que el enfoque de solución entregado para cada problema respectivo puede ser adaptado al problema tratado en el presente trabajo.

En [106] se estudia el movimiento de los transeúntes en la vía pública. Generan un modelo de predicción para el movimiento de las personas a partir de las imágenes adquiridas en videos de baja resolución. Para ello utilizan un enfoque de minimización de una función de energía que represente condiciones físicas, motivaciones personales e interacciones sociales (6 factores en total, combinados linealmente).

En [4] buscan predecir trayectorias de los humanos en ambientes con multitudes. Para ello, realizan dos métodos de predicción. El primero, "Social Forces model" en el cual la navegación de las personas se modela como una función diseñadas para el entendimiento del comportamiento social, de manera similar al enfoque usado en [106]). El segundo método utiliza los métodos de LSTM para la predicción de secuencias.

Para capturar las dependencias entre diferentes secuencias correlacionadas, realizan una conexión de secuencias cercanas, denominada "Social-LSTM". Aquí, cada persona tiene su patrón de movimiento, es decir velocidades, aceleraciones, y forma de caminar. Cada secuencia a predecir (cada persona) requiere del uso de LSTM, agregando un proceso de pooling preservando las características espaciales del vecindario.

Este último estudio presenta métodos de solución basados en lo expuesto en [41], donde se explica cómo funcionan las redes neuronales recurrentes y LSTM, para posteriormente exponer algunos ejemplos en los que son aplicados.

De los estudios anteriores se puede rescatar el uso de dos tipos de modelos en la predicción de secuencias, los modelos de minimización de energía y la predicción de secuencias a partir de redes LSTM, las cuales serán utilizadas en el presente trabajo.

Posteriormente se detalla el modelo para el segundo escenario, donde los siguientes estudios entregan algunas claves para su desarrollo.

En [41] se presenta el ejemplo del problema de "handwriting", donde se busca generar una secuencia para las posiciones de los caracteres dibujados por un usuario al escribir a mano. Dicha secuencia es caracterizadas por puntos del lápiz (o dedo), representando la posición, dada por las coordenadas x e y, además de un indicador de si el el lápiz se encuentra posado (proceso de escritura) o levantado (por ejemplo, al cambiar de una letra a la siguiente).

Utilizando redes neuronales recurrentes y LSTM, se logra generar resultados de distribuciones de probabilidad de los próximos puntos en la secuencia, a partir de los puntos anteriores. Es decir, a partir de los caracteres ya escrito y los patrones aprendidos en los conjuntos de entrenamiento, se predice las probabilidades de los próximos puntos a ser marcados.

En particular, se utiliza las denominadas mixture "density network". En la imagen 3.6 se presenta un ejemplo de resultados obtenidos con estos métodos. Se observa un mapa de calor que muestra la secuencia de distribuciones de probabilidad para las posiciones predichas al escribir la palabra "under". Las densidades para predicciones sucesivas son agregadas de manera conjunta.

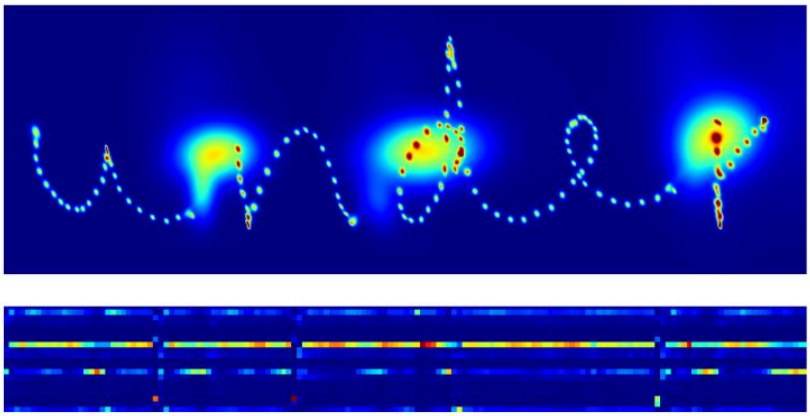


Figura 3.6: Resultados de mixture density para la predicción de escritura a mano.

En el estudio [108] se genera un modelo LSTM bidirecional con densidad de mezcla utilizado para predecir la trayectoria en los lanzamientos del balón realizados en diferentes partidos de la NBA. Aquí se explica una extensión al modelo de handwriting en cuanto a la arquitectura de la red recurrente LSTM.

Finalmente, el estudio [43] y sus códigos de programación, son usados como base para la estimación de resultados en el segundo escenario. En el paper original se busca la generación de bosquejos relacionados algún tema en concreto, como se muestra en la imagen 3.7. Para ello, establecen un modelo con arquitectura encoder-decoder, donde los resultados entregados por el decoder corresponden a los de un modelo de mezcla de densidades. Explicados en las siguientes secciones.

Se utilizan diferentes dibujos realizados a mano como datos de entrenamiento. Los modelos entrenados estiman las distribuciones gaussianas necesarias para a partir de alguna semilla entregada en el encoder, necesarias para generar muestreos, que corresponden a los bosquejos entregados por el decoder.

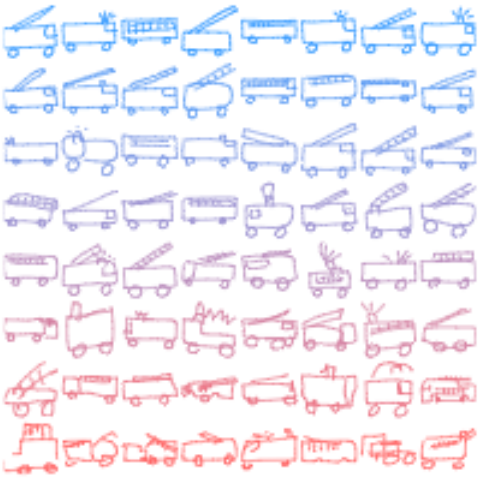


Figura 3.7: Generación incondicional de camiones de bomberos.

Capítulo 4

Marco Teórico

4.1. Clasificación con múltiples etiquetas

La clasificación con múltiples etiquetas, o en inglés, multilabel classification, es una tarea predictiva, que ha sido utilizada en áreas como, por ejemplo, bio-informática, buscando relacionar cada proteína con diferentes funcionalidades que desempeñan en un sistema biológico; Text Mining, desarrollando la categorización de textos; o Retrieval image, donde se busca relacionar cada imagen con múltiples conceptos.

4.1.1. Definición formal

Los métodos de clasificación son herramientas predictivas usadas en Data mining. Lo que intentan es predecir alguna variable, denominada etiqueta (label en inglés), la cual corresponde a una única clase. En particular, si el modelo posee tan solo dos clases posibles (0 o 1), se habla de clasificación binaria. Por ejemplo, predecir la fuga de un cliente en función de sus antecedentes e historial de compra, es un modelo de clasificación binaria con clases "sí se fuga" y "no se fuga".

Para lograr la predicción de la etiqueta se utilizan modelos con diferentes configuraciones, o algoritmos basados en la instancia. El conjunto de datos consta de dos partes: Variables predictivas y las etiquetas o variables dependientes.

La generalización del modelo de clasificación binario es la clasificación en múltiples clases, donde la etiqueta puede tomar más de dos posibles valores, por ejemplo, la identificación de especies de flor dada sus características. Sin embargo, una limitación importante que presentan estos modelos es que en cada instancia, la etiqueta atribuida solo puede pertenecer a una familia, por ejemplo sólo un tipo de flor, siendo valores mutuamente excluyentes.

A diferencia de la clasificación multi-clases, la clasificación con multi-etiquetas, asocia diferentes etiquetas a clases binarias. Siguiendo la definición formal dada en [45]:

Definición 4.1 *Sea* $\mathcal{X} = A1 \times A2 \times ... \times Af$ *un espacio de entrada, donde* f *es el número de conjuntos de atributos de entrada y* $A1, A2, ...Af$ *conjuntos numerables. Sea un elemento*

$X \in \mathcal{X}$.

Definición 4.2 *Sea $\mathcal{L}$ el conjunto de las posibles multi-etiquetas, es decir, vectores donde para cada coordenada se entrega una etiqueta. $\mathcal{P}(\mathcal{L})$ es el conjunto de potencia de $\mathcal{L}$, que contiene todas las posibles combinaciones $l \in \mathcal{L}$ incluyendo el conjunto vacío. $k = |\mathcal{L}|$ es el número total de etiquetas.*

Definición 4.3 *Sea $\mathcal{Y}$ el conjunto de salida, con todos los posibles vectores $Y \in \mathcal{P}(\mathcal{L})$, siempre de largo k.*

Definición 4.4 *Sea $\mathcal{D}$ el conjunto de datos que contiene una cantidad finita de subconjuntos de $A1 \times A2 \times ... \times Af \times \mathcal{P}(\mathcal{L})$. Cada elemento $(X, Y) \in \mathcal{D} | X \in A1 \times A2 \times ... \times Af, Y \in \mathcal{P}(\mathcal{L})$ será una instancia. $n = |\mathcal{D}|$ será el número de elementos en $\mathcal{D}$.*

Definición 4.5 *Se define $\mathcal{F}$ un clasificador multi-etiqueta, definido como $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$. La entrada de $\mathcal{F}$ son instancias $X \in \mathcal{X}$, y el ouput será una predicción $Z \in \mathcal{Y}$. De esta forma, la predicción del vector de etiquetas asociado a una instancia se obtiene como $Z = \mathcal{F}(X)$.*

4.1.2. Feature Selection

Uno de los problemas más comunes en el área de machine learning corresponde a la alta dimensionalidad. La alta dimensionalidad en las variables independientes provoca modelos más complejos, afectando también el tiempo requerido para su entrenamiento, y en algunos casos menos efectivos (menos generalizables a nuevos conjuntos de datos). Por lo tanto, se presenta brevemente los métodos de selección de variables utilizados durante el trabajo.

MLMIM y MLJMI

Las siglas en inglés corresponden a "Maximization of mutual information" (MLMIM) y "Join mutual information" (MLJMI). Según lo expuesto en [91], se considera que las instancias son generadas por el proceso $p : \mathcal{X} \to \mathcal{Y}$, el cual es aproximado por la predicción $\mathcal{F}$. Sea θ un vector binario que indica las variables a considerar del total presente en $\mathcal{X}$, $\bar{\theta}$ el complemento (variables no utilizadas) y τ los parámetros del modelo de predicción $\mathcal{F}$, entonces la verosimilitud ($\mathcal{L}$) y la log-verosimilitud (l) se expresan de la forma:

$$\mathcal{L}(\theta, \tau, y|x) = \prod_{\mathrm{i}=1}^{n} \mathcal{F}(Y^{\mathrm{i}}|{X_\theta}^{\mathrm{i}}, \tau) \Leftrightarrow l(\theta, \tau, Y|X) = \frac{1}{n} \sum_{\mathrm{i}=1}^{n} \mathcal{F}(Y^{\mathrm{i}}|{X_\theta}^{\mathrm{i}}, \tau)$$

Según Brown [20] la log-verosimilitud se puede descomponer en tres términos, uno que depende del ajuste de la predicción al proceso real, el segundo que dependerá de la elección de variables y un término irreductible, según la siguiente fórmula [1]:

$$\lim_{n\to\infty} (-l) = \mathbb{E}_{XY} \left\{ \log \frac{p(Y|X_\theta)}{\mathcal{F}(Y|X_\theta, \tau)} \right\} + I(X_{\bar{\theta}}; Y|X_\theta) + H(Y|X)$$

Bajo la hipótesis de que la selección de variables y la clasificación son independientes, θ^* que maximiza la verosimilitud $\mathcal{L}$ se puede calcular minimizando el término I. Utilizando un

[1] ver detalles en el paper original.

algoritmo greedy, se busca maximizar la información mutua condicional (CMI en inglés):

$$J_{CMI}(X_a) = I(X_a; Y|X_\theta), X_a \in X_{\bar{\theta}}$$

Entonces, los modelos multilabel (ML) pueden reducir su dimensionalidad en cuanto a variables mediante la maximización de la información mutua (MIM) y la información mutua conjunta (JMI), las que corresponden a una aproximación de CMI definidas como:

$$J_{MIM}(X_a) = I(X_a; Y) \qquad J_{JMI}(x_a) = \sum_{j=1}^{|X_\theta|} I(X_{\theta_j} X_a; Y)$$

Se observa que la maximización de CMI no asume ningún tipo de independencia entre etiquetas, JMI lo hace parcialmente y MIM asume independencia de forma , permitiendo considerar las variables relevantes pero sin considerar redundancia.

MLMRMR

Según [84] "Mínima redundancia máxima relevancia" es un algoritmo de selección incremental de primer orden, que escoge las variables a considerar de manera secuencial. Permite un alto ajuste y demuestra una alta velocidad de computo.

Este algoritmo entrega una forma de maximizar la dependencia sin tener que estimar probabilidades multivariable. No busca variables que sean completamente independientes, ya que esto puede entregar peores resultados, sólo minimiza la redundancia seleccionando variables con mayor relevancia con respecto a las etiquetas.

Es decir, en vez de calcular directamente la máxima dependencia, dada por la ecuación:

$$\text{máx}\, D(S, Y),\ D = I(X\text{i},\ \text{i} = 1...m;\ Y)$$

donde S es un subconjunto de $\mathcal{X}$ con m características que se busca encontrar, $\{X_\text{i}\}_{\text{i}=1,...,m}$, el algoritmo la aproxima maximizando la relevancia:

$$\text{máx}\, D(S, Y),\ D = \frac{1}{|S|} \sum_{X_\text{i} \in S} I(X_\text{i};\ Y)$$

y al mismo tiempo, minimizando la redundancia entre las variables seleccionadas:

$$\text{mín}\, R(S),\ R = \frac{1}{|S|^2} \sum_{X_\text{i} X_j \in S} I(X_\text{i}, Xj)$$

MIFS

Según [51] la "selección de características informada" es un método que presenta principalmente dos propiedades. Hace uso de la semántica latente de la multi-etiqueta para guiar la fase de selección de características y hace uso de la correlación entre las características resultantes a través de las diferentes etiquetas.

En los problemas de clasificación multilabel, cada instancia es atribuida con diferentes clases, las cuales pueden estar correlacionadas. Usando esta información sería posible encontrar

características comunes para etiquetas fuertemente relacionadas y diferentes características para etiquetas menos correlacionadas.

El método consiste principalmente en la descomposición del espacio de etiquetas Y, como el producto de dos matrices, permitiendo la explicación y reconstrucción de Y. De esta manera la matriz semántica latente que presenta una menor dimensionalidad demuestra la correlación entre etiquetas y reduce significativamente el ruido presente en el conjunto original.

Es en el espacio de descomposición del espacio original donde se realiza la selección de variables. En particular se busca resolver un problema de optimización que permita la descomposición y selección características simultáneamente.

RFS

Según [77] la "selección robusta de características" (en inglés, RFS) es un método de selección de variables basado en la minimización de la norma $\ell_{2,1}$ en la función de pérdida y regularización, siendo un método eficiente en cálculo y robusto ante la presencia de outliers.

En general la norma $\ell_{r,p}$ de una matriz M se define como:

$$\|M\|_{r,p} = \left(\sum_{\mathrm{i}=1}^{n} \left(\sum_{j=1}^{m} |m_{\mathrm{i}j}|^r \right)^{\frac{p}{r}} \right)^{\frac{1}{p}}$$

La regularización de la norma $\ell_{2,1}$ es usada para la selección de variables a través de todos los puntos con dispersión conjunta. El algoritmo optimiza una función de pérdida similar al problema de optimización resuelto por método de clasificación de mínimos cuadrados, pero sin el término al cuadrado, lo que le permite mayor robustez ante la presencia de outliers, más un término de regularización para el cálculo de la matriz de proyección $W \in \mathbb{R}^{\mathrm{d}\times c}$:

$$\min_{W} J(W) = \sum_{\mathrm{i}=1}^{n} \left\| W^\top X_{\mathrm{i}} - Y_{\mathrm{i}} \right\|_2 + \gamma R_4(W) = \left\| X^\top W - Y \right\|_{2,1} + \gamma \left\| W \right\|_{2,1}$$

4.1.3. Métricas

En general, el desempeño de los diferentes algoritmos de clasificación multi-etiqueta existentes puede variar según la naturaleza de los datos utilizados. Por esta razón, una práctica común corresponde a la evaluación de diferentes algoritmos y configuraciones para la elección de el de mejor desempeño. A continuación se definen las métricas utilizadas al momento de elegir un modelo de clasificación.

Definición 4.6 (Hamming loss) *Cuenta el número de predicciones perdidas, normalizadas por el número de etiquetas y la cantidad de instancias* [2].

$$HammingLoss = (1 - HammingScore) = \frac{1}{n}\frac{1}{k}\sum_{\mathrm{i}=1}^{n} |Y_{\mathrm{i}} \triangle Z_{\mathrm{i}}|$$

[2]El operador $\triangle$ corresponde a la diferencia simétrica entre la etiquetas reales y las predichas.

Definición 4.7 (Accuracy) *Es la proporción entre el número de etiquetas predichas correctamente y las etiquetas realmente activas, en el conjunto de etiquetas real y predicho. Se promedia a través de todas las instancias.*

$$Accuracy = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}\frac{|Y_\mathrm{i} \cap Z_\mathrm{i}|}{|Y_\mathrm{i} \cup Z_\mathrm{i}|}$$

Definición 4.8 (Subset Accuracy) *Porcentaje de instancias correctamente etiquetadas según la comparación del número total de etiquetas* [3].

$$SubsetAccuracy = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}[\![Y_\mathrm{i} = Z_\mathrm{i}]\!]$$

Definición 4.9 (Precision) *Se calcula como la proporción entre el número de etiquetas correctamente etiquetadas y el número total de etiquetas.Intuitivamente corresponde al porcentaje de etiquetas predichas como verdaderas y que son realmente importantes para la instancia.*

$$Precision = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}\frac{|Y_\mathrm{i} \cap Z_\mathrm{i}|}{|Z_\mathrm{i}|}$$

Definición 4.10 (Recall) *Porcentaje de etiquetas predichas correctamente a lo largo de todas las etiquetas realmente relevantes.*

$$Recall = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}\frac{|Y_\mathrm{i} \cap Z_\mathrm{i}|}{|Y_\mathrm{i}|}$$

Definición 4.11 (F-measure) *Es el promedio entre Precision y Recall calculado de forma armónica. Mide cuantas etiquetas relevantes son predichas y cuantas de las etiquetas predichas son relevantes.*

$$F - measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Definición 4.12 (Coverage) *Cuenta el número de pasos realizados a través de las etiquetas ordenadas según ranking, necesarios para encontrar una etiqueta relevante. Aquí x_i corresponde a una instancia.*

$$Coverage = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}\arg\max_{y \in Y_\mathrm{i}}\langle rank(x_\mathrm{i}, y)\rangle - 1$$

Definición 4.13 (One Error) *Porcentaje de casos en el cual la etiqueta mas confiable para el clasificador es un falso positivo.*

$$OneError = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}[\![[\arg\max_{y \in Z_\mathrm{i}}\langle rank(x_\mathrm{i}, y) \notin Y_\mathrm{i}]]\!]$$

[3]el operador $[\![]\!]$ corresponde al corchete de Iverson (1 en caso de que la proposición lógica es verdadera, 0 si no)

Definición 4.14 (ranking Loss) *Para cada instancia cuenta las veces que una etiqueta no relevante es rankeada sobre una relevante, según lo entregado por el clasificador. Esta cuenta es normalizada por el número de etiquetas relevantes y no relevantes y promediadas por el numero de instancias evaluadas.*

$$RankingLoss = (1 - RankingScore) = \frac{1}{n}\sum_{\mathrm{i}=1}^{n}\frac{1}{|Y_\mathrm{i}||\bar{Y}_\mathrm{i}|}|y_a, y_b : rank(x_\mathrm{i}, y_a), (y_a, y_b) \in Y_\mathrm{i} \times \bar{Y}_\mathrm{i}|$$

Definición 4.15 (Discounted Cumulative Gain) *Las etiquetas relevantes apareciendo con menor ranking son penalizados proporcionalmente a la posición de la etiqueta. La ganancia acumulada descontada (DCG) es normalizada por el DCG ideal (IDCG), es decir el máximo valor de DCG posible para una posición p, según la fórmula de nDCG. |REl| representa la cantidad de etiquetas relevantes.*

$$DCG_p = \sum_{\mathrm{i}=1}^{p}\frac{rank(x_\mathrm{i}, y)}{\log_2(\mathrm{i}+1)} \qquad IDCG_p = \sum_{\mathrm{i}=1}^{|REL|}\frac{2^{rank(x_\mathrm{i}, y)} - 1}{\log_2(\mathrm{i}+1)} \qquad nDCG_p = \frac{DCG_p}{IDCG_p}$$

Definición 4.16 (Macro-F1 y Micro-F1) *F-Measure y otras métricas pueden ser calculadas como un promedio utilizando diferentes estrategias. En inglés se conocen como macro-averaging y micro-averaging.Dado EvalMet la métrica a calcular, según la cantidad de predicciones verdaderas positivas (TP), falsas positivas (FP), verdaderas negativas (TN) y falsas negativas (FN), las fórmulas a utilizar son:*

$$MacroMet = \frac{1}{k}\sum_{l\in\mathcal{L}} EvalMet(TP_l, FP_l, TN_l, FN_l)$$

$$MicroMet = EvalMet(\sum_{l\in\mathcal{L}} TP_l, \sum_{l\in\mathcal{L}} FP_l, \sum_{l\in\mathcal{L}} TN_l, \sum_{l\in\mathcal{L}} FN_l)$$

Definición 4.17 (Área under curve) *Una curva Receiver operating characteristic (ROC) demuestra la habilidad de un clasificador para discriminar entre positivo y negativo, al cambiar los valores de threshold. Un área bajo la curva de 1 representa una predicción perfecta. Su calculo en el caso multilabel corresponde a:*

$$MacroAUC = \frac{1}{k}\sum_{l\in\mathcal{L}}\frac{|x', x'' : rank(x', y_l) \geq rank(x'', y_l), (x', x'') \in X_l \times \bar{X}_l|}{|X_l||\bar{X}_l|}$$

$$X_l = \{x_\mathrm{i}|y_l \in Y_\mathrm{i}\}, \bar{X}_l = \{x_\mathrm{i}|y_l \notin Y_\mathrm{i}\}$$

4.1.4. Enfoques de clasificación

Binary relevance

Una de las formas más simples de clasificación multi-etiqueta consiste en considerar una etiqueta a la vez y entrenar un clasificador binario que indique una predicción de etiqueta activa o inactiva (1 o 0 respectivamente). Los resultados de cada clasificador son combinados para generar el conjunto de datos predicho para cada instancia.

Aunque es un método simple, es un método que puede presentar problemas. El entrenar clasificadores binarios de forma independiente no considera las posibles correlaciones entre las etiquetas y, al mismo tiempo, cada clasificador puede presentar un conjunto de datos de entrenamiento extremadamente desbalanceado.

Classification Chain

Es un método basado en un conjunto de clasificadores binarios. Aleatoriamente se escoge un orden para las etiquetas, a utilizar como forma para entrenar los clasificadores. El primer clasificador es entrenado usando sólo atributos de entrada. El resultado de este clasificador es agregado al conjunto de atributos y el nuevo espacio de entrada es utilizado para entrenar al segundo clasificador. Así sucesivamente.

Se puede observar que a diferencia del método de relevancia binaria, este método permite considerar las posibles dependencias entre etiquetas. Por otro lado, como el orden de la cadena de clasificadores es aleatorio, los resultados pueden variar según el patrón de etiquetas utilizado.

PowerLabelset

El método de transformación PL (power labelset) considera que cada combinación entre las k etiquetas es una clase, con lo cual la predicción corresponde a un patrón completo entre todas las etiquetas. Es decir, se utiliza cualquier método de clasificación multilabel para predecir la variable.

La mayor desventaja corresponde a que las posibles combinaciones para estas son un total de 2^k elementos, por lo que en conjuntos de datos muy complejos puede ser un número demasiado grande. Por otro lado, el resultado es directo (sin necesidad de combinaciones) y al mismo tiempo este método considera la relación existente entre etiquetas.

4.1.5. Métodos de clasificación multi-etiqueta

Ridge Regression

En la regresión de mínimos cuadrados se tiene un conjunto de variables, agrupados en la matriz X para la predicción de algún valor y. Esto se logra mediante la minimización del cuadrado de los errores entre las predicciones y los valores reales en el conjunto de entrenamiento, donde la función de predicción corresponde a $F(X) = \beta X$ y así $y = F(X) + \varepsilon$. En el caso de Ridge Regression se restringen los valores posibles para los coeficientes β's esperando valores pequeños pero no nulos, es decir, sin considerar variables irrelevantes. Para ello, se busca minimizar:

$$\min_{\beta} \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Así, se logra una predicción que a pesar de ser sesgada (no así, la regresión lineal común), posee menor varianza en las estimaciones de los coeficientes β. Esto permite mejor desempeño ante variables que pueden tener fuerte correlación. Los resultados obtenidos serán valores continuos entre el intervalo $[0, 1]$ los cuales son transformados a las clases $\{0, 1\}$ mediante algún valor de threshold, por ejemplo, asignando 1 a los valores mayores a 0,5.

El método de estimación de parámetros corresponde a fijar diferentes valores de λ para la posterior estimación de β. A mayores valores de λ el segundo término en la minimización posee mayor importancia, por lo que los coeficientes β tenderán a ser menores.

KNN

K-Nearest Neighbors, o en español, K-vecinos más cercanos, es un método en el cual los nuevos datos, presentes en el conjunto de prueba, son clasificados en función de los K datos presentes en el conjunto de entrenamiento que se encuentren más cercanos, basados en una medida de similitud. La medida utilizada corresponde a la norma euclideana (L2):

$$\mathrm{d}(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2}$$

Así, la clasificación asignada a un nuevo elemento corresponde a la clase mayoritaria entre los K vecinos seleccionados. En este modelo por lo tanto, existe sólo un parámetro a considerar, que corresponde al número de vecinos que serán evaluados para la asignación de clases a los nuevos elementos (K).

SVM

Support vector machine, realiza una separación mediante un hiperplano que permita clasificar los nuevos registros de datos según su posición. Si el nuevo dato se encuentra sobre el hiperplano se le asigna valor 1 y bajo el hiperplano, el valor -1. En nuestro caso se utiliza una regularización L2 y una función de pérdida L2 con lo cual se busca minimizar la función:

$$\min_{w} \frac{1}{2} w^\top w + C \sum_{\mathrm{i}=1}^{l} \max\left(0, 1 - y_\mathrm{i} w^\top x_\mathrm{i}\right)^2$$

Los parámetros w encontrados permiten una estimación a través de la función $sing(w^\top x_\mathrm{i})$ la cual entrega valor 1 en caso de que el signo de $w^\top x_\mathrm{i}$ sea positivo y -1 en caso negativo, para cada muestra de datos en el conjunto de entrenamiento, indexado por i. En realidad, a la función de predicción se le agrega un sesgo $b = 2$, con lo cual $w^\top x_\mathrm{i} + b$ puede ser escrito como $w^\top x_\mathrm{i}$ si es que $x_\mathrm{i} = [x_\mathrm{i}|b]$.

El primer término representa la regularización L2, es decir, los parámetros estimados w son penalizados cuadráticamente (se busca valores pequeños a través de la minimización). Este término en realidad lo que busca es maximizar $1/\|w\|$ que corresponde a la separación entre ambas clases presentes en el conjunto de entrenamiento. El segundo término corresponde a la función de pérdida, ponderada por la constante C, la cual le otorga un peso mayor o menor a este término. Representa la normalización L2 en la función de pérdida, ya que el error, es decir, la diferencia entre el valor predicho y el valor real de las clases en los datos de entrenamiento, se minimiza cuadráticamente.

Se utiliza regularización L2 y una función de pérdida L2 ya que es la más común en la literatura. La función de pérdida L2 es menos robusta, pero entrega una solución única la cual es más estable que el caso L1 (término minimizado a través del valor absoluto). La regularización L2 es computacional-mente más eficiente y dado que la selección de variables se realiza en un proceso anterior, puede usarse.

MLKNN

Multi-label K nearest neighbors, es un método basado en KNN. Sigue el mismo principio, es decir, para una nueva muestra se identifican las K muestras más cercanas en el conjunto de entrenamiento y luego, basado en la información estadística de las etiquetas entregada por esos K vecinos, osea las clases a la cual pertenece cada uno, se utiliza el principio de máxima probabilidad a posteriori para determinar las etiquetas de esta nueva instancia. Este método es usado directamente, por lo que a diferencia de la división en subproblemas realizado con Binary Relevance o Classification Chain, se consideran las posibles correlaciones existentes entre las diferentes etiquetas.

Para una nueva muestra de testeo t, se define $y_t(l)$ como la clasificación de la etiqueta l, pudiendo tomar valores 0 o 1. Sea H_b^l el evento de que la etiqueta l tome el valor $b = \{0, 1\}$ para la muestra t y $E^l_{C_t(l)}$ el evento en que existan $C_t(l)$ vecinos con la etiqueta l activa para la muestra t. Entonces $y_t(l)$ queda definido por el argumento que maximiza la probabilidad:

$$\begin{aligned} y_t(l) &= \arg\max_{b\in\{0,1\}} P(H_b^l | E^l_{C_t(l)}), \qquad l \in \mathcal{Y} \\ &= \arg\max_{b\in\{0,1\}} \frac{P(H_b^l) P(E^l_{C_t(l)} | H_b^l)}{P(E^l_{C_t(l)})} \\ &= \arg\max_{b\in\{0,1\}} P(H_b^l) P(E^l_{C_t(l)} | H_b^l) \end{aligned}$$

Las últimas dos probabilidades se obtienen de los cálculos a priori realizados mediante frecuencia en los datos de entrenamiento. Es decir, el primer término representa la fracción de elementos con la etiqueta l activa en el conjunto de entrenamiento y el segundo la fracción de muestras con $y_t(l) = b$ que poseen $C_t(l)$ vecinos con dicha etiqueta activa.

Al cálculo de frecuencias realizado en el conjunto de entrenamiento se le agrega el parámetro *smooth*, el cual controla la fuerza de la probabilidad a priori. Un valor de 1 en este parámetro permite la suavización de Laplace. Valores superiores implican acercar las probabilidades a una distribución uniforme, lo que permite probabilidades no nulas. De esta forma, el algoritmo almacena probabilidades calculadas a partir de los datos de entrenamiento y, para nuevas instancias, se aplican estos valores almacenados para la asignación de las clases a cada etiqueta.

4.2. Métodos de Generación de Clusters

Según [38] la clusterización de datos (también llamada análisis de segmentación, análisis de taxonomía o clasificación no supervisada) es un conjunto de métodos usados para crear grupos de objetos en los cuales, los objetos presentes en el mismo cluster son muy similares y presentan gran diferencia respecto a los elementos de otros clusters. Se diferencian de los métodos de clasificación principalmente en que estos últimos vinculan los objetos en clases ya definidas, en cambio, en la clusterización las clases serán parte del resultado.

4.2.1. Definición formal

Matemáticamente, cada observación x_k consistente de n variables medibles se representa por un vector n-dimensional $x_k = [x_{k1}, x_{k2}, ..., x_{kn}]^\top \in \mathbb{R}^n$.El conjunto de N observaciones denominado $X = \{x_k | k = 1, 2, ..., N\}$ tiene la forma:

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \dots & x_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

Los métodos de clusterización, entregan como resultado una matriz U, llamada matriz de partición, en la cual cada entrada u_{ik} corresponde a la membresía del objeto i al cluster k:

$$U = \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1c} \\ u_{21} & u_{22} & \dots & u_{2c} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N1} & u_{N2} & \dots & u_{Nc} \end{bmatrix} \in \mathbb{R}^{N \times c}$$

Dependiendo de la forma que presenta el resultado U, los métodos de clusterización se pueden clasificar en métodos Hard Partition y Fuzzy Partition.

Hard Partition Cluster

En términos simples, este tipo de algoritmos entrega como solución una partición en la cual, cada objeto i $= 1, ..., N$ es atribuido sólo a un cluster. Por lo tanto, una partición hard será una familia de subconjuntos $\{A_i | 1 \leqslant i \leqslant c \subset P(X)\}$ que satisfaga las restricciones:

$$\bigcup_{i=1}^{c} A_i = X$$

$$A_i \cap A_j = \phi, 1 \leqslant i \neq j \leqslant c$$

$$\phi \subset A_i \subset X, 1 \leqslant i \leqslant c$$

Escribiendo las restricciones desde las perspectivas de la matriz de partición U, la restricción 4.1 establece que solo existen las posibilidad de pertenencia o no de un objeto a un cluster. La restricción 4.2 refleja que cada objeto solo puede pertenecer a un cluster. La restricción 4.3 dice que no pueden existir cluster sin objetos.

$$u_{ij} \in \{0, 1\}, \qquad 1 \leqslant i \leqslant N, 1 \leqslant j \leqslant c \tag{4.1}$$

$$\sum_{k=1}^{c} u_{ik} = 1, \qquad 1 \leqslant \mathrm{i} \leqslant N \tag{4.2}$$

$$0 < \sum_{\mathrm{i}=1}^{N} u_{ik} < N \tag{4.3}$$

Así el espacio de particiones hard queda definido por $M_{Hc} = \{U \in \mathbb{R}^{N \times c} | (4,1), (4,2), (4,3)\}$.

Fuzzy Partition Cluster

En términos simples, este tipo de algoritmos relaja la condición de que cada objeto pertenezca a un solo cluster, con lo cual, la restricción (4.1) es reemplazada por:

$$u_{ij} \in [0, 1], \qquad 1 \leqslant \mathrm{i} \leqslant N, 1 \leqslant j \leqslant c \tag{4.4}$$

Así, los términos de la matriz de partición pueden entenderse como el grado de vinculación existente entre cada objeto y cada cluster. Intuitivamente, un elemento en la frontera de dos cluster tendrá membresía similar para ambos, perteneciendo en igual medida a cada uno de ellos. Es posible convertir una partición difusa en una partición Hard, considerando para cada objeto la pertenencia al cluster con mayor membresía, teniendo cuidado de respetar las restricciones antes mencionadas.

Así el espacio de particiones difusas se define por $M_{Fc} = \{U \in \mathbb{R}^{N \times c} | (4,4), (4,2), (4,3)\}$.

4.2.2. Algoritmos utilizados

Durante el presente trabajo se requiere la utilización de ciertos métodos de clasificación no supervisado, de los cuales se consideran sólo los métodos de Hard Partitioning, debido a la simpleza de las soluciones que estos entregan. A continuación se definen de manera muy general los dos métodos usados: Kmean y Kmeiods.

K-mean: Este es un método en el cual, dado el conjunto de objetos, se genera un total de k clusters. El algoritmo recibe el número de clusters a utilizar (k) y genera los centros de manera aleatoria, o también se le puede entregar directamente los centros iniciales. Dado esto, se puede asignar a cada cluster, los objetos más cercanos, dependiendo de la medida de distancia utilizada. En nuestro caso usaremos la distancia euclideana.

el algoritmo recalcula nuevos centros a partir de los objetos que componen cada uno de los k clusters de manera iterativa. Para la actualización de los centros en cada iteración, se utiliza la media de los objetos presentes en el cluster:

$$v_{\mathrm{i}} = \frac{\sum_{j=1}^{N_{\mathrm{i}}} x_j}{N_{\mathrm{i}}}, \qquad x_j \in A_{\mathrm{i}}$$

Tras varias iteraciones se mejora el resultado de la minimización de la suma de cuadrados de los errores dentro de los clusters:

$$\text{mín} \sum_{\mathrm{i}=1}^{K} \sum_{j \in A_{\mathrm{i}}} \|x_j - v_{\mathrm{i}}\|_2$$

K-meiods: Es similar al método de K-means, sin embargo, el centro de los clusters es actualizado no por el promedio de los objetos que contiene, sino que por el objeto más cercano a dicho valor. Esto permite que este método sea más robusto ante posibles outliers en los datos.

4.2.3. Métricas de evaluación

Para las diferentes combinaciones de clusterización posibles, se debe evaluar cual alternativa es mejor. A continuación se definen tres métricas utilizadas en este trabajo. SC y S son útiles para comparar particiones con igual número de clusters. Para mejorar la partición se busca que las tres métricas sean del menor valor posible.

Definición 4.18 (Índice de partición (SC)) *Es el ratio entre la suma de compactación y separación de los clusters. Es una suma de medidas de validación individuales de cada cluster normalizadas por la cardinalidad de cada cluster.*

$$SC(k) = \sum_{\mathrm{i}=1}^{k} \frac{\sum_{j=1}^{N} (u_{\mathrm{i}j})^m \|x_j - v_{\mathrm{i}}\|^2}{N_{\mathrm{i}} \sum_{c=1}^{k} \|v_c - v_{\mathrm{i}}\|^2}$$

Definición 4.19 (Índice de separación (S)) *Es similar al índice de partición, solo que utiliza la distancia mínima para validar la partición.*

$$S(k) = \frac{\sum_{\mathrm{i}=1}^{k} \sum_{j=1}^{N} (u_{\mathrm{i}j})^2 \|x_j - v_{\mathrm{i}}\|^2}{N \min_{\mathrm{i},c} \|v_c - v_{\mathrm{i}}\|^2}$$

Definición 4.20 (Índice de Xie y Beni (XB)) *Cuantifica el ratio entre la variación total dentro de los clusters y la separación entre clusters.*

$$XB(k) = \frac{\sum_{\mathrm{i}=1}^{k} \sum_{j=1}^{N} (u_{\mathrm{i}j})^m \|x_j - v_{\mathrm{i}}\|^2}{N \min_{\mathrm{i},j} \|x_j - v_{\mathrm{i}}\|^2}$$

Definición 4.21 (Índice de Dunn (DI)) *Es un índice propuesto para la identificación de los clusters compactos y bien separados, con varianza pequeña entre los miembros de un mismo cluster. En palabras simples corresponde a la distancia mínima entre clusters relativo al máximo diámetro de los clusters. Para un mismo número de clusters, un valor mayor es mejor. Uno de sus problemas es el costo computacional al aumentar el numero de clusters y la dimensionalidad de los datos.*

$$DI(c) = \min_{\mathrm{i} \in c} \{ \min_{j \in c, \mathrm{i} \neq j} \{ \frac{\min_{x \in C_{\mathrm{i}}, y \in C_j} \mathrm{d}(x, y)}{\max_{k \in C} \{ max_{x,y \in C} \mathrm{d}(x, y) \}} \} \}$$

Definición 4.22 (Índice de Dunn alternativo (ADI)) *Es similar al Indice de Dunn, con simplificaciones computacionales en sus cálculos, donde la distancia mínima entre dos clusters es reemplazada por la desigualdad triangular (donde v_j es el centro del cluster j)*

$$\mathrm{d}(x, y) \geq |\mathrm{d}(y, v_j) - \mathrm{d}(x, v_j)|$$

$$ADI(c) = \min_{\mathrm{i} \in c} \{ \min_{j \in c, \mathrm{i} \neq j} \{ \frac{\min_{x \in C_{\mathrm{i}}, y \in C_j} |\mathrm{d}(y, v_j) - \mathrm{d}(x_{\mathrm{i}}, v_j)|}{\max_{k \in C} \{ max_{x,y \in C} \mathrm{d}(x, y) \}} \} \}$$

4.3. Redes Neuronales

En esta sección se explica brevemente los modelos de redes neuronales utilizados para la predicción de secuencias, en ambos escenarios. En un comienzo se muestra la red más básica correspondiente a la red neuronal recurrente (RNN) y posteriormente modelos más complejos. Lo expuesto en este capítulo no presenta una formulación matemática detallada, ya que esta se presenta posteriormente en el capítulo 6.

Las descripciones de los modelos RNN y LSTM se basan en la explicación entregada en los estudios [40, 27]. La explicación del modelo MDN se obtuvo de [15]. Finalmente, la integración de redes recurrentes con MDN se detalla en [41].

4.3.1. Recurrent Neural Network (RNN)

Este tipo de redes neuronales es utilizado en tareas que involucren el tratamiento de data secuencial, por ejemplo, predicción o clasificación. A pesar de su invención hace varias décadas, actualmente presentan mayor utilización en tareas como reconocimiento de escritura a mano, reconocimiento de audio, transformación de imágenes a texto, entre otras aplicaciones.

En el día a día existen muchos datos de forma secuencial, por ejemplo, las series de tiempo de diferentes procesos, los diferentes frames en un video, o las palabras usadas para la descripción de una imagen. Todos estos ejemplos se pueden representar de una forma $x = \{x_1, x_2, \ldots, x_\tau\}$, donde x_t corresponde a un valor en la serie de tiempo, una imagen dentro de un video o una palabra dentro de la frase, en los ejemplos antes mencionados.

Los datos secuenciales poseen el atributo de que para un paso t, existe cierta correlación con los valores anteriores de la secuencia. Por esta razón, lo que las redes neuronales recurrentes intentan capturar es la historia anterior a la predicción del próximo valor de la secuencia. Para ello utilizan los denominados estados ocultos, nombre heredado de las cadenas de markov oculto, modelos usados anteriormente para el manejo de datos secuenciales.

En la imagen 4.1 se observa la estructura de las redes neuronales recurrentes. Estas funcionan de la siguiente manera:

- Se genera un estado oculto inicial, el cual puede ser inicializado en zero, o calculado a partir de otros atributos.
- La red posee conexiones en cada periodo desde algún input $x^{(t)}$ para la generación de un nuevo estado oculto $h^{(t)}$. De igual forma, existe la conexión recurrente entre $h^{(t-1)}$ y $h^{(t)}$, es decir, la historia almacenada en los estados anteriores.
- Las redes neuronales recurrentes producen un output en cada paso de tiempo $o^{(t)}$ a partir de los estados ocultos $h^{(t)}$.
- La red es entrenada, es decir, calcula los parámetros a utilizar a partir de la función de pérdida L y el método backpropagation.

Las redes recurrentes generan los outputs a partir de los estados ocultos y no directamente de los inputs. De esta forma los outputs se generan en función de las correlaciones existentes con los pasos previos en la secuencia. Sin embargo, esto presenta las desventajas de que, en los

últimos pasos temporales de la red, la información de los estados ocultos puede ser confusa, y además, dado el largo de la red, provoca dificultades en la estimación de parámetros a partir de métodos que involucren el cálculo de los gradientes.

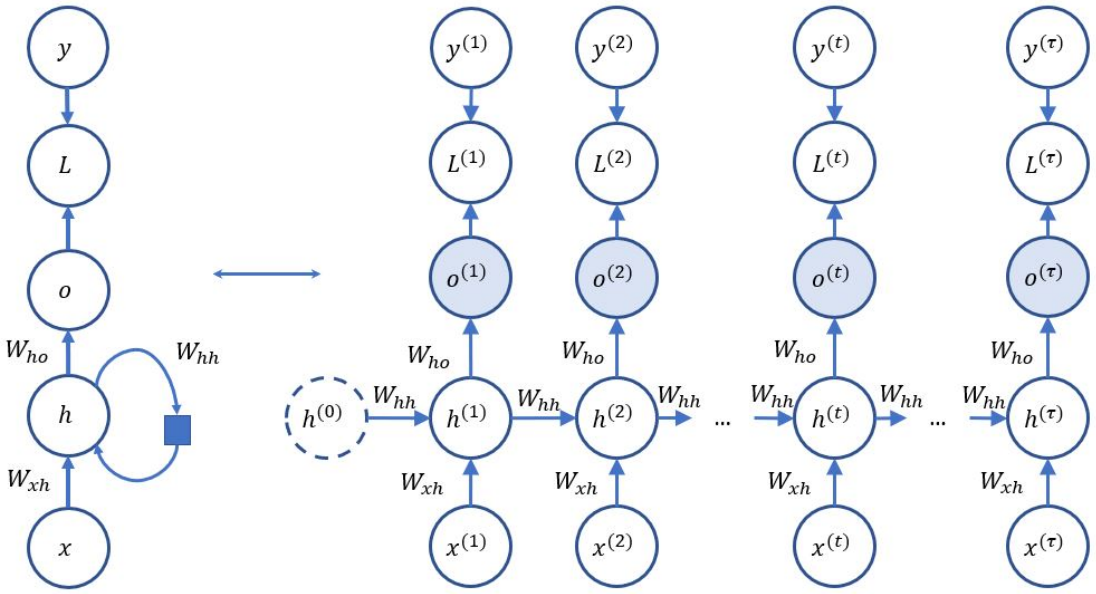


Figura 4.1: Red neuronal recurrente. A la izquierda se ve la estructura de la red neuronal recurrente, en la cual el nodo $h^{(t)}$ es calculado en función del estado oculto anterior $h^{(t-1)}$. A la derecha se observa la red desplegada, en la cual cada recurrencia fluye hacia la derecha desde un estado oculto inicial $h^{(0)}$ hasta un estado final $h^{(\tau)}$. En cada etapa t, se genera un output $o^{(t)}$ el cual relacionado a $y^{(t)}$ afecta a la función de pérdida $L^{(t)}$.

4.3.2. Long short term memory (LSTM)

Las redes Long short term memory (LSTM) mantienen la esencia de las redes recurrentes, donde en cada paso temporal, las predicciones son generadas en función de algún estado oculto que almacena la información de toda la secuencia predicha hasta el dicho paso. Sin embargo, sus mejoras permiten la adición de la información de memoria y al mismo tiempo se encarga del problema del desvanecimiento del gradiente usado en la estimación de parámetros en la etapa de backpropagation.

LSTM agrega las estructuras de celdas esquematizadas en la figura 4.2 para la predicción en cada etapa. Esta estructura posee un flujo a través de diferentes puertas, permitiendo mayor flexibilidad en la utilización de información del pasado.

LSTM es una red neuronal que trabaja con estados de celda c_t. Las puertas mencionadas anteriormente corresponden a "input" (i_t), "forget" (f_t), "modulation" ($\widetilde{c}_t$) y "output" (o_t). Todas ellas reciben como entrada $[h_{t-1}, x_t]$ y utilizan funciones de activación aplicada sobre combinaciones lineales de las entradas. Los valores generados son combinados posteriormente mediante productos, representados en la imagen por el símbolo $\odot$, y por sumas, para finalmente generar los valores del nuevo estado de la celda c_t y el nuevo estado oculto h_t.

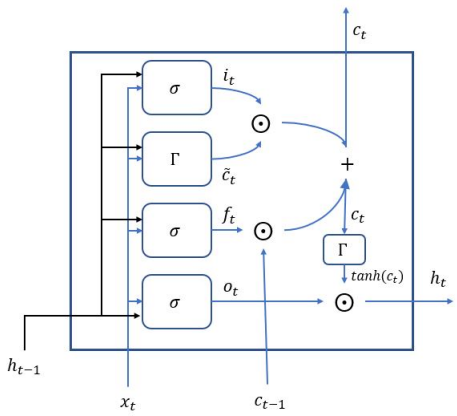


Figura 4.2: Celda en red Long short term memory (LSTM)

La compuerta f_t utiliza una función de activación sigmosoidal, por lo cual, genera valores entre $[0, 1]$ que indican que parte de la información presente en el estado anterior c_{t-1} se debe "mantener" (valores cercanos a 1) y que parte se debe "olvidar" (valores cercanos a 0), a partir de los datos de entrada.

Para agregar información a partir de los datos de entrada al nuevo estado de la celda, se genera un proceso en dos partes. La compuerta i_t genera un vector indicando que valores de c_{t-1} deben ser actualizados. La compuerta $\widetilde{c}_t$ genera valores candidatos para la integración en la información del nuevo estado de la celda. La combinación de ambas compuertas es adicionada a la información filtrada a través f_t de la celda anterior generando c_t.

Finalmente, para la generación de un nuevo estado oculto h_t, se utiliza el estado de la celda c_t, al cual se le aplica la función tangente hiperbólica para forzar valores en el rango $[-1, 1]$ y luego es filtrada a partir de los valores generados en la compuerta o_t. De esta manera, la información almacenada en el estado h_t dependerá de la estimación de parámetros de la red y los inputs entregados.

Gated Recurrent Units (GRU)

La idea de las celdas GRU, es similar a las LSTM, sin embargo, como se puede observar, sólo poseen dos puertas: "reset gate" r_t y "update gate" z_t. La primera, determina como combinar el nuevo input x_t con la memoria previa h_{t-1}. La segunda determina cuanto de la memoria mantener. Otra diferencia es que las celdas GRU no poseen la memoria c_t.

Las celdas GRU son capaces de guardar y filtrar la información usando sus puertas de "update" y "reset", lo que elimina el problema de desvanecimiento de gradiente. Al poseen menos parámetros que la celda LSTM, es más rápida de entrenar, además de mejora la generalización en escenario con pocos datos de entrenamiento. Sin embargo, no existe consenso respecto a cual celda es mejor, y esto dependerá del escenario.

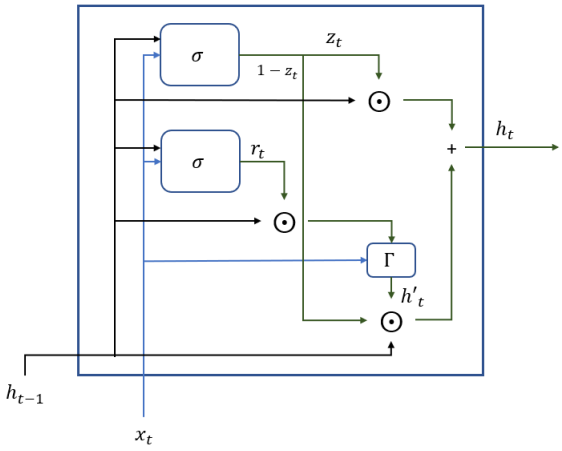


Figura 4.3: Celda en red Gated Recurrent Units (GRU)

Stacked RNN

En los modelos LSTM-GRU originales se componen de una capa oculta de celdas seguidas por una capa de resultados. En la versión apilada ("stacked RNN") se presentan múltiples capas con celdas LSTM-GRU, haciendo el modelo más profundo y preciso. Cada capa se provee una secuencia de resultados en vez de un resultado simple para la siguiente capa, es decir, un output por input en cada paso temporal, más que un output para todos los pasos temporales. En la capa final existe la conexión que permite encontrar los resultados finales. Su arquitectura se presenta en la figura 4.4.

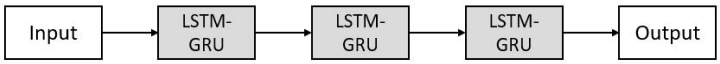


Figura 4.4: Arquitectura modelo RNN de dos capas.

4.3.3. Mixture density networks (MDN)

Corresponden a la combinación de redes neuronales convencionales con un modelo de densidad de mezcla. En [15] se explica la idea utilizando una red neuronal Feed-forward, utilizada frecuentemente para los problemas de clasificación.

La minimización de suma de errores cuadráticos o entropía cruzada permite a las redes neuronales aproximar el valor promedio de las variables objetivo. En los problemas de clasificación, las variables objetivos pertenecen a una posible clase dentro de un conjunto finito, por lo cual, cada resultado obtenido en la capa de outputs de la red puede ser interpretado como las probabilidades posteriores de pertenecer a una clase. Sin embargo, para el caso en el que se quiere predecir una variable continua, el promedio condicional representa una descripción limitada de las variables objetivo.

En la minimización de errores cuadráticos se puede representar la distribución condicional de la variable objetivo t respecto a x con una función Gaussiana, donde la media corresponde

al promedio condicional a los inputs de las variables objetivo, y la varianza al error residual.

De lo anterior, nace la idea de reemplazar la distribución Gaussiana por un modelo de mezcla, a través de una combinación lineal de Kernels:

$$p(t|x) = \sum_{\mathrm{i}=1}^{m} \pi_{\mathrm{i}}(x)\phi_{\mathrm{i}}(t|x)$$

donde m corresponde al número de componentes en la mezcla. π_{i} son los coeficientes de mezcla, interpretados como las probabilidades a priori de que las variables objetivo t sean generadas por la i-ésima componente de la mezcla, condicionado en x. Las funciones $\phi_{\mathrm{i}}(t|x)$ corresponden a la probabilidad de que la variable objetivo adquiera los valores t condicional en x para el i-ésimo kernel Gaussiano:

$$\phi_{\mathrm{i}}(t|x) = \frac{1}{(2\pi)^{c/2}\sigma_{\mathrm{i}}(x)^{c}} \exp -\frac{\|t - \mu_{\mathrm{i}}(x)\|}{2\sigma_{\mathrm{i}}(x)^{2}}$$

Donde las varianzas σ_{i} representan los parámetros de escala y las medias representan los parámetros de locación.

Sin entrar en mayor detalle las formulaciones, en la imagen 4.5 se muestra la estructura de este tipo de redes. En particular, se observa que la red Feed-forward en vez de generar resultados respecto a las clases para la clasificación, genera los parámetros para el modelo de mezcla (π_{i}, μ_{i} y σ_{i}), el cual entrega la representación de la función de probabilidad de las variables objetivo, condicional a los inputs.

Escogiendo un modelo de mezcla con suficientes número de funciones Kernel y una red neuronal con suficiente capas ocultas, MDN puede aproximar muy cerca los valores de la probabilidad condicional $p(t|x)$.

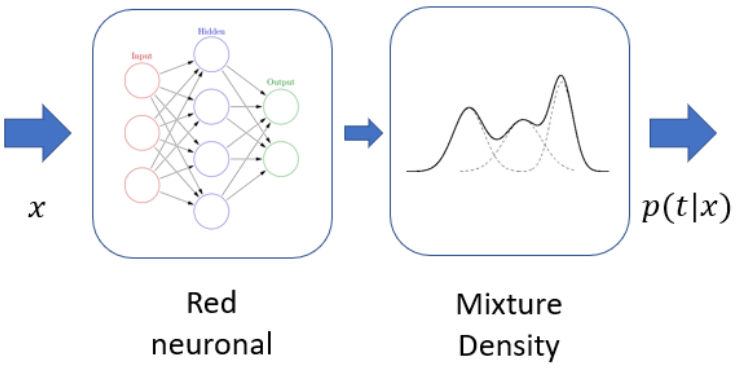


Figura 4.5: Esquematización Mixture Density Network.

Al calcular el promedio de las variables objetivos t condicional en los valores de x, se obtiene un resultado equivalente a los entregados por una red neuronal entrenada por minimización de cuadrados o entropía cruzada, por lo que estos últimos corresponde a un caso particular de los modelos MDN.

4.3.4. LSTM-MDN

Según lo expuesto en 4.3.3 la idea principal de MDN es utilizar los outputs de la red como parámetros para caracterizar las distribuciones gaussianas a ser mezcladas.

Estos modelos también pueden ser usados con redes neuronales recurrentes (ver figura 4.6). En este caso las distribuciones generadas no dependerán solamente de los inputs, sino también de la historia de los inputs anteriores. Es así como en cada paso temporal se genera una distribución de probabilidad consistente con los resultados adyacentes temporalmente.

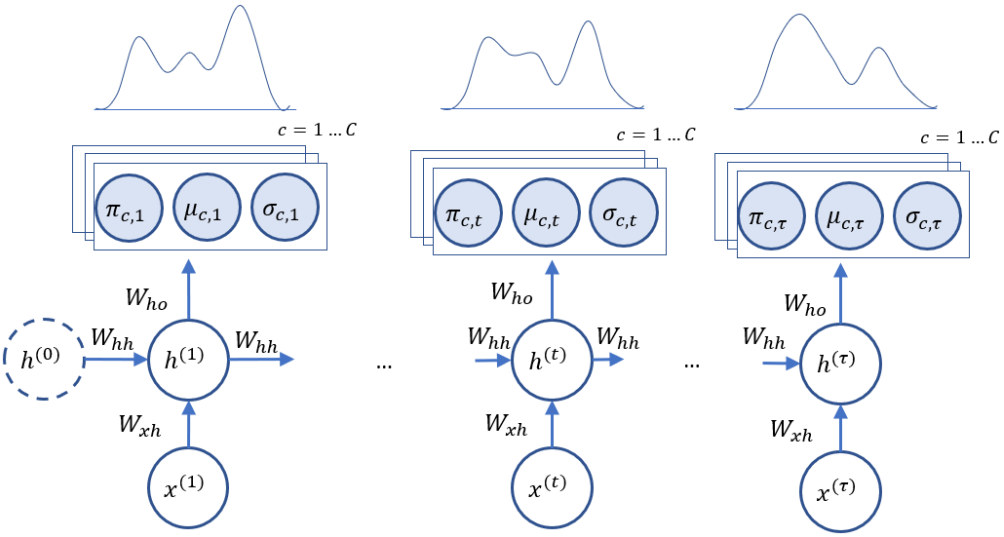


Figura 4.6: Esquematización redes LSTM-MDN.

Tanto RNN como LSTM combinados con MDN generan una distribución de probabilidad como resultado para cada etapa temporal t. Estos resultados pueden ser normalizados para generar los denominados mapas de saliencia, los cuales pueden ser agregados para obtener un mapa de saliencia estático, o ser analizados temporalmente.

Las distribuciones gaussianas pueden ser bivariadas, usadas, por ejemplo, para la predicción conjunta de coordenadas en dos dimensiones (x, y). En este caso la red debe estimar los parámetros de correlación ρ_{i}, además de μ_{i}, σ_{i} y π_{i} expuestos anteriormente, donde μ_{i}, σ_{i} deben ser calculados en dos coordenadas, es decir, $\mu_{\mathrm{i}} = (\mu_{x,\mathrm{i}}, \mu_{y,\mathrm{i}})$, $\sigma_{\mathrm{i}} = (\sigma_{x,\mathrm{i}}, \sigma_{y,\mathrm{i}})$.

4.3.5. Arquitectura Encoder-Decoder

La arquitectura encoder-decoder detallada en los papers [96, 29], es un método comúnmente utilizado para los problemas de traducción y la predicción de secuencias en los denominados problemas "Seq2Seq". Su principal ventaja es permitir largos distintos para las secuencias de entrada y salida en el modelo.

En general consta de una red de codificación de los datos entregados a un vector de contexto, denominada "Encoder", una red de decodificación donde se generan resultados objetivo, denominada "Decoder" y en ocasiones es utilizada una tercera red denominada "Attention",

que permite una mejor decodificación del vector de contexto en la secuencia entregada por el modelo.

El encoder, presenta la habilidad de las redes de aprender representaciones eficientes de los datos de entrada. Aquí se utiliza una red que mapea los datos crudos de entrada a representaciones de características. El decoder, aprende a tomar esta representación de características, procesarlo y tomar decisiones, produciendo los datos de salida. Teóricamente, la red de encoder puede trabajar en crear un vector de características de manera independiente de la red de decoder. Sin embargo, el entrenamiento de la red se realiza de manera "end to end", y no por elementos separados.

Existen diferentes tipos de redes encoder-decoder, por ejemplo, en el caso del procesamiento de imágenes, se puede usar una red convolucional para codificar una imagen en un vector de características, que puede ser decodificada por ejemplo, por una red RNN para la generación de un título para la imagen.

Para el presente trabajo, se utilizará el enfoque de redes encoder-decoder RNN. En la imagen 4.7 se muestra un ejemplo de estas redes en la resolución del problema de generar una respuesta a partir del decoder para un email entrante en el encoder, donde los datos de entrada y salida corresponden a secuencias de palabras.

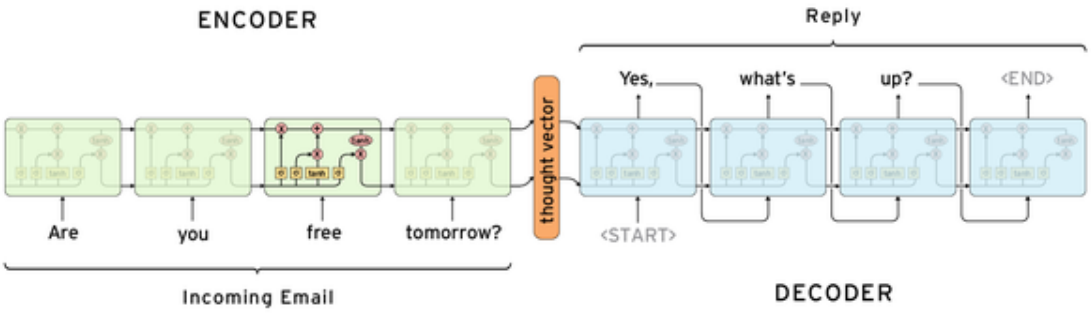


Figura 4.7: Arquitectura encoder-decoder redes RNN

Los modelos de atención fueron expuestos en el paper [8] extendiendo los modelos encoder-decoder. Se propone como una solución al problema de la arquitectura encoder-decoder de poder resumir la información de los datos de entrada en un vector de contexto de largo fijo. Al mismo tiempo, permite identificar cual de los datos de entrada (que paso temporal) es más importante para cada paso temporal de la secuencia de salida del modelo, lo que es útil por ejemplo, en los problemas de traducción de frases.

4.3.6. Selección de variables en redes neuronales recurrentes.

Principal Feature Analysis.

En el estudio [71] se propone un método de reducción de dimensionalidad de un conjunto de variables a partir de la selección de un subconjunto de ellas que contenga la información esencial, usando el mismo criterio que el método de PCA. A diferencia de este último, en este

método, se genera un subconjunto de variables en vez de una transformación de la totalidad de ellas a un espacio de menor dimensión.

Sea X un vector de variables de dimensión n con media cero. Sea Σ la matriz de covarianza de X y A la matriz cuyas columnas son vectores propios ortonormales de $\Sigma = A\Lambda A^\top$

$$\text{Donde} \quad \Lambda = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_2 \end{bmatrix}, \quad \text{con } \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \text{ valores propios de } \Sigma$$

Sea A_q las primeras q columnas de A y $V_1, V_2, \ldots, V_n \in \mathbb{R}^q$ las filas de A_q, las cuales representan la proyección de la i-ésima variable del vector X en un espacio de dimensión menor, es decir, los q elementos del vector V_i corresponden a los pesos de la i-ésima variable en cada eje del subespacio.

Las variables independientes presentan una máxima separación entre los vectores de peso, mientras que variables completamente correlacionadas presentan vectores de pesos iguales, independientemente de un cambio de signo.

Este método utiliza las estructuras de los vectores V_i para encontrar las variables más correlacionadas y en seguida escoger una variable de cada subconjunto. Los pasos del algoritmo se detallas a continuación:

- Calcular la covarianza de la matriz muestreada o usar la real si está disponible. En los casos en que las variables tienen varianzas diferentes unas de otras, el uso de la matriz de covarianza puede causar que PCA asigne mayores pesos a las variables con mayor varianza. Por esta razón en esos casos se prefiere usar la matriz de correlación dada por:
$$\rho_{\text{i}j} = \frac{\mathbb{E}\lfloor x_\text{i} x_j \rfloor}{\mathbb{E}[x_\text{i}^2] \cdot \mathbb{E}[x_j^2]}$$
- Calcular las componentes principales y valores propios de la matriz de covarianza o correlación.
- Escoger un subespacio de dimensión q y construir la matriz A_q de A. Se puede escoger según la cantidad de variabilidad que se busca mantener de los datos.
$$\text{Variabilidad retenida} = \frac{\sum_{\text{i}=1}^{q} \lambda_\text{i}}{\sum_{\text{i}=1}^{n} \lambda_\text{i}} \cdot 100\,\%$$
- Clusterizar los vectores $|V_1|, \ldots, |V_n|$ en $a \geq q$ clusters usando K-Means con distancia euclideana. Se escoge p mayor a q ya que para obtener la misma cantidad de varianza retenida se requieren entre 1-5 variables más
- Para cada cluster escoger el vector V_i que esté más cerca de la media del cluster. Escoger x_i como variable principal. Repetir para obtener un subconjunto de p variables.

Métodos de envoltura

Tal como se expone en [69] existen principalmente dos enfoques en la evaluación de la calidad de los subconjuntos de variables: y métodos de filtro y los métodos de envoltura

("filter" y "wrapper" en inglés). El primero evalúa cada variable de manera individual, entregando un puntaje para cada variable de acuerdo a características presentes en el conjunto de entrenamiento.

Los métodos de envoltura, por lo general son muy caros computacionalmente. lo que hacen es usar un subconjunto de variables para entrenar un modelo y a partir de ello decidir si agregar o eliminar variables, basándose esencialmente en el problema de búsqueda de las variables. En la figura 4.8 se muestra el proceso desarrollado en estos métodos.

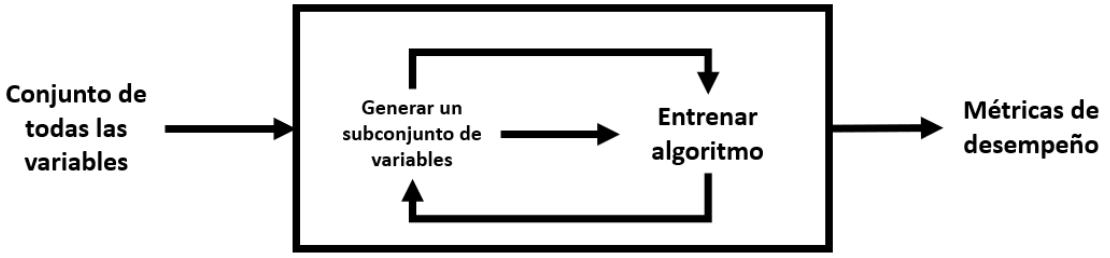


Figura 4.8: Métodos de envoltura de selección de variables.

Los métodos más utilizados en estas categorías corresponden a "Backward Selection" y "Forward Selection" definidos a continuación para el caso en que se validan los resultados a partir de k-fold.

Para ello se realiza un proceso de k iteraciones. En cada una de ellas se definen los conjuntos de entrenamiento y test. El conjunto de entrenamiento es subdividido en los conjunto de entrenamiento interior (EI) y test interior (TI).

El conjunto EI corresponde al 70 % de los datos presentes en el conjunto de entrenamiento y son utilizados para entrenar modelos con diferentes subconjuntos de variables. Cada subconjunto es medido a partir del conjunto TI para la selección interna de los mejores subconjuntos. Finalmente los resultados entregados serán calculados a partir del conjunto de test, el cual no ha sido utilizado en el proceso de selección de variables.

Forward Selection: Es un método iterativo en el que se empieza sin considerar ninguna variable en el modelo. En cada iteración se agrega la variable que entregue una mayor mejora en los resultados del modelo. Sea el conjunto de todas las variables $S_1 = \{f_1, \ldots, f_n\}$, realizar los siguientes pasos.

- Comenzar con la variable f_s que mejor que individualmente tenga mejor desempeño en el conjunto TI y agregar la variable al conjunto de las mejores variables B_1. Definir $S_2 = S_1 \setminus \{f_s\}$.
- para $m = 2, \ldots, n-1$:
  - Para todo $f_{\mathrm{i}} \in B_m$ calcular el desempeño de $B_{m-1} \cup \{f_{\mathrm{i}}\}$ en el conjunto TI y ordenar las variables según su desempeño.
  - Agregar la variable de mejor desempeño f_{i} al conjunto de las mejores variables $B_m = B_{m-1} \cup \{f_{\mathrm{i}}\}$ y el conjunto $S_{m+1} = S_m \setminus \{f_{\mathrm{i}}\}$

Finalmente se escoge el subconjunto de variables B_m con mejor desempeño en el conjunto TI y calcular el desempeño en el conjunto de Test.

Backward Selection: Es un método iterativo en el que se empieza considerando todas las variables en el modelo. En cada iteración se elimina la variable que entregue una mayor mejora en los resultados del modelo. En este caso no se requiere de los conjuntos S_m y se define $B_n = \{f_1, \ldots, f_n\}$.

- Comenzar con todas las variables f_i en el conjunto B_n.
- para $m = n-1, \ldots, 1$:
  - Para todo $f_\text{i} \in B_{m+1}$ calcular el desempeño de $B_{m+1} \cup \{f_\text{i}\}$ en el conjunto TI y ordenar las variables según su desempeño.
  - Eliminar la variable f_i correspondiente al conjunto de mejor desempeño $B_{m+1} \setminus \{f_\text{i}\}$ y actualizar el conjunto $B_m = B_{m+1} \setminus \{f_\text{i}\}$

Finalmente se escoge el subconjunto de variables B_m con mejor desempeño en el conjunto TI y calcular el desempeño en el conjunto de Test.

Capítulo 5

Problema Abordado

En este capítulo se explican los antecedentes del problema y una descripción más detallada acerca del desarrollo del experimento, diseño de la página web, forma de interacción entre el usuario y el sitio web, necesario para comprender la fuente de datos a utilizar. Posteriormente se explica el pre-procesamiento de datos necesario para la congruencia con la descripción entregada para el sitio web. Al final del capítulo se plantean modelos para cada uno de los escenarios descritos en la sección 2.2.

5.1. Antecedentes

El trabajo se basa en el estudio llevado a cabo en la tesis presentada por Cristian Retamal [88] "Estudio del comportamiento de la carga cognitiva de usuarios que navegan en un sitio web", bajo el contexto del proyecto Fondecyt "A Cognitive Resource-Aware Mobile Service Framework to Support Human-Computer-Interactions in Ubiquitous Computing Environments", liderado por el Profesor Ángel Jiménez, en la cual se demuestra la hipótesis de que es posible medir la carga cognitiva para actividades de navegación web frente a un computador mediante señales psico-fisiológicas y que, además, se produce una baja en la carga mental del usuario en momentos de transición entre el análisis de un elemento web y otro.

En dicha tesis se lleva a cabo la experimentación de la interacción entre usuarios y un sitio web específico. El diseño experimental considera la medición de 61 voluntarios utilizando 6 sensores psico-fisiológicos: Eye tracker, sensor de respuesta electrodermal (GSR), Sensor de Temperatura de la piel (ST), Fotopletismógrafo (PPG), electrocardiógrafo (ECG) y electroencefalógrafo (EEG). Cada participante navega libremente por una página web que se presenta en 3 versiones manteniendo el diseño y variando su contenido.

Los datos recopilados al finalizar este trabajo, son utilizados para abordar la generación de un modelo de predicción de fijaciones oculares de los usuarios durante la navegación en el sitio web, con medición de señales en tiempo real. Se espera obtener resultados satisfactorios para el sitio web implementado, bajo condiciones controladas de luminosidad.

5.2. Descripción

En este problema se cuenta con datos de las señales de diferentes sensores psico-fisiológicos registrados a usuarios durante la navegación en un sitio web específico, en un ambiente controlado (luminosidad y movimiento de los usuarios).

El sitio web, denominado "turincon.com" cuenta con un diseño inalterable entre mediciones, no así su contenido, el cual cambia la información de las noticias y anuncios publicitarios presentes en la página, para cada usuario y cada interacción, de manera aleatoria, a pesar de que la ubicación, forma y tamaño de cada componente de la página se mantiene.

En la figura 5.5 se presenta el sitio web de forma estática y completa, aunque en la realidad, las dimensiones de la pantalla usada para la experimentación no permiten una visualización completa, requiriendo el uso de la barra de desplazamiento por parte del usuario. Al mismo tiempo, los usuarios puede interactuar con diferentes componentes presentes en la página, lo que le da un carácter mas dinámico al problema. La imagen resalta con diferentes colores las componentes presentes en el sitio web, las cuales se definen a continuación:

Logo: Corresponde a la ilustración que simboliza el nombre ficticio del sitio web, entregando un diseño adecuado a los estándares de páginas web esperables por el usuario. Además, considera un botón de selección de idioma, el cual se representa con el icono de la bandera chilena. Al ser oprimido genera una lista desplegable en la cual el usuario puede seleccionar el país en el cual se encuentra (véase figura 5.1). En caso de cambiar de país, la página realizará una recarga, sin embargo, ni el contenido ni el idioma sufrirán cambios.

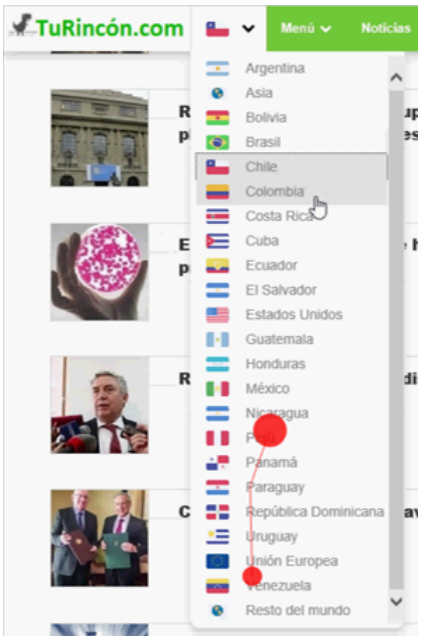


Figura 5.1: Menú desplegable al oprimir botón de selección de países.

BarMenu: Alude a las barras de menú presente en los diseños web tradicionales. Presenta los botones "Menú", "Noticias", "Tecnología", "Deportes", "Misceláneos" y "Foros", además de iconos relacionados a "Redes sociales", "Compartir contenido", "Registro de usuario" y

"Búsquedas". Presionar cualquiera de estos botones no afecta en la página web, a excepción de los botones "Menú" y "Registro de usuario". Esta barra se sitúa siempre en la parte superior de la página, independiente del scroll realizado por el usuario.

En el caso del botón "Menú", al ser presionado se despliega una lista de secciones en la página, que al ser presionadas no afectaran a la página web. La lista desplegable se muestra en la figura 5.2.

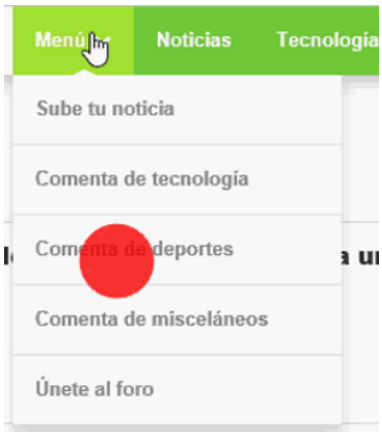


Figura 5.2: Menú desplegable al oprimir botón "Menú"

En el caso del botón presentado con el icono de "Registro de usuario", al ser presionado generará una ventana sobre la página, oscureciéndola y permitiendo un mejor enfoque en el recuadro generado. Dentro de esta ventana, existen dos botones, "Ingresar" y "Regístrate", que al ser presionados cambian las dimensiones de dicha ventana. Los demás botones presentados no generan efecto en la página y para cerrarla se requiere de realizar un clic en cualquier área de la página fuera de la ventana. Las figuras 5.3 y 5.4 muestran el contenido al presionar el botón estando en la sección de Ingreso y Registro respectivamente.

Title: Corresponde al título de la sección principal de la página. Hace referencia al usuario que se encuentra en la sección de "Noticias" dentro de la página web.

imgNews: Corresponde a las imágenes presentadas en cada una de las siete noticias presentes en la página web. Cada una acompaña el texto relacionado a la noticia y todas poseen igual tamaño. Los clic realizados en estas componentes no surgen efecto alguno.

titleNews: Corresponde a los títulos de las noticias presentes en el sitio. Estas pueden diferir en la cantidad de palabras, sin embargo se presentan cuadros de igual tamaño. Los clic realizados en estas componentes no surgen efecto alguno.

Banner: Corresponde a los cada uno de los cuatros anuncios publicitarios presentados en la página. Todos poseen igual tamaño. Al presionar sobre alguno de ellos, la pagina realizará una recarga, sin alterar el contenido o diseño de ella.

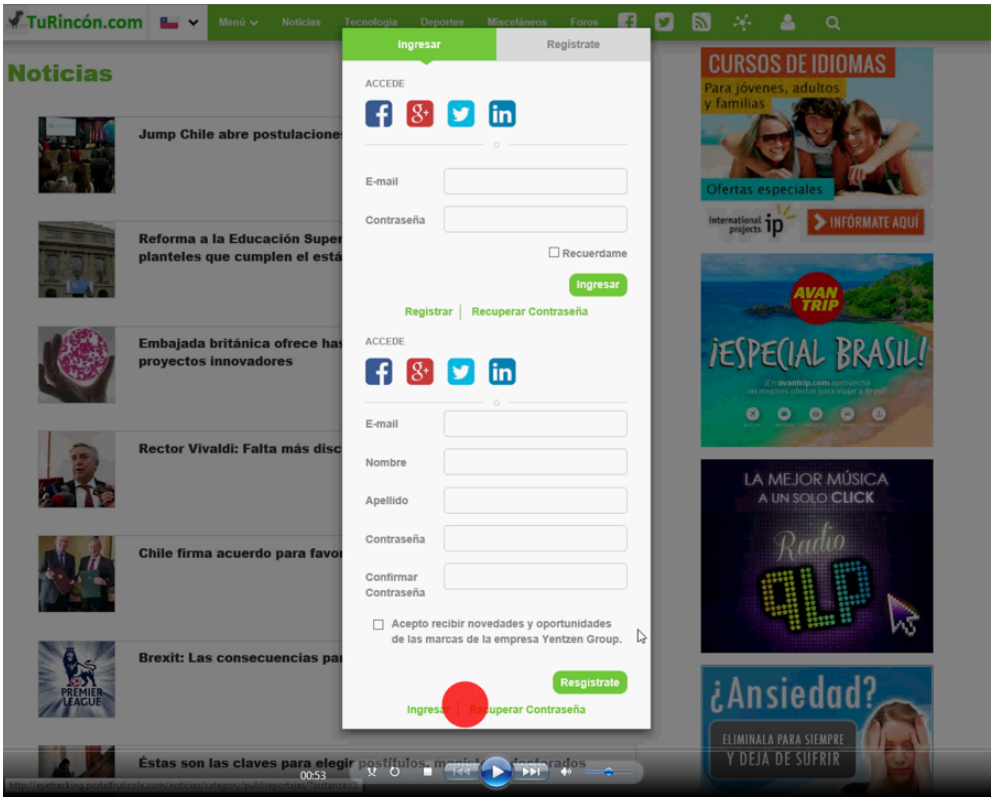


Figura 5.3: Ventana de Ingreso.

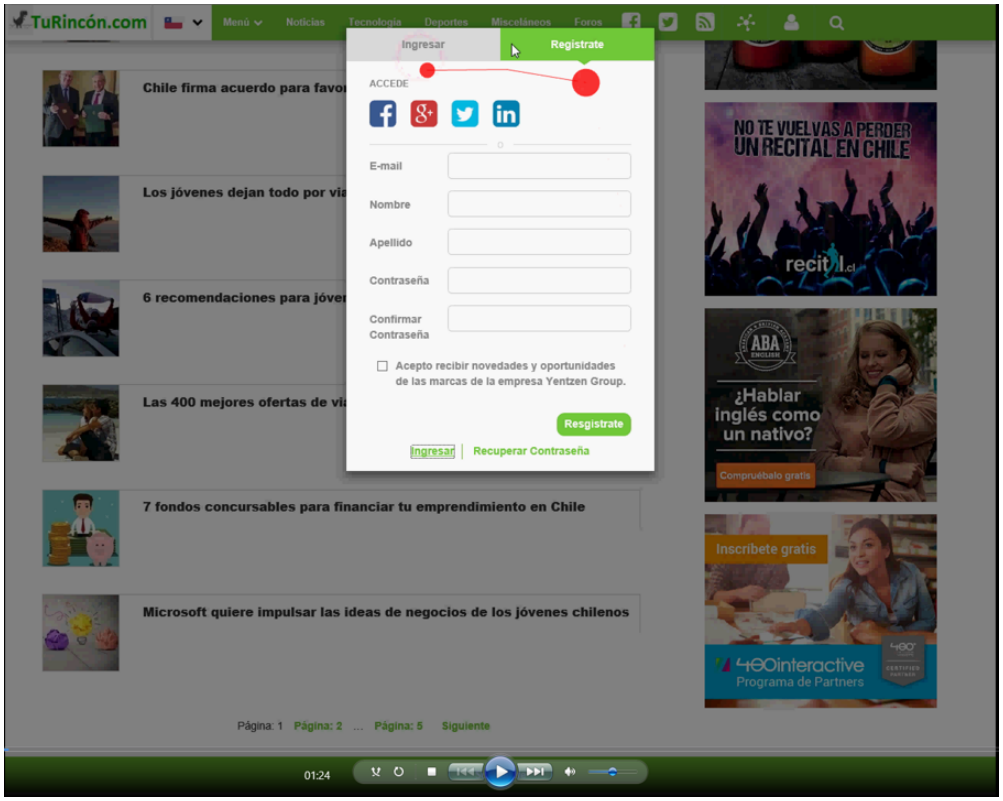


Figura 5.4: Ventana de Registro.

numberPages: corresponde a la barra de navegación, donde cada número representa qué noticias serán presentadas en la página web, según algún orden. Los clic realizados en esta componentes no surgen efecto.

barSubscribite: Corresponde a una franja inferior presentada en la página web para registro de usuarios. Oprimir esta área no afecta a la página.

Figura 5.5: Definición componentes presentes en sitio web

El experimento original consiste en la exposición de 61 usuarios a la página web. Para cada usuario, la exposición consta de cuatro etapas, tres escenarios de interacción con el sitio web, diferenciando el tiempo de carga al iniciar, y finalmente un cuestionario acerca de la experiencia. Al iniciar la página web, se presentaba una pantalla en blanco, en la cual sólo se muestra el primer banner, con contenido aleatorio (véase figura 5.6). Estas pantallas de inicio se presentaban por 0 segundos, 500 milisegundos y 5 segundos, antes de cargar completamente la página, en cada una de las tres exposiciones, respectivamente.

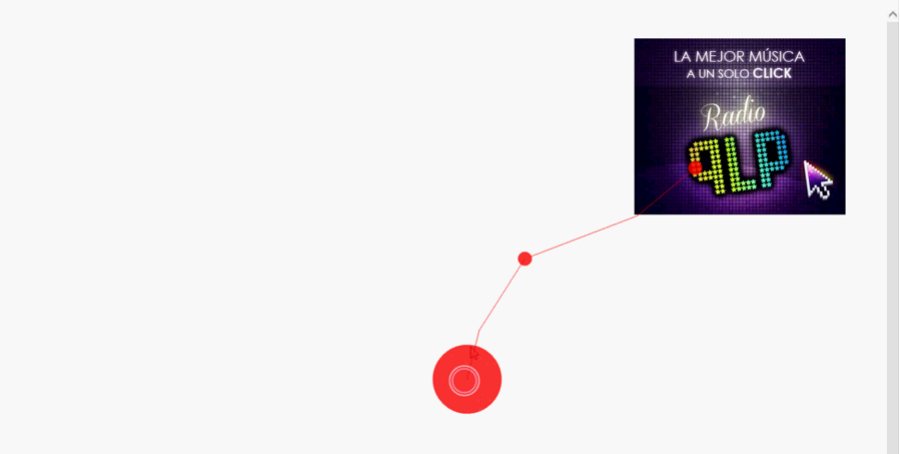


Figura 5.6: Pantalla de inicio de la experiencia.

En este trabajo se considera que cada una de las tres experiencias realizadas por un usuario son independientes. Esta suposición se basa en el hecho de que en cada instancia el contenido presentado es diferente, por lo que los usuarios pueden navegar libremente sin considerar que el sitio es completamente similar al presentado en la instancia anterior. Ellos no poseen la información de que el diseño de la página no se ve alterado, por lo que en las nuevas instancias se observan análisis similares a la primera interacción (inspección total de la página).

Utilizando el software TobiiStudio para el tratamiento de señales recolectadas por el "Eye-Tracker", se considera que cada señal comienza cuando el usuario hace clic en el cuadro de diálogo que le indica la siguiente exposición al sitio, y finaliza cuando el usuario decide pasar a la siguiente instancia. Es así como las señales originales son divididas temporalmente en tres nuevas señales para cada usuario.

Utilizando los datos temporales de la toma de estos datos, el resto de las señales recolectadas, también son divididas en tres señales para cada usuario. Es así como las señales de 61 participantes se convierten en 183 conjuntos de señales independientes de interacción con el sitio web, o como se tratará de aquí en adelante, 183 registros de usuarios.

5.2.1. Análisis exploratorio y preprocesamiento de datos.

A continuación se explica el preprocesamiento realizado a los datos. De aquí en adelante, se consideran los datos obtenidos del trabajo original, pero el tratamiento llevado a cabo es independiente, relacionado únicamente a esta investigación.

De las componentes descritas anteriormente, se observa que algunas de ellas presentan comportamientos dinámicos, es decir, alteran el diseño de la página web por ciertos periodos de tiempo, desplegando barras de menú para la selección de opciones, abriendo recuadros por sobre otras componentes o, en el caso de la barra de menú, se mueven en conjunto a la barra de desplazamiento lateral, permaneciendo en todo momento en la posición superior de la pantalla a la vista del usuario, por sobre otras componentes.

De esta manera se establecen dos tipos de componentes, las estáticas y las dinámicas.

Las dinámicas corresponden a la barra de menú, las listas desplegables (selección de idioma y menú) y el recuadro de registro. La principal dificultad que presenta el manejo de información respecto a estas componentes es que su aparición dependen de la actividad del usuario.

Se definen las componentes estáticas como posiciones fijas en la página web (coordenada de inicio, ancho y alto de la componente en píxeles) y, por otro lado, las componentes dinámicas son definidas manualmente, mediante inspección de los registros de cada una de las interacciones realizadas por los usuarios. En particular, se revisa el video de cada una de estas interacciones y se delimitan las áreas de las componentes dinámicas, cuadro a cuadro.

Una vez definidas las componentes estáticas y dinámicas, se limpia el registro seguimiento ocular, evitando que una fijación correspondiente a una componente dinámica sea registrada para alguna componente estática que la traslape. Se observa que en ocasiones los usuarios realizan interacciones que interrumpen la visual de la página. Por ejemplo, al realizar un clic derecho sobre esta, las fijaciones realizadas en el menú desplegado no se consideran como fijaciones realizadas a la pagina web. Estas condiciones encontradas son definidas manualmente, cuadro a cuadro, al igual que en el caso de las componentes dinámicas.

Para el resto de las señales psico-fisiológicas solo se realiza la partición en cada una de las instancias, utilizando las marcas de tiempo presente en los datos de seguimiento ocular (tiempo de inicio y tiempo de finalización).

Finalmente se obtiene una estructura de 61 conjuntos de datos, correspondientes a cada participante, con su información correspondiente a "nombre", "edad", "género", "trabajo", "fecha de exposición". Cada entrada posee tres columnas (señales 1, 2 y 3), correspondientes a las instancias con tiempo de carga inicial de página de 0 segundos, 500 milisegundos y 5 segundos respectivamente. Dentro de las ventanas de tiempo de cada una de las señales se encuentran los datos registrados de eyeTracker, GSR, ST, ECG, PPG y EEG.

Una vez realizado el preprocesamiento de datos, se analizan para comprender de mejor manera el problema abordado. En la tabla 5.1 se muestra la cantidad de registros de usuarios que presentan cada una de las señales psico-fisiológicas, de los 183 totales. En la columna final se observa el número de registros que presentan todas las señales correctamente registradas.

Tabla 5.1: Cantidad de registros por sensor

ECG	**EEG**	**GSR**	**ST**	**PPG**	**EyeTracker**	**Todos**
177	174	183	183	183	183	168

Para algunos usuarios el registro de ciertas señales no existe por lo cual dichos casos son eliminados del modelo desarrollado. En particular, se tiene pérdida de los datos de ECG para dos usuarios y de EEG para tres usuarios, en sus tres interacciones, por lo que se pierden 15 interacciones.

Como se explica más adelante, las señales fisio-psicológicas se ajustan por la línea base del usuario correspondiente. De los usuarios a considerar, no se posee los registros de línea base de 5 usuarios, eliminando un total de 15 interacciones.

Finalmente queda un total de 153 registros para la elaboración del modelo. En la tabla 5.2 se observan algunas estadísticas descriptivas de los datos relacionados con el seguimiento ocular de los usuarios a considerar.

Número de registros	Total	153
Tiempo de registro (seg)	Mínimo	16,53
	Promedio (desviación estándar)	78,16 (51,98)
	Máximo	399
Sacadas (N° Total)	Mínimo	62
	Promedio (desviación estándar)	262 (165)
	Máximo	1129
Fijaciones por registro	Mínimo	62
	Promedio (desviación estándar)	262 (164,6)
	Máximo	1130
Duración fijación por registro	Mínimo	8 (miliseg)
	Promedio (promedios)	0,22 (seg.)
	Máximo	3,273 (seg.)

Tabla 5.2: Estadísticas descriptivas datos EyeTracker

Como se observa, a través de los diferentes registros se tienen tiempos de navegación distintos y consecuentemente el número de sacadas y fijaciones realizadas por cada usuario difiere. Por otro lado, en la tabla se observa que las fijaciones de menor duración corresponden a 8 milisegundos, las cuales serán eliminadas ya que no representan una atención real del usuario.

5.3. Modelo Planteado.

En este capítulo se presenta el modelo realizado para abordar la predicción de fijaciones en cada escenario y la predicción de indicadores mentales, expuestos en la sección 2.2 y su motivación para plantearlos de esa forma. En el capítulo 6 se presenta la ejecución de los modelos.

5.3.1. Modelo predicción de fijaciones

Primer escenario: predicción a nivel semántico de AOI's

El primer escenario es una simplificación del problema en el sentido que no busca la predicción completa de las secuencias de fijaciones realizadas por el usuario (scanpaths), sino que las secuencias de áreas de interés definidas en la página web. Posteriormente el segundo escenario busca abordar el problema en mayor completitud (prediciendo de fijación en fijación). Se utiliza este orden ya que así se comprende si el segundo escenario permite mejores resultados al considerar información más detallada y si el primer escenario permite una primera mirada acerca de la posible generalización de los resultados.

Dadas las componentes explicadas en 5.2, la primera variable de decisión es la definición de las áreas de interés (AOI) en la página web, pudiendo ser secciones delimitadas por tamaño, o agrupaciones de componentes, entre otras alternativas. Supongamos un caso en el que se definen un número n de AOI's.

El siguiente factor a considerar corresponde a el tiempo que el usuario navegará por el sitio web. Como se presentó en la tabla 5.2 para los registros de entrenamiento con lo que se cuenta, estos varían entre 16 segundos y 6 minutos 39 segundos. Además, los tiempos entre transiciones realizadas entre diferentes AOI's no necesariamente son constantes entre usuarios o para el mismo usuario durante toda su navegación.

El número de transiciones entre Áreas de interés dependerá del usuario, y cada 5 segundos se tiene un promedio de 2.6 transiciones realizadas (desviación estándar de 1.6). Esto unido a que el tiempo promedio de navegación es de 77.8 segundos, se tiene para un usuario promedio alrededor de 200 transiciones por navegación.

Si se considera que en cada navegación un usuario realiza 200 transiciones, y se tienen n AOI's definidas, el número de posibles caminos visuales es de $n(n-1)^{199}$. Este es un problema debido a la que sólo se posee 168 caminos realizados (registros de usuarios), por lo que el enfoque de predicción directa no es factible de realizar.

Dado lo anterior, se propone un enfoque de predicción por ventanas temporales, es decir, dado un tamaño τ para las ventanas, la señal original de tamaño T será dividida en $\lceil T/\tau \rceil$ secciones (véase figura 5.7). De esta manera se busca predecir los scanpaths realizados en cada una de estas secciones, simplificando el camino visual a predecir, ya que ahora el número de transiciones es considerablemente menor. El valor de τ es una variable de decisión.

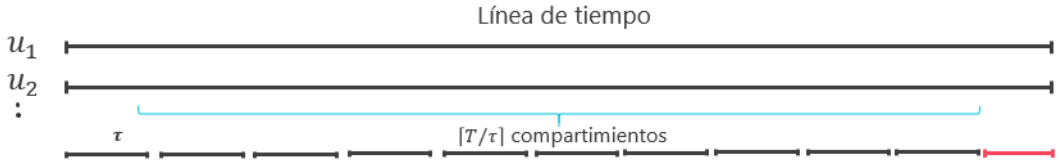


Figura 5.7: Diagrama de predicción por ventanas.

Es así como la predicción de caminos visuales se transforma en una predicción en tiempo real. Es decir, dado un punto en el tiempo de navegación, se busca la predicción de camino visual para la siguiente ventana. Esto permite que las señales registradas en las ventanas anteriores sean utilizadas para la predicción (incluido el orden de las secuencias visitadas anteriormente). Es importante considerar que durante la navegación del usuario, se recibirán las señales continuamente.

En cada ventana temporal se obtiene información del usuario sobre su navegación (áreas visitadas, orden de las visitas, clics, etc), en conjunto a señales del eyeTracker, y el resto de sus señales psico-fisiológicas. Se puede pensar en que estas últimas entregan un conjunto de información que permite caracterizar sus procesos corporales, como por ejemplo, sus movimientos oculares. Por otro lado, los datos de navegación pueden ser considerados de forma externa a la biología del individuo, representando un historial que permita estimar o conocer los intereses del usuario y las intenciones que posee en cuanto a su navegación futura.

Aunque ambos grupos de información se capturan de manera conjunta, la idea explicada permite un tratamiento de forma separada, de esta manera que se puede extraer conocimiento desde ambas fuentes, y así, generar un modelo que las integre en la predicción de las secuencias visuales.

Dado lo anterior, se establecen tres etapas en la predicción. Las primeras dos corresponden al tratamiento de la información desde ambas fuentes y la tercera etapa la integración de ambos resultados al modelo de predicción correspondiente. A continuación se entrega un planteamiento de cada una de ellas.

Etapa 1: Predicción intención de visita AOI's.

La primera hipótesis a considerar es que el comportamiento de los usuarios queda caracterizado por dos componentes que son las decisiones de a qué áreas prestar atención en cada momento y la forma en la cual alcanzarán sus objetivos. Por ejemplo, si un usuario pretende ver la noticia 1 y la noticia 2, puede hacerlo en cualquier orden (1 a 2, o 2 a 1) y con tiempos de visita y transición aleatorios. Pensado en el modelo de ventanas, se puede pensar que al iniciar cada una, el usuario decide qué zonas visitar en los próximos τ segundos y la forma de hacerlo puede variar.

Si τ es suficientemente pequeño, tomar decisiones al inicio de cada ventana puede ser una aproximación a tomar decisiones continuamente. Por otro lado, la intención de qué zonas mirar en los próximos segundos puede ser analizada a partir del conjunto de datos relacionados a la navegación del usuario y EyeTracker.

En la etapa uno del modelo propuesto, se hace alusión a la toma de decisiones de a qué partes de la página el usuario desea prestar atención. En particular, se desarrolla un método que establezca, dada las áreas de interés definidas, un indicador de interés por parte del usuario a visitar cada una de las AOI. Como esta decisión es probabilística puede que el modelo establezca que se prestará atención cuando en la realidad esto no ocurre. Por lo tato, de ahora en adelante nos referiremos a la "intención de visita" del usuario a visualizar un AOI, evitando que el lector entienda que el comportamiento del usuario deba ser acorde a este indicador.

Definición 5.1 (Intención de Visita) *Dado el estímulo visual (página web), las Áreas de Interés* $\{A_j|j = 1...n\}$ *y la indexación de las ventanas* $\{v = 1, 2, ...\}$*, se define la intención de visita,* $Iv(A_j, v) \in \{0, 1\}$ *como una variable binaria que indica si existe o no la decisión por parte del usuario en visitar* A_j *en la ventana de tiempo* v*.*

Se observa que la intención de visita para las n AOI's definidas puede ser expresada como un vector $IV(v) = [Iv(1, v), ..., Iv(n, v)]$. La hipótesis mencionada al inicio de esta sub sección implica que al comienzo de cada ventana de tiempo v, el usuario decide este vector. En realidad, se puede considerar que el proceso de elección de este vector se produce durante todo el tiempo anterior al comienzo de la ventana v y no necesariamente en ese punto, pero para mejor comprensión nos referiremos a la última situación.

Sea X_v un conjunto de variables relacionadas a la decisión de la intensión de visita. Dentro de estas variables se puede considerar el historial de navegación, es decir, el registro de qué

zonas a visitado anteriormente el usuario en la página, cuánto tiempo demora entre transiciones, número de fijaciones, tendencia en los movimientos oculares, etc. No se considerará los datos psico-fisiológicos, ya que estos se usarán en la segunda etapa del modelo.

Se considera que el vector $IV(v)$ no es independiente de X_v y así, se busca una función $\mathcal{F}$ que permita una aproximación $IV(v|X_v) \approx \mathcal{F}(X_v)$. Se observa que la función $\mathcal{F}$ entrega como resultado vectores binarios de dimensión n. Entonces, un enfoque para resolver esta etapa corresponde a los métodos de clasificación binaria multi-etiqueta (véase sección 4.1).

Existen diferentes métodos de clasificación multi-etiqueta, alguno de los cuales entregan directamente resultados binarios $\{0, 1\}$ y otros en los cuales se obtienen valores de confianza $\{C_j|j = 1...n\} \in [0, 1]$, los cuales pueden ser utilizados de esta forma o, dado un threshold δ, entregan un valor 1 si $C_j > \delta$ y 0 en caso contrario.

Etapa 2: Asignación de estado mental del usuario.

Tal como se explicó en la sección 3.2, el uso de señales psico-fisiológicas ha sido ampliamente estudiado en la literatura. En esta etapa del modelo, se pretende utilizarlas para caracterizar un **estado mental** (distinto a los índices mentales que se predicen en el modelo 2.2.2) en el que el usuario se encuentra al momento de realizar una inspección visual de la página web en una ventana de tiempo. Este estado mental puede estar relacionado con la carga cognitiva que presenta la persona, sus emociones, arousal, etc. A continuación se define formalmente.

Definición 5.2 (Estado Mental) *En el marco de este trabajo se define como una categorización del usuario en un determinado tiempo t, denominado Em(t), calibrado a partir de indicadores extraídos de las señales psico-fisiológicas registradas en todo momento anterior a t.*

Bajo el enfoque de predicción de ventanas, se establecerá el indicador $EM(v) = Em(T_o(v))$, donde $T_o(v)$ representa el tiempo de inicio de la ventana v. Este indicador tomará valores dentro de las categorías e $= 1, ..., E$.

Lo que se busca capturar con estas categorizaciones son las capacidades que presenta el usuario para realizar movimientos visuales, o las tendencias que puede mostrar al momento de realizar una inspección del estímulo. Es decir, intenta capturar el cómo lo hará para alcanzar cada uno de los objetivos propuestos.

Para lograr la diferenciación en las propiedades de transición de los usuarios, se seguirá un enfoque similar al presentado en [73], en el cual se categoriza las densidades de probabilidad de las sacadas de los usuarios en diferentes rangos etarios. En este caso, no se conocen atributos de los individuos que permitan hacer una categorización significativa. Por ejemplo, en el estudio citado, se dice que los niños poseen menor probabilidad de realizar sacadas de mayor amplitud que los adultos. Sin embargo, en el presente estudio se busca una categorización dada por el estado mental, el cual es desconocido.

Se considera que dadas las señales psico-fisiológicas hasta el momento $T_o(v)$, denominadas S_v, existe una función $\mathcal{G}$ que genera un estado mental $EM(v) = \mathcal{G}(S_v)$, para cada ventana de

tiempo v. En la segunda etapa del modelo se busca una función $\mathcal{G}$ y las categorías de estado mental para cada ventana de tiempo.

La función $\mathcal{G}$ entrega como resultado valores de una categorización y estas no se conocen a priori, por lo cual se aplicarán métodos de aprendizaje no supervisado, en particular, métodos de generación de clusters (véase sección 4.2).

Etapa 3: Algoritmo de generación de caminos visuales.

Una vez realizada las dos etapas anteriores, se ha logrado extraer desde ambas fuentes de datos señaladas anteriormente las intenciones de a que AOI pretende el usuario prestar atención en la próxima ventana de tiempo, y el estado mental en el que se encuentra, lo que caracteriza su forma de desplazar la mirada (lo que se detallará más adelante).

La generación de caminos visuales es la última etapa del modelo, en la cual se espera encontrar, a partir de los datos calculados en las dos etapas anteriores, predicciones de las secuencias que realizarán los usuarios en cada una de las ventanas de tiempo definidas. Así, se define:

Definición 5.3 (Camino Visual) *En el marco de este trabajo se define como un camino visual, una secuencia de largo variable l, es decir $x = \{x^{(1)}, x^{(2)}, \ldots, x^{(l)}\}$ es una secuencia, en la cual cada $x^{(t)}$ toma un valor dentro del conjunto de áreas de interés definidas, $\{x^{(t)} = A_j |\ j = 1, \ldots, n\}$ o cero en caso de no presentar atención por parte del usuario a algún área de interés.*

El largo de las secuencias (l) será establecido más adelante, dependiendo de la forma que adquieren los caminos visuales dentro de los conjuntos de entrenamiento. Por lo tanto, para esta etapa se requiere utilizar métodos de predicción de secuencias, para los cuales es posible usar modelos de markov oculto o redes neuronales recurrentes, entre otras alternativas.

Para formalizar, se requiere un modelo $\mathcal{M}$, el cual, para cada ventana de tiempo v, genere una secuencia $x = \mathcal{M}(IV(v), EM(v), EEG(v))$ la cual será comparada al camino real presentado en la ventana $CV(v)$, buscando minimizar alguna medida de distancia.

El término $EEG(v)$ corresponde a características extraídas desde la señal de EEG, que se relacionan con los procesos de movimiento ocular del usuario. Estas serán especificadas más adelante, lo importante es que teóricamente no deben relacionarse a la clasificación en estados mentales.

Segundo escenario: predicción píxel a píxel

En el segundo escenario se busca predecir la secuencia de fijaciones realizadas por el usuario de manera directa, es decir, sin considerar áreas de interés. Lo primero que se debe considerar es que la cantidad de combinaciones de puntos (x, y) es tan grande como la resolución de la página web (1280 x 1024 píxeles).

En este escenario también se utiliza un enfoque por ventanas, en las cuales para cada una de ellas se busca predecir un **scanpath** de largo fijo l, el cual se define como:

Definición 5.4 (Scanpath) *En el marco de este trabajo se define como un scanpath, una secuencia de largo fijo l, es decir $x = \{x^{(1)}, x^{(2)}, \ldots, x^{(l)}\}$ es una secuencia, en la cual cada $x^{(t)}$ toma un valor dentro del conjunto de puntos $\{(x, y)|\ x \in [0, 1280],\ y \in [0, 1024]\}$ dentro de la pagina web.*

Se observa que la definición es similar al caso de "Camino visual" entregada en el primer escenario. De igual forma, se requiere un modelo $\mathcal{M}$, el cual, para cada ventana v de tiempo, genere una secuencia $x = \mathcal{M}(X(-v))$ que será comparada a los datos reales buscando minimizar alguna medida de distancia, donde $X(-v)$ corresponde a variables calculadas a partir de los registros de señales anteriores al inicio de la ventana de tiempo v.

Por lo tanto, para este modelo se requiere la predicción de secuencias y al mismo tiempo, dado que los resultados corresponden a puntos continuos en el plano, se requiere del uso de mezclas de densidades gaussianas (ver 4.3.3).

Aunque el planteamiento de este escenario es similar al caso anterior, en el capítulo 6 se exponen diferencias en los métodos implementados, razón por la que desde ya se considera como un escenario separado. En particular, se requerirá de diferentes variables independientes y además los resultados de los diferentes modelos pueden variar entre ambos escenarios.

5.3.2. Predicción de indicadores mentales

En este tercer modelo también se utiliza el enfoque de predicción en ventanas de tiempo. De esta manera, para una ventana de tiempo v se crea un conjunto de etiquetas de K índices mentales $IM(v) = [IM_1(v), IM_2(v), \ldots, IM_K(v)]$. Como no se tiene conocimiento de estos valores reportados por los usuarios, se generan a partir de los datos de los conjuntos de entrenamiento, en particular usando el método de clusterización K-mean. En particular, un usuario tendrá en cada ventana de tiempo una categorización para cada uno de los K indicadores mentales.

Dada las etiquetas de cada ventana, se requiere de un modelo para cada indicador, dado por los registros de las señales anteriores al inicio de la ventana de tiempo v, nombrados como S_{-v}. De esta manera se busca la generación de k clasificadores:

$$IM_k(v) = \mathcal{G}_k(S_{-v})$$

Capítulo 6

Implementación

En este capítulo se detalla la implementación del modelo descrito en la sección 5.3. Se explica la programación realizada y los tratamientos de las señales para la obtención de variables usadas en cada escenario. Para cada modelo se presentan y discuten los resultados obtenidos. En el siguiente capítulo se entregan las conclusiones finales, se exponen las contribuciones logradas y se proponen desafíos para la investigación futura.

6.1. Software y Hardware (Recursos computacionales).

Para el desarrollo de este trabajo, se utiliza un computador portátil. Cada modelo posee sus propios scripts. Para el primer escenario del modelo de predicción de fijaciones, la mayor parte de la programación se hace en Matlab. Para los otros dos modelos, sus programación ha sido llevada a cabo en Python. Las características técnicas se presentan a continuación:

- Sistema Operativo: Windows 10 Home Single Language 64-bits (10.0 compilación 15063)
- Memoria RAM: 16 GB DDR4
- CPU: Intel(R) Core(TM) I7-6700HQ 2.60 GHZ
- Scripts: Matlab R2016a, JAVA 64 bits, Python 3.5.6
- Solver: Gurobi
- Seguimiento Ocular: Tobii Studio 3.4.4
- Framework machine learning: Tensorflow 1.10.0

6.2. Validación cruzada.

En esta sección se explica el método utilizado para crear los conjuntos de datos de entrenamiento y prueba para la implementación de cada uno de los modelos.

En primer lugar, se estructuran los datos mediante ventanas de tiempo, de parámetro $\tau = 5$ segundos. Así, cada registro, es dividido en ventanas identificadas mediante un indicador de tres valores $[\ user, signal, ventana\ ]$, donde $user = \{1, ..., 56\}$, $signal = \{1, 2, 3\}$

[1] y $ventana = \{1, ..., \lceil\ T(user, signal)/\ \tau \rceil\}$, siendo $T(user, signal)$ el tiempo total de la navegación para esa señal de dicho usuario.

Se puede observar que cada ventana de tiempo (u, s, v) en realidad está caracterizada por dos parámetros $T_o(u, s, v)$ y $T_f(u, s, v)$ que representan los límites de tiempo en formato Unix (cantidad de segundos transcurridos desde la medianoche UTC del 1 de enero de 1970, sin contar segundos intercalares), formato necesario para el tratamiento y coordinación de las demás señales.

Una vez formateado los datos, se asume un esquema de validación cruzada. La validación cruzada (en inglés, cross-validation) es una técnica para evaluar los resultados de un análisis o modelo, en la cual, las medidas de desempeño son calculadas en diferentes conjuntos de entrenamiento-prueba, garantizando independencia entre los datos de ambos grupos.

Se utiliza la técnica de validación cruzada ya que es útil en casos en los que se tiene un conjunto limitado de datos. En este caso, considerando ventanas de tiempo de 5 segundos, se tiene un total de 2693 ventanas. Dado que no se tiene un conjunto de datos específico de prueba, el total de datos es dividido en un conjunto de entrenamiento, con el cual se ajustan los parámetros del modelo y un conjunto de prueba en el que se miden los resultados. Existen diferentes maneras de llevar a cabo la validación cruzada, algunas de ellas se explican a continuación:

K-iteraciones: Los datos de la muestra se dividen en K subconjuntos, de los cuales uno de ellos es usado como datos de prueba y el resto ($K - 1$ subconjuntos) como datos de entrenamiento. Este proceso es repetido por K veces, es decir, permitiendo que cada subconjunto sea usado como datos prueba, pronosticando sus resultados. Las métricas finales son la media aritmética de cada métrica calculada en cada partición.

Aleatoria: Es similar al caso anterior, sin embargo la división entrenamiento prueba se realiza aleatoriamente. En este caso existen muestras que quedan sin evaluar y otras que son evaluadas más de una vez. Los resultados finales se calculan de manera similar, es decir, como el promedio de las métricas calculadas en cada partición.

Dejando uno fuera: En inglés Leave-one-out cross-validation (LOOCV), separa los datos de forma que en cada iteración se tenga una muestra para los datos de prueba y el resto para los datos de entrenamiento.

Dada la naturaleza del problema, en el cual se quiere comparar los caminos visuales predichos con los reales, es necesario que cada muestra sea testeada, razón por la cual el enfoque aleatorio no funciona. La técnica LOOCV es preferible ya que permite mejores resultados (errores calculados más bajos, según la literatura), sin embargo, posee un costo computacional muy alto. Para evaluar diferentes configuraciones se utiliza un enfoque 3-iteraciones. Para el modelo final se utiliza el método 10-iteraciones, donde el número fue escogido de manera que cada conjunto de prueba represente un 10 % de los datos.

A continuación se explica el método de división del total de ventanas en los conjuntos de

[1] recordar que "signal" corresponde a cada una de las instancias en que un usuario interactúa con el sitio web en el experimento original.

entrenamiento y prueba manteniendo la independencia entre ambos grupos y permitiendo un balance en la cantidad de muestras, para cada una de las iteraciones de validación cruzada, lo que denominaremos **partición por usuarios**. Aunque este modelo de partición es pensado originalmente para el modelo de predicción de fijaciones en el primer escenario, los resultados pueden ser usados para los otros modelos sin perder demasiado el balance de muestras al considerar diferentes valores de τ en cada uno de ellos.

Partición por usuarios

Las señales de cada usuario se consideran de manera independiente, sin embargo, al querer generar los conjuntos de entrenamiento y prueba disminuyendo la correlación entre los datos que presentan, se prefiere asignar usuarios a los conjuntos en vez de sus registros, lo que se ha denominado, "partición por usuarios". Así, la validación cruzada no permitirá que registros de navegación de un mismo usuario se encuentre tanto en los datos de entrenamiento como en los de prueba al realizar las combinaciones correspondientes al método K-Fold.

Considerando que hay registros de mayor duración temporal que otros en la navegación, se requiere un balance entre el total de ventanas de τ segundos presentes en cada conjunto.

La desventaja de este método es que se ignora el balance de clases. Es decir, existen ciertas clases en los datos de variables objetivo que se encuentran en mayor cantidad que otras en ciertos conjuntos, lo cual puede llevar a sobre-ajuste de los modelos. Sin embargo, esto es tratado mediante métodos de muestreo explicados posteriormente.

Para este método de partición se ejecuta un modelo de programación mixta lineal entera (6.1). Se definen las variables $\{y_{u,\mathrm{i}} | u = 1, \ldots, N, \mathrm{i} = 1, \ldots, K\}$ que valen uno si se asigna el usuario u al fold i, y cero si no. N es el número de usuarios y K la cantidad de conjuntos. Las variables $\{x_\mathrm{i} | \mathrm{i} = 1, \ldots, K\}$ representan la suma de ventanas en el fold i, dada la asignación de usuarios. Así, se asigna cada usuario a un fold, minimizando la diferencia entre la suma total de ventanas en cada fold i y el número promedio de ventanas en todos los fold. En el mejor caso, todos los fold tendrán un número de ventanas igual al promedio $\bar{x}$.

$$\min \quad \sum_{\mathrm{i}=1}^{k} |x_\mathrm{i} - \bar{x}| \tag{6.1a}$$

$$\text{s.t.} \quad \sum_{\mathrm{i}=1}^{K} y_{u,\mathrm{i}} = 1, \qquad u = 1, \ldots, N, \tag{6.1b}$$

$$x_\mathrm{i} = \sum_{u=1}^{N} y_{u,\mathrm{i}} * P_u, \qquad \mathrm{i} = 1, \ldots, K, \tag{6.1c}$$

$$\bar{x} = \frac{\sum_{\mathrm{i}=1}^{K} x_\mathrm{i}}{K}, \tag{6.1d}$$

$$x_\mathrm{i} \geqslant 0, \qquad \mathrm{i} = 1, \ldots, K, \tag{6.1e}$$

$$y_{u,\mathrm{i}} \in \{0, 1\}, \qquad u = 1, \ldots, N \qquad \mathrm{i} = 1, \ldots, K \tag{6.1f}$$

La restricción (6.1b) indica que cada usuario puede ser asignado sólo a uno de los K folds. La restricción (6.1c) define las variables x_i, representando cada una la suma ponderada entre los usuarios asignados y las ventanas que posee en el total de interacciones (P_u). La restricción

(6.1d) define la media aritmética de la cantidad de ventanas en los folds.

Para resolver (6.1) se agregan nuevas variables U_{i}, $\mathrm{i} = 1, \ldots, K$ que representen $|x_{\mathrm{i}} - \bar{x}|$. Se cambia la función objetivo y se agregar las restricciones (6.2b) y (6.2c), que permiten a estas variables lograr la representación deseada. Así se resuelve el modelo (6.2).

$$\min \quad \sum_{\mathrm{i}=1}^{k} U_{\mathrm{i}} \tag{6.2a}$$

$$\text{s.t.} \quad U_{\mathrm{i}} \geq x_{\mathrm{i}} - \bar{x}, \qquad \mathrm{i} = 1, \ldots, K, \tag{6.2b}$$

$$U_{\mathrm{i}} \geq -(x_{\mathrm{i}} - \bar{x}), \qquad \mathrm{i} = 1, \ldots, K, \tag{6.2c}$$

$$U_{\mathrm{i}} \in \mathbb{R}, \tag{6.2d}$$

$$\{(6{,}1b), (6{,}1c), (6{,}1\mathrm{d}), (6{,}1\mathrm{e}), (6{,}1f)\} \tag{6.2e}$$

El modelo (6.2) es resuelto mediante programación en Python con la utilización del solver Gurobi. Los resultados se muestran en las tablas 6.1 y 6.2 al generar particiones en 3 y 10 folds respectivamente. Se observa que dentro de una partición, no todos los conjuntos presentan la misma cantidad de participantes. Por otro lado, la cantidad de ventanas presentadas en cada uno se encuentra bien balanceado. Se concluye que los resultados obtenidos son útiles y no se requiere de métodos adicionales como, por ejemplo, la simulación de datos.

Conjunto	**Total de Usuarios**	**Usuarios considerados**	**N° de ventanas**
Fold 1	15	[2, 3, 4, 10, 11, 16, 17, 28, 30, 33, 34, 35, 38, 39, 48]	795
Fold 2	18	[5, 8, 12, 13, 14, 18, 20, 24, 25, 26, 29, 37, 40, 41, 42, 46, 47, 49]	796
Fold 3	18	[1, 6, 7, 9, 15, 19, 21, 22, 23, 27, 31, 32, 36, 43, 44, 45, 50, 51]	796

Tabla 6.1: Resultados partición por participantes para 3-Fold.

Conjunto	**Total de Usuarios**	**Usuarios considerados**	**N° de ventanas**
Fold 1	5	[34, 37, 38, 46, 51]	239
Fold 2	4	[7, 23, 32, 50]	239
Fold 3	6	[4, 8, 17, 27, 31, 36]	239
Fold 4	6	[6, 12, 13, 26, 33, 42]	238
Fold 5	6	[9, 11, 21, 40, 45, 47]	238
Fold 6	5	[1, 20, 22, 24, 25]	239
Fold 7	3	[35, 39, 41]	239
Fold 8	6	[3, 14, 15, 18, 29, 44]	239
Fold 9	5	[2, 16, 28, 30, 43]	238
Fold 10	5	[5, 10, 19, 48, 49]	239

Tabla 6.2: Resultados partición por participantes para 10-Fold.

6.3. Modelo prediccion de fijaciones: primer escenario

Antes de comenzar con la implementación de cada etapa, se explica brevemente la selección de parámetros, la estructuración de los resultados y el esquema de programación usado.

6.3.1. Selección de parámetros: Definición de áreas de interés

En este modelo se establecen áreas de interés de manera semántica (véase sección 2.1.1) a partir de las componentes explicadas en la sección 5.2. Se definen seis áreas de interés, cinco mostradas en la figura 6.1 más la ventana de registro como sexto AOI. El resto de la página, se considera Whitespace.

Figura 6.1: Agrupación semántica de las áreas de interés.

El área de interés número 1 es dinámica, ya que las componentes “logo” y “barMenu” se desplazan en conjunto con el usuario, presentándose siempre en la parte superior de la pantalla. Al mismo tiempo, los menú desplegables “selección de país” y “menú” forman parte de esta área de interés y su activación dependerá de la interacción del usuario al presionarlos.

Las áreas de interés 2 y 3 corresponden a las noticias presentes en el sitio. Se define el área superior con un total de tres noticias, ya que por resolución de la pantalla, el hacer un desplazamiento lateral en totalidad hacia abajo, se visualiza completamente el AOI 3. Estas áreas se componen de los títulos de las noticias y las imágenes que las acompañan.

Las áreas 4 y 5 corresponden a los anuncios publicitarios. Dado que existen 4 en total, estos se segmentan en dos áreas, cubriendo igual tamaño dentro de la página.

El área 6 corresponde a la ventana de registro presentada cuando el usuario acciona el botón correspondiente. Esta ventana es completamente dinámica y puede cambiar de tamaño. Además, su posición sobrepone a otras AOI's. Aunque cuando el AOI 6 está activa, el resto de la página se oscurece, aún pueden verse parte de las AOI's 1 a 5, por lo que se consideraran que dichas áreas pueden estar activas aún cuando AOI 6 lo está, en las zonas libres de traslape.

6.3.2. Creación de caminos visuales

En cada ventana de tiempo, en los conjuntos de entrenamiento y prueba, se generan las secuencias correspondientes a los caminos visuales. Esto se lleva a cabo con la utilización de los datos de fijaciones realizadas por cada usuario, sus coordenadas (x, y) y sus duraciones. Así, se establecen dos métodos.

En el primer método, llamado **Camino Visual Simple**, considera la secuencia de AoI's sin la dimensión temporal. Es decir, se genera una secuencia de AoI's atendidas por el usuario de manera ordenada, pero sin considerar cuanto tiempo permanece en cada una de ellas, ni en que instante de la ventana son observadas. Para ello, se identifican las fijaciones dentro de las componentes pertenecientes a cada AoI y se genera una secuencia de AoI's atendidas, sin considerar las fijaciones realizadas en whitespace. Finalmente, en caso de que la secuencia presente repetición de AoI's de manera adyacentes, estas son eliminadas. En la figura 6.2 se presenta un ejemplo de la generación de caminos visuales simples.

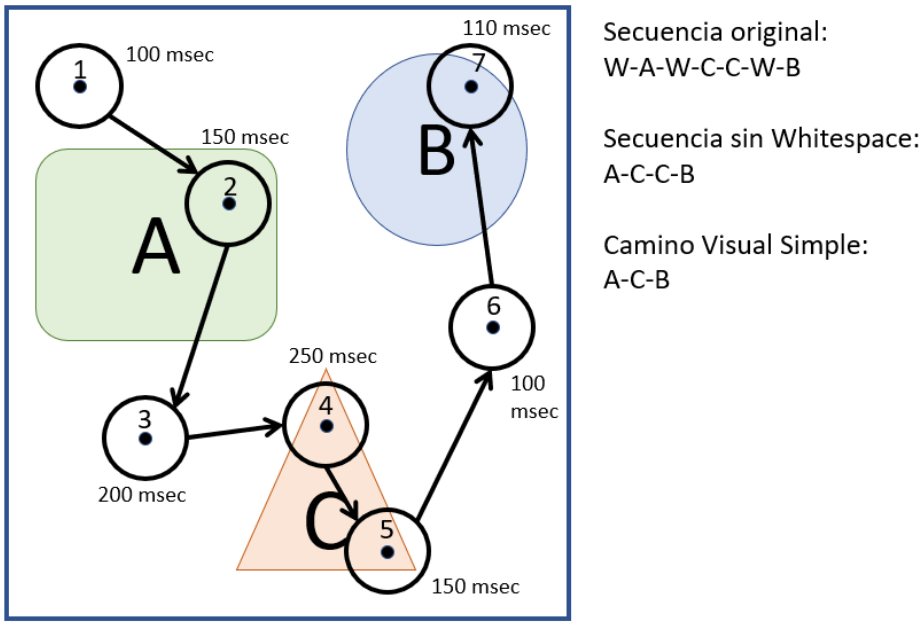


Figura 6.2: Generación de caminos visuales por el primer método.

En la imagen, se presenta un estímulo con tres áreas de interés definidas (A, B, C) y siete fijaciones registrada durante una ventana de tiempo. Las fijaciones son dibujadas con un circulo negro y su duración está en milisegundos. Además, se presenta un número que

permite conocer el orden de la secuencia de fijaciones registrada y flechas ficticias para mayor comprensión. La secuencia original presenta tres fijaciones en Whitespace que no son consideradas, y la repetición de C es eliminada. Además, las fijaciones de duración menor a 8 milisegundos son eliminadas.

La predicción de este tipo de resultados permite generar la secuencia de orden de las áreas de interés, no así su ubicación temporal ni duración dentro de la ventana. Sin embargo, presenta menor complejidad y permite un primer acercamiento a los resultados deseados.

El segundo método se denomina **Camino visual Temporal**, ya que intenta que los resultados obtenidos con la predicción de secuencias reconstruyan las secuencias de áreas de interés desarrolladas por los usuarios temporalmente. En la imagen 6.3 se entrega un ejemplo.

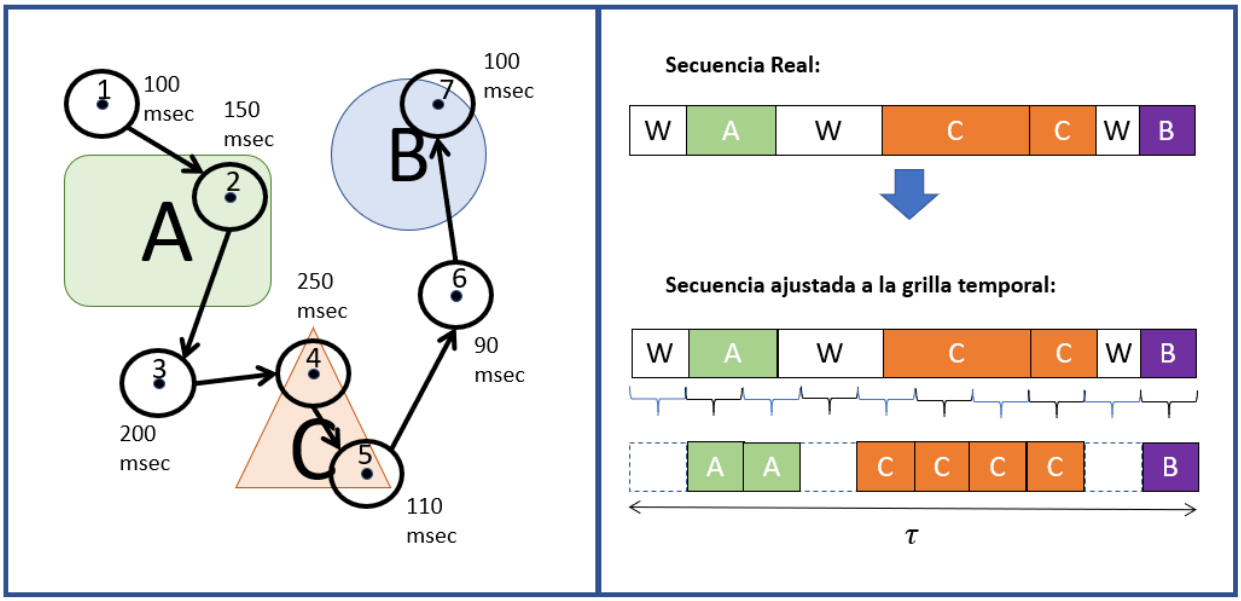


Figura 6.3: Conjunto de fijaciones y línea temporal para una ventana de tiempo.

A la izquierda de la imagen 6.3 se presenta el ejemplo de la figura 6.2. A la derecha se muestra el diagrama temporal de las fijaciones registradas en la ventana de tiempo.

La ventana de tiempo es dividida en compartimientos de igual tamaño, en este ejemplo de $q = 100$ milisegundos. En cada uno de estos compartimientos se considera el área de interés atendida como aquella con mayor tiempo de fijación. Es decir, un compartimiento no mostrará más de un área de interés atendida. Al mismo tiempo, si el AOI es atendido por menos de la mitad del tiempo que dura el compartimiento, se considerará aquel espacio como atención a whitespace.

Este método, a diferencia de un Camino Visual Simple, permite la reconstrucción del camino visual en intervalos aproximados de tiempo. Para llevarlo a cabo solamente es necesaria la elección del valor para el parámetro q, el cual en conjunto al parámetro τ, entregará secuencias de largo $\lfloor \tau/q \rfloor + \mathbb{1}_{rem(\tau,q)>0}$.

6.3.3. Esquema de la solución

El archivo principal de la programación es ejecutado en Matlab. En la imagen 6.4 se presenta el esquema de solución desarrollado.

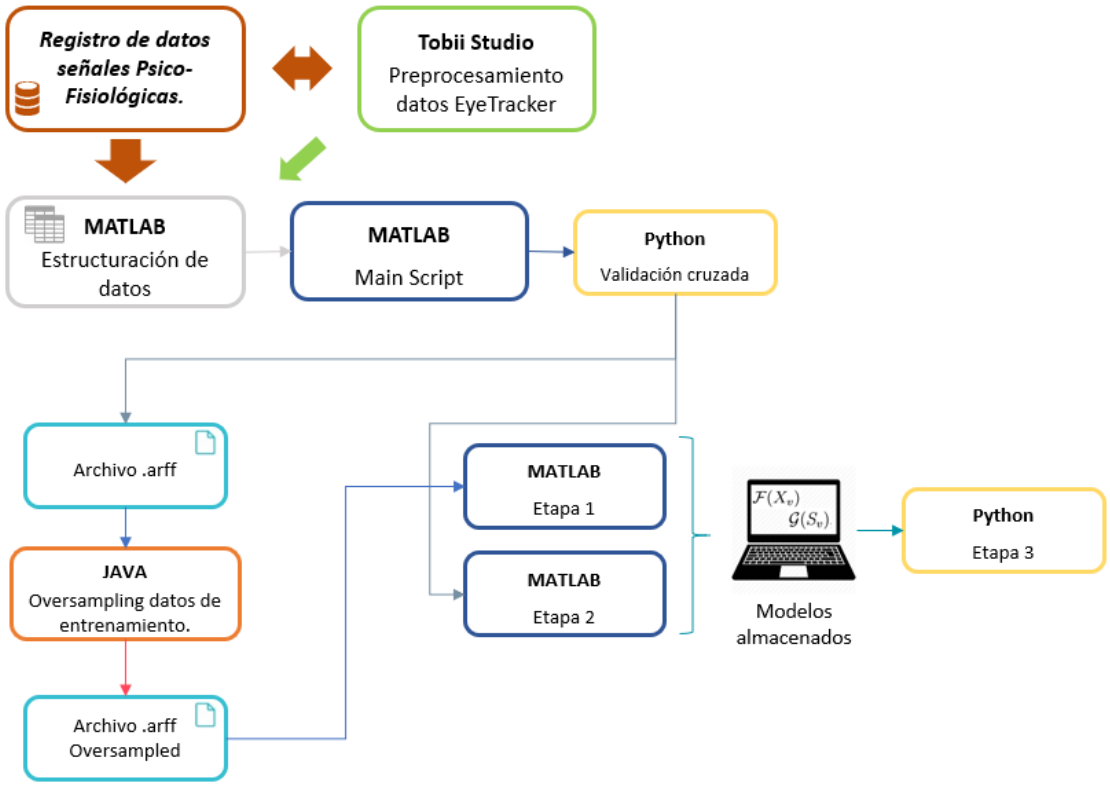


Figura 6.4: Esquema de la solución.

Antes de cualquier proceso en el archivo principal, se requiere el pre-procesamiento de datos de seguimiento ocular, llevado a cabo con el software Tobii Studio v3.4.4, identificando las componentes explicadas en la sección 2.1.1.

Luego, los datos son estructurados para su procesamiento durante la ejecución, según lo expuesto en la sección 5.2.1, siendo entregados al archivo principal.

Se programan scripts en Pyhton para resolver problemas optimización lineal entera mixta con la ayuda del solver Gurobi (véase sección 6.2), para crear conjuntos de entrenamiento-prueba de los datos. Se utiliza programas compilados en JAVA para el balance de muestras de datos (véase sección 6.3.4).

Los datos de entrenamiento-prueba son entregados para el procesamiento en las etapas 1 (véase 6.3.4) y 2 (véase 6.3.5). La etapa de "Modelos Almacenados" representa el almacenamientos de datos y modelos procesados para los conjuntos de entrenamiento, los cuales son posteriormente aplicados a los conjuntos de prueba para el pronóstico, según el algoritmo de la etapa 3 (véase 6.3.6) desarrollado en Python.

6.3.4. Etapa 1: Predicción de intención de visita AOI's.

En esta sección se presenta los métodos de balance de muestras 6.3.4 para la aplicación de métodos de clasificación multi-etiqueta, la creación de variables independientes 6.3.4 y los métodos de clasificación multi-etiqueta en conjunto con los resultados obtenidos.

Balance de las muestras

En los problemas de clasificación multi-etiqueta, es común que las frecuencias de las etiquetas no sean las mismas. Este trabajo no es la excepción, es decir, en los diferentes conjuntos de entrenamiento, existen etiquetas que están activas una cantidad mayor de veces que otras.

En un clasificador binario, ocurre que el desbalance de muestras puede provocar una inclinación a pronosticar la clase mayormente activa. Por ejemplo, en los casos de detección de cáncer ocurre que la mayor parte del tiempo un usuario no posee la enfermedad, y tan sólo un 1 % de los usuarios estén realmente enfermos. Por lo tanto, si esta proporción se mantiene en el conjunto de prueba, un clasificador que prediga siempre el caso de que el paciente no está enfermo acertará el 99 % de los casos (tasa conocida como "Bayes rate"), sin embargo, no se logrará un modelo útil.

Lo anterior se puede generalizar para clasificadores multi-etiqueta. Para solucionar existen principalmente tres enfoques "Data Resampling", "Algorithm adaptation" y "Cost-sensitive learning". En este trabajo se decide la utilización del primero de ellos.

Los métodos de remuestreo se basan en la eliminación de muestras de las clases mayoritarias de etiquetas, agregar muestras de las etiquetas de menor frecuencia, o ambas operaciones. Existen diferentes métodos desarrollados con este fin, en particular, en este trabajo se utiliza el método MLSMOTE [25].

MLSMOTE considera varias etiquetas como clases minoritarias y toma muestras donde estas etiquetas aparezcan como semilla para la generación de nuevas instancias. Para cada una, busca los vecinos más cercanos obteniendo las características de las nuevas muestras mediante técnicas de interpolación. Por lo tanto, las nuevas instancias son sintéticas más que clonaciones de otros datos. Para la generación de etiquetas, existen tres enfoques, los primeros dos correspondientes a la intersección o unión de las etiquetas presentes en los vecinos más cercanos, y la tercera (con mejores resultados y que sera usada aquí), utiliza sistemas de puntuación de etiquetas presentes en los vecinos más cercanos, activando aquellas que se presenten en más de la mitad de los casos.

El método MLSMOTE se ejecuta en JAVA, mediante la programación proporcionada en internet [74], para cada uno de los conjuntos de entrenamiento. Se desarrollan los scripts que permitan la conversión de datos para la manipulación en cada uno de los lenguajes de programación utilizados.

Creación de variables para clasificación multi-etiqueta

Para la clasificación multi-etiqueta necesaria para calcular la intención de visita de los usuarios a las áreas de interés en la próxima ventana temporal, se genera un conjunto de variables independientes, las cuales son procesadas a partir de las señales de seguimiento

ocular e historial de navegación.

Sean $\mathrm{i} = 1, \ldots, 6$ las área de interés definidas y $j = 1, \ldots, N$ las componentes presentes en la página web. Dada la cantidad de componentes existentes, se crean 226 variables, las cuales se definen a continuación. En primer lugar se definen las relacionadas a las áreas de interés y componentes:

- *user* : indicador de usuario. Toma valores $1, \ldots, 61$. No es usada como variable en ningún modelo de clasificación.
- *signal*: indicador de la interacción del usuario. Valor 1 para la experiencia sin delay, 2 para el caso en que la carga de la página demora 500 milisegundos y 3 en el caso de los 5 segundos. Este valor no es usado como variable en los modelos de clasificación.
- *ventana*: indicador de ventana para un usuario en una interacción dada. Tampoco es usada como variable en los modelos de clasificación.
- $\{AoI\}_{\mathrm{i}}$: corresponden a la variable dependiente en los modelos de clasificación (etiquetas). Toma el valor 1 si es que el área de interés es visitada en la ventana de tiempo y 0 en caso contrario.
- $\{AoI\_tuv\}_{\mathrm{i}}$: Las variables "tiempo desde la última visita" de cada área de interés se calculan como la diferencia entre el tiempo de inicio de la ventana actual y el tiempo de la ultima vez en que $\{AoI\}_{\mathrm{i}} = 1$. Los valores varían en el intervalo $(8, \infty)$ milisegundos, donde el límite inferior corresponde al tiempo de registros de datos por parte del EyeTracker. Para el caso en que no exista la ultima visita, se evita tener valores iguales al tiempo de navegación del usuario, asignando un valor fuera del rango (valor cero para la variable).
- $\{AoI\_r1\}_{\mathrm{i}}$, $\{AoI\_r2\}_{\mathrm{i}}$ y $\{AoI\_r3\}_{\mathrm{i}}$ corresponde a variables de serie de tiempo. Si $\{AoI\}_{\mathrm{i}} = 1$ para una ventana v, entonces $\{AoI\_rt\}_{\mathrm{i}} = 1$ para la ventana $(v + t)$. Estas tres variables son independientes, debido a que las ventanas de tiempo lo son.
- $\{AoI\_finr1\}_{\mathrm{i}}$: Esta variable vale 1 si es que la ventana precedente finaliza con una fijación en la AOI correspondiente y 0 si no. Esta variable captura el hecho de que las fijaciones pueden ser interrumpidas por las transiciones entre ventanas, es decir, que las fijaciones que comienzan en una ventana de tiempo pueden terminar en la siguiente.
- $finr1$: Corresponde a un indicador del AoI en el cual se finaliza la ventana anterior. Es una combinación de las variables $\{AoI\_finr1\}_{\mathrm{i}}$ por lo que no pueden ser usadas en conjunto. Se calcula como:

$$finr1 = \sum_{\mathrm{i}=1}^{6} \mathrm{i} * AoI\_finr1_{\mathrm{i}}$$

- $\{Heat\_AoI\}_{\mathrm{i}}$: En la literatura [56] se establece que en general los humanos tienden a ser los mejores predictores de donde miran otros usuarios. Sin embargo, estos no predicen exactamente la dirección de otros usuarios debido a la variabilidad de factores humanos y la complejidad del estímulo. Por otro lado, el desempeño de esta variable mejora al aumentar el número de usuarios a considerar. De esta forma, utilizando las señales

registradas en el conjunto de entrenamiento, se genera una variable de frecuencia de visualizaciones en cada una de las áreas de interés definidas, en las ventanas de tiempo realizadas por otros usuarios v_j en los tiempos correspondientes a la ventana de tiempo analizada v, es decir:

$$Heat\_AoI(v) = \frac{\sum_{j=1}^{N_train} AOI(v_j)}{N_{train}}$$

- $\{[Componente]\}_j$: Similar a la variable $\{AoI\}_\text{i}$ pero por componente. No son usadas en los métodos de clasificación sino que para la construcción de las variables correspondientes a las áreas de interés.

- $\{[Componente]\_his\}_j$: La variable "historial" para cada componente toma el valor 1 si es que la componente ha sido visitada durante la navegación y 0 si no.

- $\{[Componente]\_fin\}_j$: Esta variable indica si una ventana termina con el ultimo registro equivalente a la visualización de la componente por parte del usuario.

- $\{[Componente]\_finr1\}_j$: Similar a las variables $\{AoI\_finr1\}_\text{i}$ pero por componente, construida a partir de la variable $[Componente]\_fin$.

- $\{[Componente]\_tuv\}_j$: Las variables "tiempo desde la última visita" para cada componente es similar a $\{AoI\_tuv\}_\text{i}$.

- $\{[Componente]\_tpv\}_j$:Las variables "tiempo promedio de las visitas" para cada componente entregan el tiempo promedio en milisegundos de las fijaciones realizadas en la componente durante la navegación. En caso de que no existan fijaciones en dicha componente, se le asigna el valor 0.

- $\{[Componente]\_tpev\}_j$: Las variables "tiempo promedio entre visitas" para cada componente entregan el tiempo promedio en milisegundos que ha tardado el usuario en realizar una nueva fijación en la componente. Se le asigna el valor 0 en caso de no existir fijaciones.

Además, se definen variables en función de las coordenadas de los registros de posición de mirada para el usuario, tanto en posición, velocidad y aceleración en la coordenada X. Las definiciones para las variables de las coordenadas Y ($coordY$, $MeanY$, $StdY$, $VelY$, $MeanVelY$, $StdVelY$, $AclY$, $MeanAclY$ y $StdAclY$) son similares.

- $coordX$: Se considera la última posición de la mirada antes de comenzar la ventana de tiempo, registrada para el usuario.

- $MeanX$, $StdX$: Media y la desviación estándar de la posición en la coordenada X registradas en la navegación antes de comenzar la ventana de tiempo.

- $VelX$: Derivada de los datos de posición en la coordenada X. Esta variable toma el valor correspondiente al último registro de aceleración durante la navegación.

- $MeanVelX$, $StdVelX$: Media y la desviación estándar de los datos de velocidad en la coordenada X registradas en la navegación antes de comenzar la ventana de tiempo correspondiente.

- $AclX$: Derivada de los datos de velocidad en la coordenada X. Esta variable toma el valor correspondiente al último registro de aceleración durante la navegación.
- $MeanAclX$, $StdAclX$: Media y la desviación estándar de los datos de aceleración en la coordenada X registradas en la navegación antes de comenzar la ventana de tiempo correspondiente.

Finalmente se agregan variables relacionadas a los registros de seguimiento ocular del usuario, considerando las fijaciones y sacadas desarrolladas:

- $NFij$: Número de fijaciones realizadas por el usuario durante su tiempo de navegación.
- $TPromFij$, $TMaxFij$, $TMinFij$: Tiempo promedio, tiempo máximo y mínimo de la duración de las fijaciones realizadas por el usuario durante su navegación.
- $NSac$: Número total de sacadas realizadas por el usuario hasta el momento anterior al comienzo de la ventana de tiempo actual.
- $APromSac$, $AMaxSac$, $AMinSac$: Amplitud promedio, máxima y mínima de las sacadas realizadas por el usuario hasta el momento anterior al comienzo de la ventana actual.

Métodos clasificación multi-etiqueta y resultados

Las métricas de evaluación para la clasificación multi-etiqueta se definen en la sección 4.1.3. Cada una de ellas es promediada aritméticamente entre los K conjuntos de prueba utilizados. Se busca maximizar las métricas Área under curve (Auc), Subset Accuracy (Exact), Hamming Score (Hamming), MacroF1 y MicroF1, F-measure (F-mea), Accuracy (Acc), Precision (Pre) y Ranking Score (Rank). Se busca minimizar OneError (One), Coverage (Cov), el tiempo necesario para entrenar el modelo utilizando el conjunto de entrenamiento (TrainT) y el tiempo necesario para aplicar el modelo al conjunto de prueba (TestT).

Los modelos de clasificación utilizados se ejecutan mediante la programación entregada en [57] ("MLC Toolbox"), ejecutado en Matlab. Se genera un propio script para la ejecución de la metodología de clasificación en cadena, de acuerdo a lo expuesto en 4.1.4. Se prueban diferentes modelos de clasificación y configuraciones.

Se ejecutan los seis métodos de selección de variables expuestos en 4.1.2 en conjunto a los métodos de clasificación. Los resultados obtenidos se resumen en las tablas presentes en anexos (sección 8.1), las cuales consideran las configuraciones con las mejores métricas para cada método de selección de variables. Allí se especifican los parámetros del modelo, los valores de las métricas en promedio y su desviación estándar entre paréntesis, para los K conjuntos de prueba. A continuación se describen brevemente las configuraciones utilizadas para los modelos expuestos en la sección 4.1.5:

Ridge Regression: Este modelo es usado mediante los enfoques de Binary Relevance y Clasification Chain, fijando el número de variables a utilizar en el rango $5:5:40$ escogidas a través de los métodos de selección de variables. Una vez fijadas las variables a utilizar, se escogen diferentes valores del parámetro λ y fijo este, se estiman los parámetros β. Bajo el enfoque de binary relevance y clasification chain se prueban los modelos con variaciones en

el parámetro $\lambda = 0{,}25 : 0{,}25 : 2$.

KNN: KNN es usado mediante los enfoques de Binary Relevance y Clasification Chain, fijando el número de variables a utilizar en el rango $5 : 5 : 40$ y los valores del parámetro $K = 5 : 5 : 30$.

SVM: SVM es usado mediante los enfoques de Binary Relevance y Clasification Chain, fijando el número de variables a utilizar en el rango $5 : 5 : 40$ escogidas a través de los métodos de selección de variables, y el parámetro $C = 0{,}2 : 0{,}2 : 2$.

MLKNN: Este método se utiliza directamente, sin los enfoques de Binary Relevance o Classification Chain, fijando el número de variables a utilizar en el rango $5 : 5 : 40$ escogidas a través de los métodos de selección de variables, el parámetro relativo al número de vecinos en $K = 15$ y el parámetro $smooth = 1$. No se prueban más configuraciones ya que éstas son costosas computacionalmente.

Resultados de la clasificación

A continuación se presentan los resultados de las mejores configuraciones según los modelos especificados anteriormente. La tabla 6.3 muestran cada métrica como un valor promedio entre los K conjunto de prueba utilizados. Entre paréntesis se encuentran los valores de la desviación estándar.

De las diferentes combinaciones puestas a prueba, se selecciona los modelos que presenten mejor medida de "accuracy" y "exact", al mismo tiempo que se prefiere los casos en que se utiliza una menor cantidad de variables.

Se utilizan los métodos de "Binary Relevance" y "Classification Chain". No se usa el método de "Power Labelset" ya que este considera relaciones entre etiquetas (coordenadas del vector objetivo) que no necesariamente son acordes a la realidad y son integradas en la creación de datos sintéticos.

Ridge Regression Binary Relevance: Los mejores resultados se consiguen con el método de selección de variables RFS, usando 30 variables predictivas y el parámetro $\lambda = 1{,}75$.

Ridge Regression Classification Chain: Los mejores resultados se consiguen con el método de selección de variables RFS, usando 30 variables predictivas y el parámetro $\lambda = 1{,}75$.

KNN Binary Relevance: Los mejores resultados se consiguen con el método de selección de variables MLMIM, usando 10 variables predictivas y el parámetro $K = 20$.

KNN Classification Chain: Los mejores resultados se consiguen con el método de selección de variables MLJMI, usando 10 variables predictivas y el parámetro $K = 15$.

SVM Binary Relevance: Los mejores resultados se consiguen con el método de selección de variables MLMIM, usando 40 variables predictivas y el parámetro $C = 0{,}2$.

SVM Clasification Chain: Los mejores resultados se consiguen con el método de selec-

ción de variables RFS, usando 30 variables predictivas y el parámetro $C = 0{,}4$.

MLKNN Los mejores resultados se consiguen con el método de selección de variables MLMIM, usando 10 variables predictivas.

Tabla 6.3: Mejores resultados clasificación multi-etiqueta.

	AUC	Exact	Hamm	MacroF1	MicroF1	Fscore	Acc	Pre	Rank	One	Cov
Ridge Regression Binary relevance	0,844 (0,006)	0,285 (0,010)	0,782 (0,004)	0,598 (0,005)	0,640 (0,007)	0,658 (0,011)	0,556 (0,010)	0,877 (0,007)	0,872 (0,007)	0,124 (0,016)	0,307 (0,008)
Ridge Regression clasification Chain	0,844 (0,006)	0,285 (0,010)	0,782 (0,004)	0,598 (0,005)	0,640 (0,007)	0,658 (0,011)	0,556 (0,010)	0,877 (0,007)	0,872 (0,007)	0,124 (0,016)	0,307 (0,008)
KNN Binary Relevance	0,843 (0,005)	0,303 (0,008)	0,786 (0,005)	0,606 (0,009)	0,647 (0,006)	0,670 (0,009)	0,571 (0,008)	0,879 (0,004)	0,858 (0,006)	0,110 (0,013)	0,312 (0,009)
KNN Classification Chain	0,839 (0,005)	0,302 (0,006)	0,786 (0,004)	0,613 (0,010)	0,660 (0,004)	0,679 (0,007)	0,579 (0,006)	0,878 (0,003)	0,848 (0,004)	0,103 (0,012)	0,319 (0,009)
SVM Binary Relevance	-	0,291 (0,012)	0,779 (0,005)	0,613 (0,009)	0,646 (0,009)	0,663 (0,013)	0,564 (0,013)	-	-	-	-
SVM Classification Chain	-	0,287 (0,009)	0,777 (0,004)	0,611 (0,007)	0,644 (0,005)	0,662 (0,009)	0,561 (0,009)	-	-	-	-
MLKNN	0,844 (0,005)	0,301 (0,009)	0,786 (0,005)	0,610 (0,008)	0,653 (0,006)	0,675 (0,009)	0,574 (0,009)	0,882 (0,005)	0,862 (0,006)	0,109 (0,011)	0,308 (0,009)

De los resultados anteriores, se decide la utilización del modelo KNN con Binary relevance, usando parámetros de $K = 20$ (número de vecinos considerados por el método), diez variables independientes y el método de selección de MLMIM, configuración que presenta los mejores resultados para las métricas de accuracy y accuracy subset. Las variables escogidas son: $coordY$, AoI_6_r1, AoI_1_finr1, AoI_3_finr1, AoI_2_finr1, $coordX$, AoI_3_r1, AoI_6_finr1, AoI_1_r1, $MeanY$, $Registro\_finr1$, $MenuF\_finr1$.

Análisis etiquetas y reestructuración de AOI's.

Analizando los resultados de la clasificación multi-etiqueta se observa que la utilización de 6 variables dependientes causa valores de accuracy y subset accuracy cercanos a 60 % y 30 % respectivamente. Por lo tanto, en esta sección se evalúa si es posible mejorar dichas métricas a partir de un cambio en las áreas de interés definidas. En particular, se realiza un análisis de los datos que permita conocer las componentes más visitadas.

En la literatura, se manifiesta que la definición de Áreas de interés a partir de los mismos datos que serán usados para la generación de modelos permite mejores resultados, sin embargo, puede provocar el error de generar áreas de poco contenido semántico, motivados por los resultados conocidos en una evaluación posterior al registro de los datos. Es por esa razón que en un primer enfoque se definen dichas áreas a partir de definiciones semánticas. Sin embargo, en este punto, puede ser de utilidad la evaluación considerando lo ya mencionado.

En la figura 6.5 se observa que las componentes no utilizadas en la definición inicial de las áreas de interés representan solo un 2 % y las fijaciones realizadas en errores producidos en la página representan un 6 % del total. Al mismo tiempo, se observa un 16 % de fijaciones realizadas en zonas de whitespace. Por lo tanto, basados en el hecho que las zonas de whitespace no poseen componentes con contenido semántico, la definición utilizada en un principio

parece conveniente.

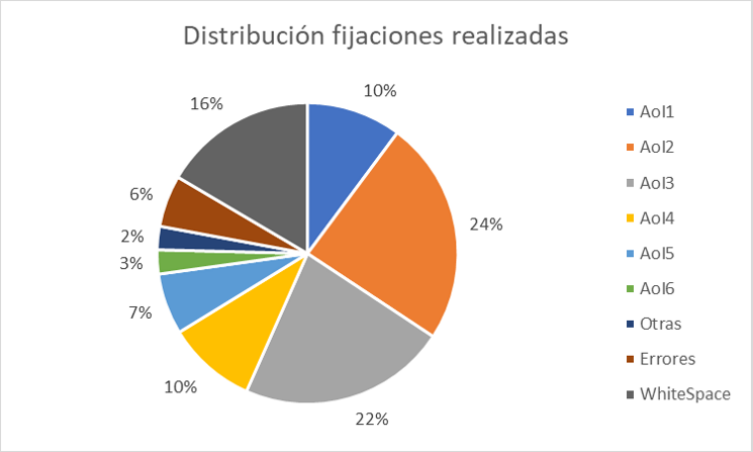


Figura 6.5: Porcentaje de las fijaciones totales realizadas.

En el gráfico 6.6 se observa el porcentaje de uso de cada AoI definida semánticamente en cada una de los conjuntos de entrenamiento. Cada columna refleja la utilización dejando fuera algún grupo de usuarios. Esto se calcula como el número de fijaciones identificadas en cada cada componente de cada área de interés definida, sobre el total de fijaciones realizadas en algún área (dejando fuera las fijaciones realizadas en zonas de whitespace del estímulo visual). Se puede observa lo siguiente:

- El AoI1, de naturaleza dinámica presenta una utilización promedio de 14 %, por lo cual, su uso es importante durante la navegación. Esto ocurre porque la barra de menú es frecuentemente consultada por el usuario para mejorar su experiencia en el sitio web.
- Las AoI 2 y 3, correspondientes a las noticias dentro del sitio, presentan una utilización promedio de 32 % y 30 % respectivamente. Esto hace pensar que a pesar que el AoI 3 posee dos componentes extras (imagen y texto de la noticia 7), el hecho de que se ubique en la zona inferior al AoI 2 (necesitando scroll para su visualización en pantalla) permite que el número de fijaciones en esta zona sea menor. Por lo tanto, los resultados parecen ser convincentes para asumir la definición de estas dos áreas.
- Las AoI 4 y 5, correspondientes a los anuncios publicitarios (banners) del sitio presentan una utilización promedio de 13 % y 9 % respectivamente. Bajo la hipótesis que todas las fijaciones poseen una duración constante, se puede pensar que estas dos AoI presentan una utilización baja en relación a las otras áreas de interés estáticas (AoI 2 y 3).
- Se observa que el AoI 6 correspondiente al registro de usuarios tiene una utilización promedio de 4 %, el menor índice de todas las áreas definidas. Esto se debe a que no todos los usuarios hacen uso de esta pestaña durante la navegación y los que lo hacen, concentran dichas fijaciones en ventanas de tiempo específicas (por ejemplo, se presenta una mayor utilización de esta área en el fold 7).

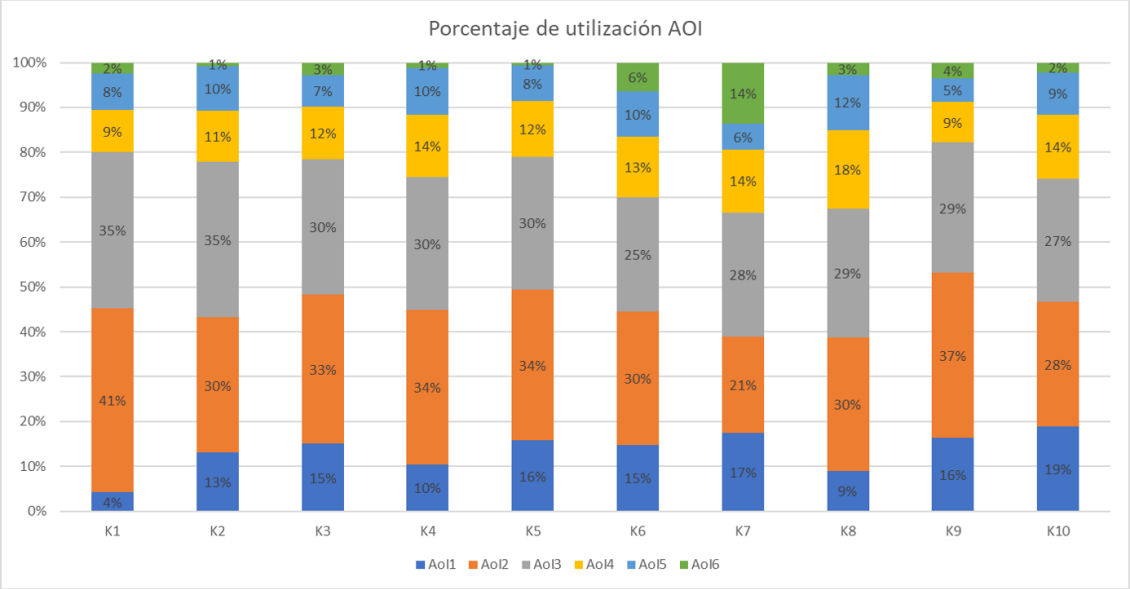


Figura 6.6: Porcentaje de utilización de cada AoI.

Por lo tanto, de lo anterior se pueden tomar algunas consideraciones para la posible mejora de resultados. El AoI 6, que no presenta demasiada utilización, es eliminado de las variables dependientes en los modelos multi-etiqueta. Esto considerando que para su aparición en pantalla, es necesario que el usuario presione el botón correspondiente, y por lo tanto se puede saber si el usuario dirigirá su vista allí en función de los clics realizados durante su navegación.

Se agrupan las AoI 4 y 5, de manera que el porcentaje de utilización sea similar a las del resto de áreas estáticas (AoI 2 y 3). Así, se evalúan los resultados de la clasificación multi-etiqueta usando sólo 4 etiquetas. A continuación se ejecuta la misma metodología presentada en la sección 6.3.4, para las nuevas áreas de interés. Las variables relacionadas a las áreas de interés agrupadas son computadas nuevamente. Los mejores resultados para cada método se presentan en anexos (sección 8.1).

Ridge Regression Binary Relevance: Los mejores resultados se consiguen con el método de selección de variables RFS, usando 40 variables predictivas y el parámetro $\lambda = 0{,}5$.

Ridge Regression Classification Chain: Los mejores resultados se consiguen con el método de selección de variables RFS, usando 40 variables predictivas y $\lambda = 0{,}5$.

KNN Binary Relevance: Los mejores resultados se consiguen con el método de selección de variables MLJMI, usando 10 variables predictivas y el parámetro $K = 15$.

KNN Classification Chain: Los mejores resultados se consiguen con el método de selección de variables MLMIM, usando 10 variables predictivas y el parámetro $K = 30$.

SVM Binary Relevance: Los mejores resultados se consiguen con el método de selección de variables RFS, usando 40 variables predictivas y el parámetro $C = 0{,}2$.

SVM Clasification Chain: Los mejores resultados se consiguen con el método de selección de variables MLMIM, usando 30 variables predictivas y el parámetro $C = 1{,}8$.

MLKNN Los mejores resultados se encuentran con el método de selección de variables MLJMI, usando 10 variables predictivas.

Tabla 6.4: Mejores resultados clasificación multi-etiqueta nuevos AOI's.

	AUC	Exact	Hamm	MacroF1	MicroF1	Fscore	Acc	Pre	Rank	One	Cov
Ridge Regression Binary relevance	0,805 (0,007)	0,311 (0,011)	0,729 (0,007)	0,686 (0,010)	0,698 (0,009)	0,695 (0,015)	0,597 (0,014)	0,900 (0,006)	0,832 (0,009)	0,139 (0,021)	0,333 (0,011)
Ridge Regression Classification Chain	0,805 (0,007)	0,311 (0,011)	0,729 (0,007)	0,686 (0,010)	0,698 (0,009)	0,695 (0,015)	0,597 (0,014)	0,900 (0,006)	0,832 (0,009)	0,139 (0,021)	0,333 (0,011)
KNN Binary Relevance	0,800 (0,006)	0,310 (0,009)	0,728 (0,005)	0,682 (0,009)	0,693 (0,008)	0,692 (0,014)	0,595 (0,012)	0,903 (0,005)	0,810 (0,007)	0,118 (0,020)	0,337 (0,009)
KNN Classification Chain	0,806 (0,007)	0,313 (0,009)	0,733 (0,004)	0,670 (0,007)	0,682 (0,006)	0,685 (0,013)	0,588 (0,011)	0,909 (0,004)	0,833 (0,007)	0,110 (0,018)	0,331 (0,011)
SVM Binary Relevance	-	0,307 (0,010)	0,723 (0,006)	0,673 (0,010)	0,682 (0,010)	0,680 (0,016)	0,583 (0,015)	-	-	-	-
SVM Classification Chain	-	0,312 (0,011)	0,724 (0,006)	0,673 (0,009)	0,681 (0,008)	0,680 (0,014)	0,584 (0,013)	-	-	-	-
MLKNN	0,805 (0,007)	0,306 (0,008)	0,730 (0,005)	0,679 (0,008)	0,691 (0,008)	0,691 (0,014)	0,593 (0,012)	0,902 (0,005)	0,822 (0,007)	0,116 (0,020)	0,336 (0,011)

De los resultados anteriores, se escoge el método KNN con Clasification Chain. Este modelo presenta el valor más grande para la métrica de Subset Acurracy ("Exact") no así para Accuracy, la cual se mantiene cercana al resto de los modelos. Sin embargo, a diferencia de otros modelos, utiliza menor cantidad de features, lo que permite mayor rapidez en el procesamiento de nuevos datos. Las features seleccionadas entre los diferentes folds son: $coordY$, AoI_1_finr1, AoI_3_finr1, AoI_2_finr1, AoI_1_r1, $MenuF\_finr1$, $coordX$, $MeanY$, $finr1$, AoI_3_r1, AoI_4_finr1.

El accuracy para las áreas de interés 1 a 4 son respectivamente, 0.786, 0.691, 0.752, 0.703 (promedio de 0.733). Para el mejor modelo en el caso de seis áreas de interés se obtienen medidas de accuracy de 0.769, 0.702, 0.760, 0.730, 0.782, 0.972, respectivamente (promedio de 0.785). Por lo tanto, las mejoras al disminuir el número de AOI's no son lo suficientemente significativas.

6.3.5. Etapa 2: Asignación de estado mental.

En esta sección se presenta los métodos empleados para el tratamiento de señales fisio-psicológicas, la extracción de características y los resultados obtenidos para la clusterización de los estados mentales según lo descrito en la sección 5.3.1.

Preprocesamiento de señales fisio-psicológicas

Dado que las señales utilizadas corresponden a registros de comportamientos biológicos de cada usuario, estas deben limpiarse de ruidos e interferencias, generadas por la electrostática de los aparatos o por el movimiento de los músculos. A continuación se detalla el proceso empleado para la limpieza de cada una de las señales.

Es importante considerar que el modelo contempla el pronostico en tiempo real, por lo cual, se considerará que luego de la predicción de cada ventana temporal, se registrarán los datos reales de dicha ventana. Estos deben ser almacenados, con lo cual al final de la navegación se tendrá un registro completo, que ha sido capturado de manera "discreta". A continuación se presenta los pre-procesamientos realizados considerando este factor, que en general corresponde a preprocesar los datos adquiridos en la ventana de tiempo correspondiente o en la unión de esta a una cierta cantidad de ventanas anteriores.

Ajuste de señales por línea base.

Antes de generar variables independientes a partir de las señales psico-fisiológicas, se debe ajustar cada una de ellas por línea base. Esto es necesario para la comparación de señales entre sujetos. A modo de ejemplo, en la sección 3.2 se explica que los valores de la señal GSR puede variar entre usuarios. Lo mismo ocurre con el resto de las señales, por lo que la ley de valor inicial juega un papel importante [54].

En [99] realizan el ajuste de la señal de tamaño pupilar a partir del área de la pupila aproximada como una elipse, registrado por un periodo de 500 ms. anteriores a la presentación del estímulo. Posteriormente esta línea base es sustraída de las áreas pupilo-gráficas registradas durante la experimentación.

En [53] la sustracción de la línea base se aplica no sólo a los datos de seguimiento ocular, sino también a todas las demás señales, utilizando la base registrada 500 ms. antes de la presentación del estímulo.

Siguiendo esta última idea, en este trabajo se utilizan los datos de las diferentes señales registrados en un periodo de 500 ms anteriores a la presentación del estímulo (página web). Para calcular el área de la pupila se asume esta como una circunferencia y esta área es sustraída de la señal pupilo-gráfica registrada durante la navegación web.

Para el resto de las señales se utiliza la línea base registrada 500 ms. antes y se sustrae el promedio a las señales registradas durante el experimento. De esta manera, la comparación de señales entre usuarios presenta mayor énfasis en las variaciones registradas, más que en los valores adquiridos, permitiendo una mejor comparación entre usuarios.

Como la línea base se registra en el periodo anterior a la exposición del usuario al estímulo, este es un valor fijo durante toda la navegación para cada una de las señales. Por lo tanto, cada vez que se registren las señales en una nueva ventana temporal, estas deberán ser ajustadas en dicho periodo a partir de la línea base.

Electrocardiograma (ECG).

Los datos de ECG se adquieren con una frecuencia de 1000 Hz. Para una ventana de tiempo se ajustan los datos por línea base, generando una señal de variaciones de ECG (respecto a la línea base). Posteriormente se aplica un filtro pasa-banda entre las frecuencias 5-45 Hz.

A pesar que se puede procesar para calcular la señal de ritmo cardiaco, se omite este procesamiento ya que el ECG es un método mucho más invasivo para el usuario prefiriendo usar la señal de PPG.

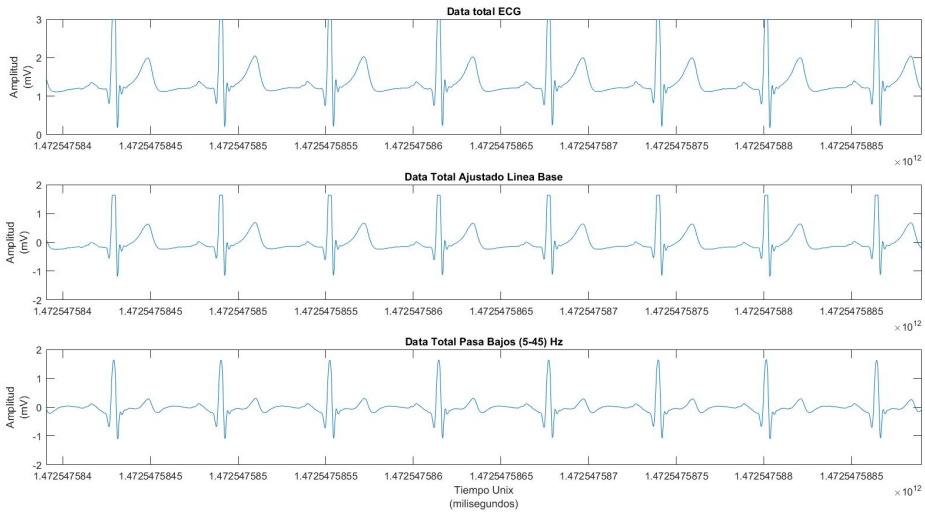


Figura 6.7: Procesamiento de la señal ECG en alguna ventana de tiempo.

Electroencefalografía (EEG).

Los datos del encefalograma son registrados con una frecuencia de muestreo de 128 Hz, para cada uno de los electrodos. Esta señal está sujeta a una gran variedad de artefactos y ruido, por ejemplo, los parpadeos, actividad oculomotora, los movimientos de la cabeza, las expresiones faciales que agregan ruido por la señal eléctrica muscular, movimiento de los electrodos, entre otros.

En primer lugar, la señal registrada en cada uno de los electrodos es ajustada por línea base del usuario. Así, se genera la señal de variaciones EEG.

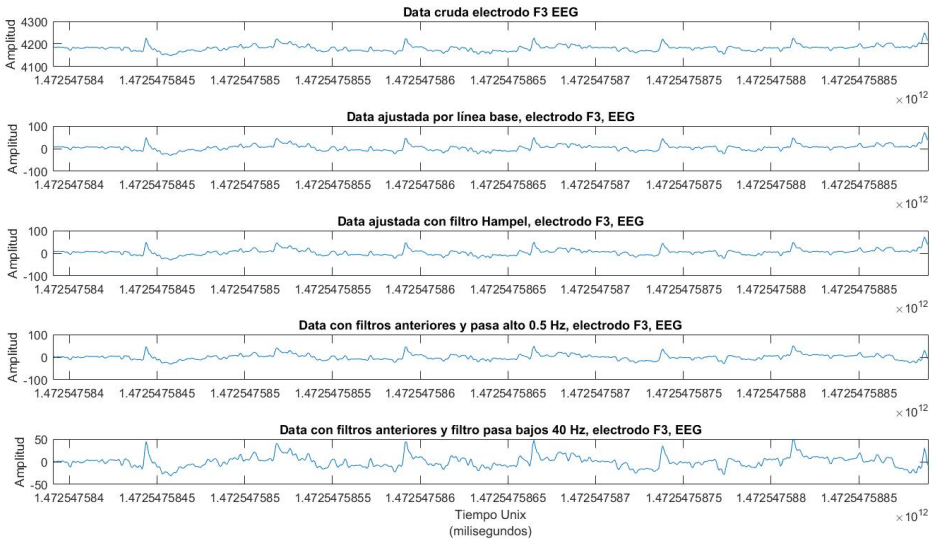


Figura 6.8: Procesamiento de la señal EEG en alguna ventana de tiempo, para el electrodo F3.

Para eliminar outliers y disminuir el efecto del artefacto parpadeo se usa un filtro Hampel [89]. Posteriormente, para eliminar el efecto del balanceo de la cabeza se utiliza un filtro pasa alto de frecuencia de corte de 0.5 Hz. Además, se usa un filtro pasa bajos de frecuencia de corte de 40 Hz, para elimina ruido de la red eléctrica (50 - 60 Hz). En la ilustración 6.8 se presenta un ejemplo para el electrodo F3 de alguna ventana de tiempo.

Con la señal preprocesada, se extraen las bandas Alpha y Theta aplicando un filtro pasa-banda de frecuencias (8-12) Hz. y (4-8) Hz. respectivamente. En la ilustración 6.9 se presenta un ejemplo para el electrodo F3 en una ventana de tiempo.

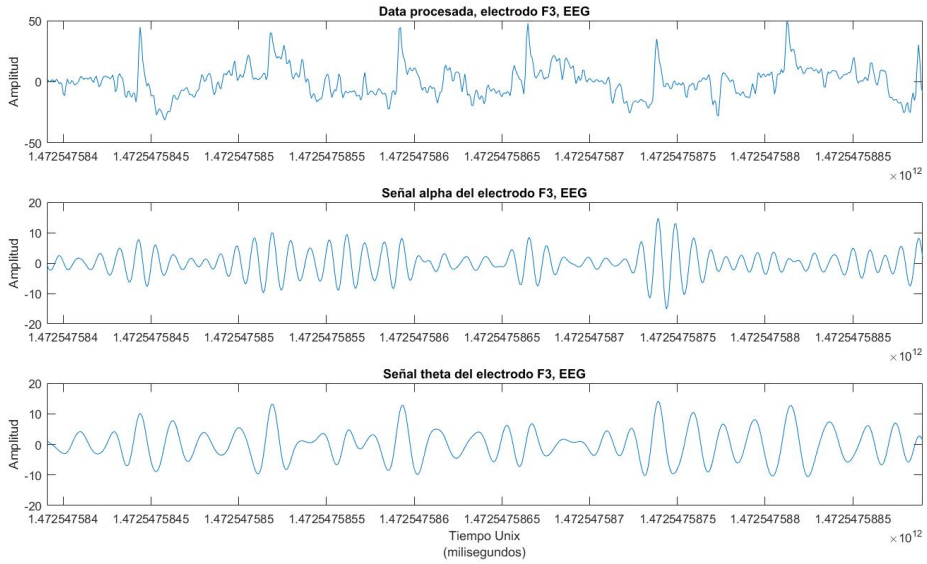


Figura 6.9: Bandas Alpha y Theta obtenidas del electrodo F3, para alguna ventana de tiempo.

Temperatura de la piel (ST).

La datos crudos registran valores de temperatura corporal en grados Celsius. Estos son almacenados a una frecuencia de muestreo de 50 Hz. Para procesar la señal de una nueva ventana de tiempo se utiliza un filtro pasa bajos de frecuencia de corte de 1 Hz. Posteriormente se realiza el ajuste por línea base generando una señal de "variaciones en la temperatura" (respecto a la línea base del usuario). En la figura 6.10 se observa el tratamiento realizado en una ventana de tiempo.

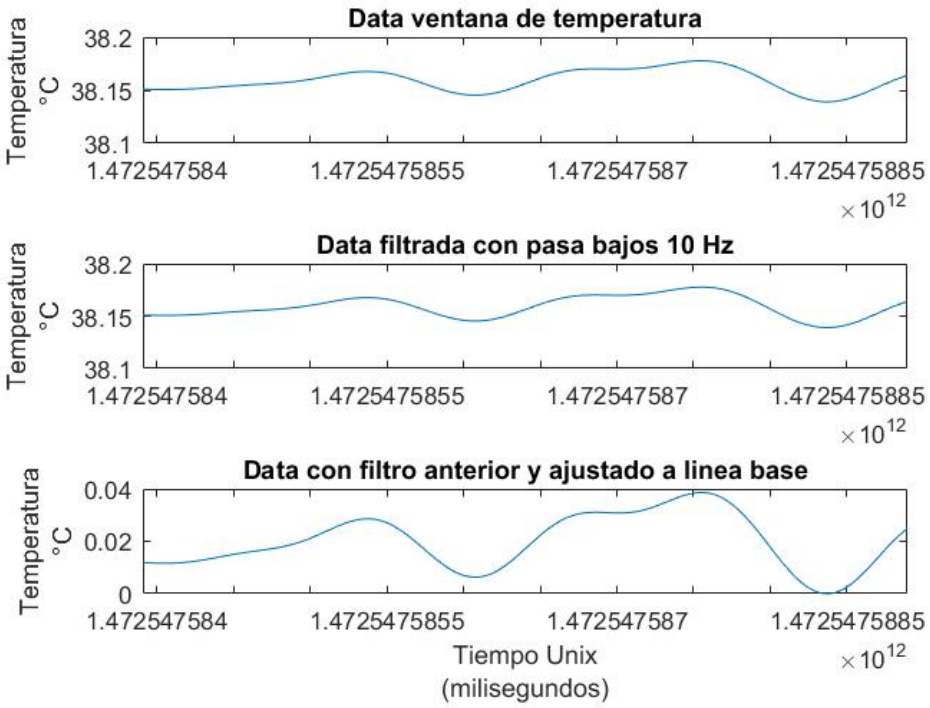


Figura 6.10: Procesamiento de la señal ST en alguna ventana de tiempo.

Fotopletismograma (PPG).

Esta señal se obtiene con un frecuencia de muestreo de 125 Hz. Esta es preprocesada para la generación de características y, de manera separada, para la obtención de la señal de ritmo cardiaco (HR).

En primer lugar, la señal registrada para cada ventana de tiempo es suavizada con una ventana de Hanning. Esta señal presenta pérdida de datos en los extremos de la señal de 0.04 segundos, para la primera ventana de tiempo registrada por un usuario. En las siguientes ventanas se evita esta pérdida de datos utilizando los datos de la ventana anterior al momento de hacer la suavización, truncando al periodo de tiempo correspondiente a la ventana.

Posteriormente, se ajusta la señal por linea base del usuario, generando la señal de variaciones de PPG (respecto a la línea base). En la figura 6.11 se muestra el procesamiento de la señal PPG.

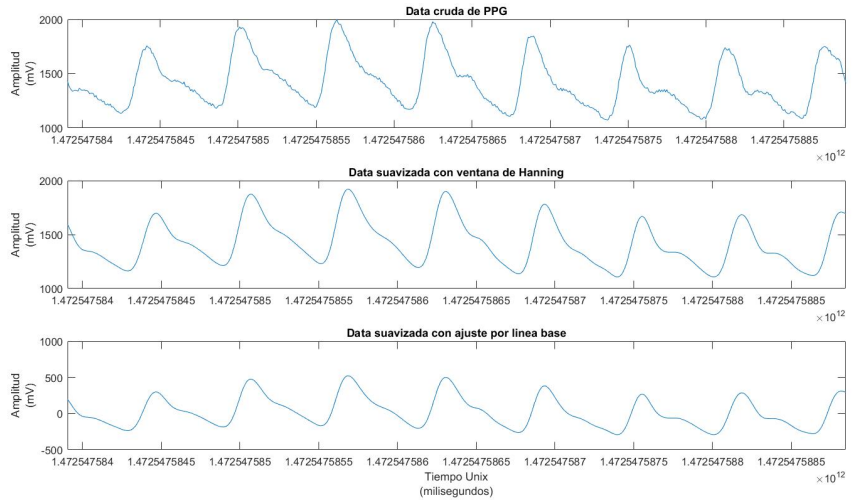


Figura 6.11: Procesamiento de la señal PPG en alguna ventana de tiempo.

Por otro lado, se genera un procesamiento diferente para la obtención de la señal HR. Para ello, se procesa la señal cruda de PPG con un filtro pasa-bajos con frecuencia de corte de 16 Hz y ventana de Blackman. Luego, se encuentran los máximos de la señal y se resta el tiempo entre ellos Δt. Posteriormente se convierten las unidades a [pulsaciones/minuto] siguiendo la fórmula:

$$bpm = \frac{60 \cdot 1000}{\Delta t} \left[\frac{pulsos}{minuto} \right]$$

Estos valores de pulsaciones por minuto son asignados a los tiempos en los que ocurren los peaks finales. Para el caso del primer peak detectado, es omitido en la primera ventana de tiempo y calculado para las demás ventanas a partir de los datos de ventanas anteriores.

Finalmente, con los datos de bpm obtenidos y asignados, se realiza una interpolación lineal cúbica, en el espacio de tiempo de la ventana, donde el primer valor corresponde al promedio de los valores bpm obtenidos. En la imagen 6.12 se muestra el procesamiento realizado para la obtención de la señal HR.

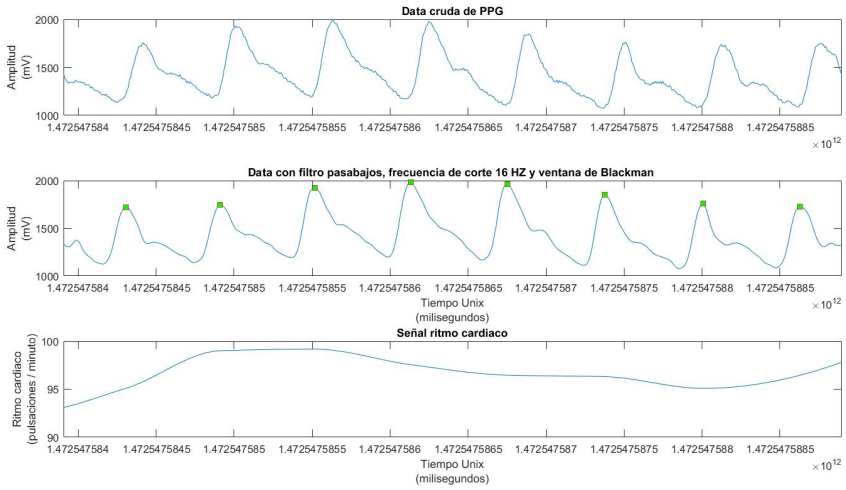


Figura 6.12: Procesamiento de la señal PPG para la obtención de la señal HR en alguna ventana de tiempo.

Seguimiento ocular (EyeTracker).

En el seguimiento ocular se registra el tamaño pupilar de cada usuario, en cada ojo, a una frecuencia de muestreo de 125 Hz. En ocasiones ocurren eventos tales como el pestañeo o el movimiento del usuario (que en este caso se encuentra controlado), donde no se logra detectar la pupila.

El pestañeo es a menudo un movimiento involuntario en el cual se cierra y abre el párpado. Durante cada pestañeo el párpado bloquea la pupila y la córnea, por lo que el EyeTracker pierde la capacidad de registro por parte del iluminador, resultando en datos perdidos para las coordenadas (x, y) de la mirada. Esto también ocurre cuando el participante desvía la mirada de la pantalla, relacionándose con los movimientos de la cabeza. Sin embargo, al inicio del experimento se le pide a los usuarios permanecen lo más estático posible.

Para la detección de pestañeos en los datos de tamaño pupilar se utiliza como base la programación realizada por Greg Siegle [95], adaptándolo para su uso en el presente problema. El script final genera un arreglo de mismo tamaño que los datos de tamaño pupilar entregados. Para cada muestra de datos, este arreglo posee los valores de 1 en caso de que se identifique en dicha posición la presencia de un pestañeo y 0 en caso contrario.

Los valores perdidos son reemplazados por cero. Así, pueden ser reconocidos, mediante la identificación de cambios bruscos en el tamaño pupilar. Se determinan artefactos marcando los cambios mayores a 0.5 mm en el tamaño pupilar que ocurren en tan solo una muestra (8 milisegundos).

Se establecen los límites superior e inferior para el tamaño pupilar, basado en los tamaños normales. Por lo tanto, se consideran artefactos aquellos datos en los cuales el tamaño pupilar está fuera del rango $[1{,}5, 5]$ mm. Sin embargo, analizando los datos presentes, estos se encuentran dentro de los rangos normales presentando un tamaño mínimo de 2,13 y máximo

de 4,12, pero para futuras implementaciones se debería considerar.

En tercer lugar, se marcan los cambios fuera del intervalos interquartil de Tukey, para evitar cambios lejanos a la desviación estándar. Finalmente los pestañeos y otros artefactos identificados son tratados con interpolación polinómica de Hermite de grado 3.

La metodología anterior es aplicada a los datos registrados en ambos ojos. Las nuevas señales con valores interpolados son suavizadas aplicando un filtro de Hampel.

Además de los datos de tamaño pupilar se tienen los indicadores de confiabilidad para estas muestras, con valores $\{1, \ldots, 4\}$, donde un menor número implica mayor confiabilidad.

Las nuevas señales son promediadas basándose en los indicadores de confiabilidad. Se genera un nuevo vector de datos con un tamaño igual al número de muestras registradas para el tamaño pupilar y en cada entrada se agrega el valor de la pupila (izquierda o derecha) con mejor indicador. En caso que ambas señales sean igual de confiables se agrega el valor del promedio aritmético entre ambas señales.

Este procesamiento es realizado para los datos de la última ventana de tiempo registrada en conjunto a los datos de la nueva ventana de datos adquirida. Así, los datos correspondientes a la nueva ventana son almacenados.

Posteriormente los valores son ajustados por la línea base, generando la señal de variación de diámetro pupilar (respecto a la línea base).

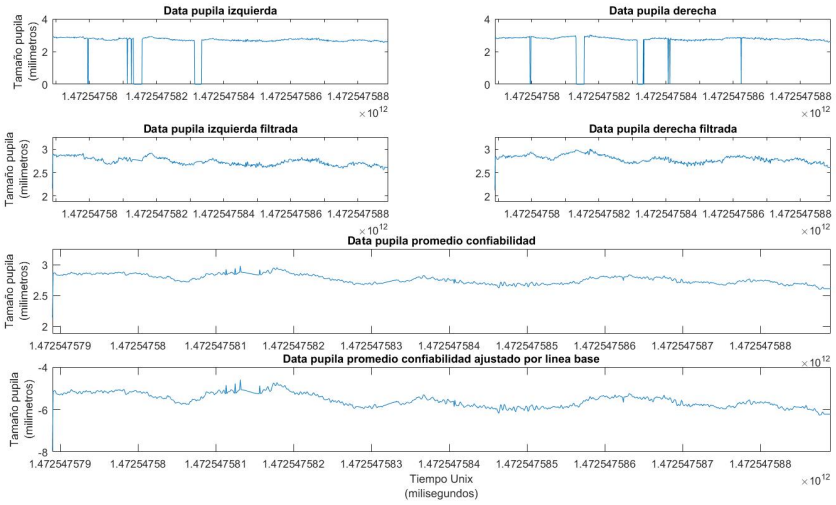


Figura 6.13: Procesamiento de diámetro pupilar en alguna ventana de tiempo.

Respuesta electro-dérmica (GSR)

La frecuencia de muestreo de esta señal es de 125 Hz. En primer lugar, para cada ventana de tiempo registrada, se aplica un filtro pasa-bajos tipo Butterworth con frecuencia de corte de 5 Hz. Posteriormente se reduce la tasa de muestreo de la señal cruda para aliviar la complejidad computacional, disminuyendo de 125 Hz a 10 Hz, sin pérdida de resolución [49].

Luego, se aplica una ventana de Hanning para suavizar la señal.

Al suavizar la señal con ventana de Hanning se produce pérdida de datos en los extremos de la señal. Dado el número de muestras especificada en la ventana de tiempo, se genera una perdida de 0.5 segundos al comienzo y al final de la señal GSR en la primera ventana de tiempo. Para las siguientes ventanas, se consideran los datos ya registrados en la ventana anterior para evitar la pérdida de datos en su comienzo.

Finalmente, la señal es ajustada según el valor de la línea base guardada para el usuario, generando una "señal de variaciones GSR" (respecto a la linea base del usuario). En la figura 6.14 se muestra el pre-procesamiento de la señal en la primera ventana de tiempo. Se observa la pérdida de datos al comienzo y al final de la señal luego de suavizar con ventana de Hanning.

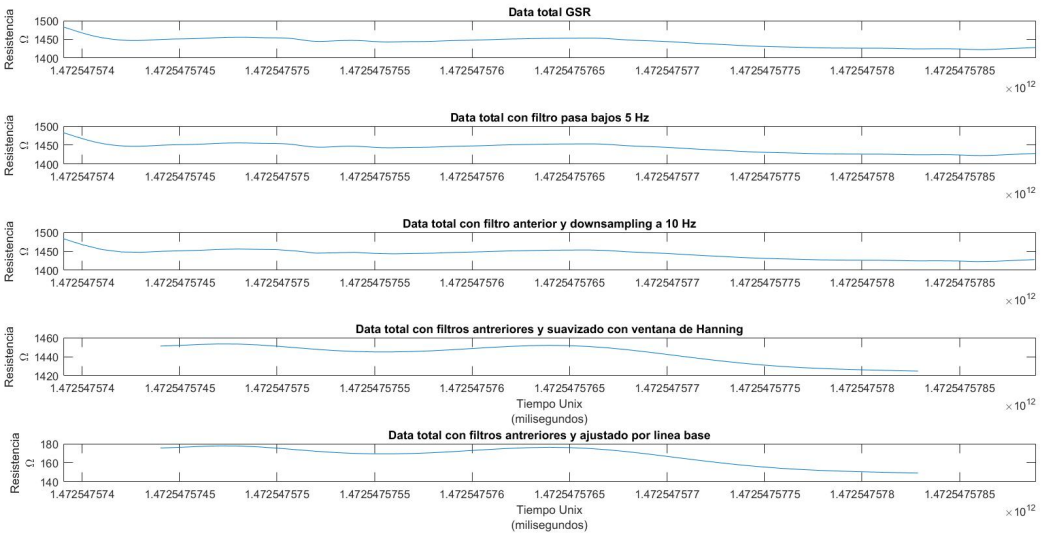


Figura 6.14: Procesamiento de la señal GSR en la primera ventana de tiempo.

Tal como se explica en la sección 3.2, la señal GSR posee dos componentes, la fásica y la tónica. Para la separación de la señal GSR pre-procesada en dichas componentes se utiliza el método de optimización convexa creado en [42] con su implementación en Matlab, ajustando los parámetros de frecuencia de muestreo con la cual se capturan estos datos.

De la señal ya preprocesada se extraerán características relacionadas a la señal de variaciones GSR y a su componente fásica. Para estimar esta última, se considera una aproximación de la señal completa como últimos 20 segundos de navegación, es decir, las últimas tres ventanas temporales registradas y la ventana para la cual se quiere calcular la componente. A este conjunto de datos se le calcula la señal fásica, considerando los últimos datos correspondientes a la ventana de tiempo a estudiar.

El método anterior es realizado ya que la extracción de la componente fásica está afectada por la tendencia de la señal a través del tiempo. Por otro lado, la utilización de las señal

completa ralentiza el procesamiento de la señal fásica.

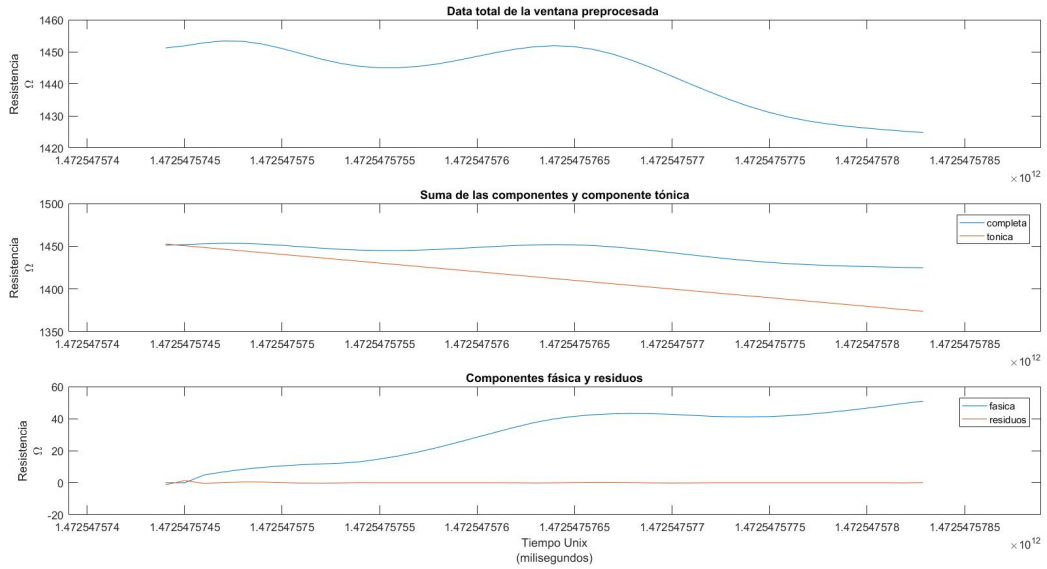


Figura 6.15: Componentes de la señal GSR en la primera ventana de tiempo.

Extracción de características

Como se explicó anteriormente, se busca caracterizar a cada usuario a partir de un estado mental, motivados por los posibles cambios en la forma de las sacadas realizadas, según el estado mental en el cual se encuentre el usuario.

Los movimientos de las sacadas varían en amplitud, duración y velocidad (peak), y la relación entre ellas se denomina "secuencia principal" para indicar que la velocidad peak (PV) y la duración aumentan con la amplitud [9].

En la literatura se establece que existe relación entre la velocidad peak de las sacadas realizadas por una persona y el nivel de carga cognitiva en el que se encuentra al momento de realizarlas. Por ejemplo, en [17] muestran que la media de las velocidades picos de las sacadas aumentan cuando la carga de trabajo aumenta. En [72, 34] respaldan este hallazgo.

A pesar de lo anterior, es importante considerar lo expuesto en [7], donde se explica que la velocidad de las sacadas no puede ser usada como indicador de algún estado mental debido a su fuerte dependencia con la amplitud de las sacadas. En la actualidad lo anterior ha sido corregido [35], sin embargo, aún es posible la utilización de carga cognitiva en relación a la amplitud de sacadas en ciertos ambientes.

En [2] se investiga el efecto en las operaciones del controlador de tránsito aéreo y la carga de trabajo del uso de pantallas meteorológicas. Dentro de sus resultados encuentran una disminución en la distancia de las sacadas al incrementar el número de aeronaves. Sin embargo, al entrar en detalle observan que dicha reducción se debe a causas relativas a la carga cognitiva y no a la reducción entre la distancia de las aeronaves.

Por otro lado, en [24], al investigar la influencia conjunta de las demandas cognitivas y movimientos sacádicos de gran amplitud en la frecuencia de parpadeo espontánea para un cierto ambiente, a diferentes niveles de complejidad, se observa que el número de cambios de la mirada de gran amplitud aumenta con el nivel de complejidad. Dado que en general los niveles de dificultad se relacionan con los niveles de carga cognitiva se puede considerar que la amplitud de las sacadas podrían tener cambios en relación a dichos niveles.

En el presente trabajo se mantiene la hipótesis que la amplitud de las sacadas se relaciona a los niveles de carga cognitiva y por lo tanto, se utilizarán diferentes características relacionadas a la estimación de carga cognitiva para crear los diferentes estados mentales, sin que necesariamente estos últimos correspondas a dichos niveles. En total se generan X variables a partir de todas las señales. Se eliminan Y de YY ventanas temporales debido a valores perdidos, las cuales no se les calculará los "caminos visuales" en la última etapa.

En cada ventana de tiempo se utilizan las señales fisio-psicológicas registradas en el tiempo anterior al comienzo de la ventana a predecir. Sin embargo, debido a la necesidad de rapidez en el computo (el modelo debería correr antes del termino de la ventana), los datos considerados serán limitados a diferentes periodos de tiempo dependiendo de la señal considerada.

Normalización de las señales preprocesadas

Para lograr una comparación entre las señales de los usuarios se ha agregado el ajuste por línea base, y además, cada una de las señales es normalizada restando la media de los datos registrados y dividiendo por la desviación estándar. Como los datos son adquiridos por ventanas de tiempo, para cada señal, se extraen la desviación estándar y la media de los datos adquiridos hasta el momento previo al inicio de la ventana a predecir, lo cual es usado al conjunto de datos utilizado para extraer cada característica.

Electrocardiograma (ECG)

Basado en estudios anteriores [53, 44] se extraen la media y la mediana de los valores de la señal de variaciones en el ECG en las dos ventanas anteriores a la predicción (10 segundos precedentes). Además se utiliza la varianza de la ECG MAD (desviación absoluta media), en dicho periodo de tiempo, calculada como:

$$ECGMAD = mediana(|ECG_{\mathrm{i}} - mediana(ECG)|)$$

La desviación absoluta media es una medida de la dispersión estadística. Además, el MAD es una estadística robusta, siendo más resistente a los valores atípicos en un conjunto de datos que la desviación estándar.

Electroencefalografía (EEG)

El EEG se analiza en base a tiempo-frecuencia, debido a su relación con el proceso psicofisiológico y estructural del cerebro. Además, este enfoque es usado para estudio de estados emocional-cognitivo, siendo recomendado al estudiar una cantidad de datos baja (periodos cortos de tiempo) [48], como en este caso.

En la literatura se establece que las bandas Alpha y Theta se relacionan a los procesos cognitivos [48]. Al mismo tiempo, la carga cognitiva se relaciona a los lóbulos parietal y occipital [37, 6].

Por lo tanto, se utilizan las señales registradas en los electrodos $\{F3, F4, F7, F8, P7, P8\}$, a los cuales se le aplica un filtro pasa-banda de (8-12) Hz y (4-8) Hz para obtener las bandas Alpha y Theta respectivamente.

Posteriormente se utiliza la transformada de Hilbert ($\mathcal{H}\{\text{ee}g\}$), la cual permite el análisis en tiempo-frecuencia, y esta dada por la siguiente fórmula:

$$\mathcal{H}\{\text{ee}g\} = (h * \text{ee}g)(t) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{\text{ee}g(\tau)}{t - \tau} \text{d}\tau$$

Finalmente, de estas señales son extraídas la potencia y la fase de cada electrodo considerado.

Temperatura de la piel (ST)

Para hacer más comparable las variables obtenidas para cada usuario, se normaliza la señal al intervalo $[0, 1]$, utilizando el método de escalamiento de características, en la señal de variaciones de temperatura de cada ventana de tiempo. Esto es, para cada muestra de la señal, se aplica la fórmula:

$$X = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Basado en los estudios [44, 70], se extraen la media y la mediana de la señal de temperatura corporal.

Fotopletismograma (PPG)

En primer lugar, se utiliza la señal de variación de PPG. Para hacer métricas comparables entre usuarios, esta señal es normalizada a la escala $[0, 1]$ usando el método de escalamiento de características. Así, se detectan los peaks de la señal (no así los valles), lo que permite explicar los máximos cambios de la señal en relación a los valores normales de navegación. De esta forma se construyen las siguientes variables.

- mu_PPG y $st\text{d}\_PPG$: Media y desviación estándar de la señal de variaciones PPG normalizada, dentro de la ventana de tiempo.
- muP_PPG y $st\text{d}P\_PPG$: Media y desviación estándar de la amplitud de los peaks detectados en la señal de variaciones de PPG normalizada dentro de la ventana de tiempo.
- AM_PPG: Amplitud máxima de la señal variaciones de PPG normalizada dentro de la ventana de tiempo.
- TM_PPG: Fracción de tiempo necesario para alcanzar el máximo respeto al tiempo total de la ventana. T_{max} es el tiempo en el que se produce el máximo en la ventana de

tiempo $[t_o, t_f]$

$$TM\_PPG = \frac{T_{max} - t_f}{t_f - t_o}$$

Ritmo cardiaco (HR)

Para hacer comparable las variables entre diferentes usuarios, la señal HR dentro de una ventana de tiempo es normalizada usando el método de escalamiento de características.

Basados en los estudios [44, 81, 14] se calcula el promedio, la desviación estándar y el valor cuadrático medio (RMS) de la señal HR dentro de la ventana de tiempo a considerar.

Seguimiento ocular (EyeTracker)

En primer lugar se utiliza los datos ajustados por la linea base para calcular las características $mean\_PupilVar$ y $std\_PupilVar$, que indican, para cada ventana de tiempo, el promedio y la desviación estándar de las variaciones del área pupilar respecto al área registrada como línea base.

Posteriormente se utiliza la señal registrada del tamaño pupilar, según grados de confiabilidad, normalizada para una mejor comparación entre usuarios. Esta presenta menor continuidad que la señal principal registrada. Al ser un muestreo en tiempo discreto, se decide usar las diferencias entre muestras para calcular dos nuevas señales, denominadas velocidad y aceleración de tamaño pupilar respectivamente, mediante las fórmulas:

$$Vel(\mathrm{i}) = \frac{MeanPupil(\mathrm{i}+1) - MeanPupil(\mathrm{i})}{tiempo(\mathrm{i}+1) - tiempo(\mathrm{i})} \qquad Acc(\mathrm{i}) = \frac{Vel(\mathrm{i}+1) - Vel(\mathrm{i})}{tiempo(\mathrm{i}+1) - tiempo(\mathrm{i})}$$

De esta manera, se extraen las siguientes características:

- $range\_Pupil$: Diferencia entre el valor máximo y mínimo del tamaño pupilar registrado en cada ventana de tiempo.
- vel_Pupil: Corresponde al promedio de los datos de la señal velocidad dentro de una ventana de tiempo.
- acc_Pupil: Corresponde al promedio de los datos de la señal aceleración dentro de una ventana de tiempo.

Además de características relativas al tamaño pupilar, se utilizan los datos de fijaciones y pestañeos realizados durante la navegación. Las características relacionadas a las sacadas que pueden ser encontradas en la literatura, no son usadas ya que se espera que la caracterización de estados mentales permita diferenciar entre las propiedades de estos movimientos.

Para las fijaciones se extraen las siguientes características:

- Fix_N: Número de fijaciones registradas durante la última ventana de tiempo de registrada.
- $Fix\_mean\_PupilVar$ y $Fix\_std\_PupilVar$: promedio y desviación estándar de la variación entre el área calculada con el promedio del tamaño pupilar, según indicadores

de confiabilidad dentro de todas las fijaciones registradas en la última ventana de tiempo registrada, y el área calculada con el registro de línea base .

- Fix_mean_T y Fix_std_T: Promedio y desviación estándar del tiempo de duración de las fijaciones registradas en la última ventana de tiempo.

Para las variables de pestañeo se consideran las señales de cada ojo, ya que estos actúan de manera independiente, calculando así las siguientes características:

- $Blink\_N$: Número de pestañeos detectados en la ultima ventana de tiempo registrada. Se calcula el promedio entre el número de pestañeos detectados en cada ojo.
- $Blink\_mean\_T$ y $Blink\_std\_T$: Promedio y desviación estándar de las duraciones de los pestañeos registrados en ambos ojos durante la última ventana de tiempo.

Respuesta electro-dérmica (GSR)

Siguiendo lo realizado en [80] respecto a la normalización de la señal GSR, pero de forma adaptada al problema aquí tratado, donde cada instancia de un mismo usuario es considerado como un usuario distinto, se generan variables que permitan conocer el comportamiento en la señal de variaciones GSR, presente en una ventana de tiempo, en comparación al registro completo de la navegación.

Primero, se aproxima la señal de variación GSR total a los últimos 20 segundos de navegación, anteriores a la próxima ventana de tiempo (objetivo de predicción). Se normalizan los datos de la siguiente forma, donde i representa el usuario, t el tiempo en que se toma dicha muestra y T el tiempo total de registro:

$$Normalized\_GSR(\mathrm{i}, t) = \frac{GSR(\mathrm{i}, t)}{\sum_{t=1}^{T} GSR(\mathrm{i}, t)}$$

De esta forma se generan las siguientes variables:

- GSR_ACC: Datos normalizados acumulados en la ventana de tiempo $[t_o, t_f]$. Esta variable representará valores mayores en las ventanas de tiempo que presenten mayores valores de la señal de variaciones GSR.

$$GSR\_ACC = \sum_{t=t_o}^{t_f} Normalized\_GSR(\mathrm{i}, t)$$

- GSR_PROMT: Promedio en función del tiempo (segundos) de los datos normalizados acumulados en la ventana de tiempo actual.

$$GSR\_PROMT = \frac{GSR\_ACC}{t_f - t_o}$$

Los valores de estas variables presentan el problema de que en los primeros 15 segundos no se posee la información necesaria, por lo cual los valores serán mayores a los de las siguientes ventanas. Esto se debe a que al comenzar no se tiene conocimiento acerca del usuario, siendo complejo identificarlo en algún estado mental. De hecho, en la primera ventana

de tiempo estas variables tomarán el mismo valor para cualquier participante. Posteriormente se estabilizarán en valores que permitan una mejor identificación. Lo mismo ocurre con las variables de potencia espectral definidas a continuación.

La potencia espectral $P(\omega)$ se calcula de la siguiente forma, donde ω es la frecuencia angular, $Y(\omega)$ se calcula con la transformada de fourier en la señal, $Y(\omega^*)$ es el conjugado complejo y N la cantidad de datos registrados en los últimos 20 segundos:

$$P(\mathrm{i},\omega) = \frac{1}{N} Y(\mathrm{i},\omega) Y^*(\mathrm{i},\omega)$$

Así, para cada dato se realiza una normalización similar al caso anterior, donde Ω representa el total de muestras de frecuencias registradas durante toda la navegación del usuario. Con dicha normalización se generan dos nuevas variables.

$$Normalized\_P(\mathrm{i},\omega) = \frac{P(\mathrm{i},\omega)}{\sum_{\omega=1}^{\Omega} P(\mathrm{i},\omega)}$$

- GSR_PE_ACC: Potencia espectral normalizada de la señal de variaciones GSR, acumulada en la ventana de tiempo $[t_o, t_f]$, con frecuencias indexadas por $[\omega_o, \omega_f]$.

$$GSR\_PE\_ACC = \sum_{\omega=\omega_o}^{\omega_f} Normalized\_P(\mathrm{i},\omega)$$

- GSR_PE_MEAN: Potencia espectral normalizada promedio de la señal de variaciones GSR en la ventana de tiempo $[t_o, t_f]$ con frecuencias indexadas por $[\omega_o, \omega_f]$ y N_v el número de muestras presentes en la ventana de tiempo actual.

$$GSR\_PE\_MEAN = \frac{\sum_{\omega=\omega_o}^{\omega_f} Normalized\_P(\mathrm{i},\omega)}{N_v}$$

Posteriormente se generan características a partir de la componente fásica de esta señal. Para ello, dicha señal es normalizada al rango $[0, 1]$ usando el método de escalamiento de características, permitiendo una mejor comparación entre las métricas para diferentes usuarios.

Para la señal fásica normalizada se detectan los peaks, considerando que una señal monótona creciente o decreciente estará compuesta de un peak y un valle en cada uno de los extremos (o en orden inverso para las señales crecientes), para así evitar valores nulos dentro las variables. En la figura 6.16 se muestra la detección de peaks en la señal fásica de una ventana de tiempo.

Finalmente, se generan las siguientes características a partir de la señal fásica normalizada:

- n_peaks: Número de peaks detectados en la señal fásica dentro de la ventana de tiempo analizada.
- $avg\_amplitud$: promedio de la amplitud de los peaks detectados.
- $max\_amplitud$: Es el máximo valor absoluto de crecimiento (peak) o decrecimiento (valle) presente en la señal durante la ventana de tiempo analizada.

- $avg\_riseTime$: Tiempo promedio de aumento entre un valle y un peak, presentes en la señal fásica durante la ventana de tiempo analizada.
- $avg\_fallTime$: Tiempo promedio de disminución entre un peak y un valle, presentes en la señal fásica durante la ventana de tiempo analizada.
- $avg\_riseLevel$: Tiempo promedio de aumento de la señal, es decir, de los tiempos que demora la señal entre un valle y un peak, presente en la señal fásica durante la ventana de tiempo analizada.
- $avg\_fallLevel$: Tiempo promedio de decrecimiento de la señal, es decir, de los tiempos que demora la señal entre un peak y un valle, presente en la señal fásica durante la ventana de tiempo analizada.

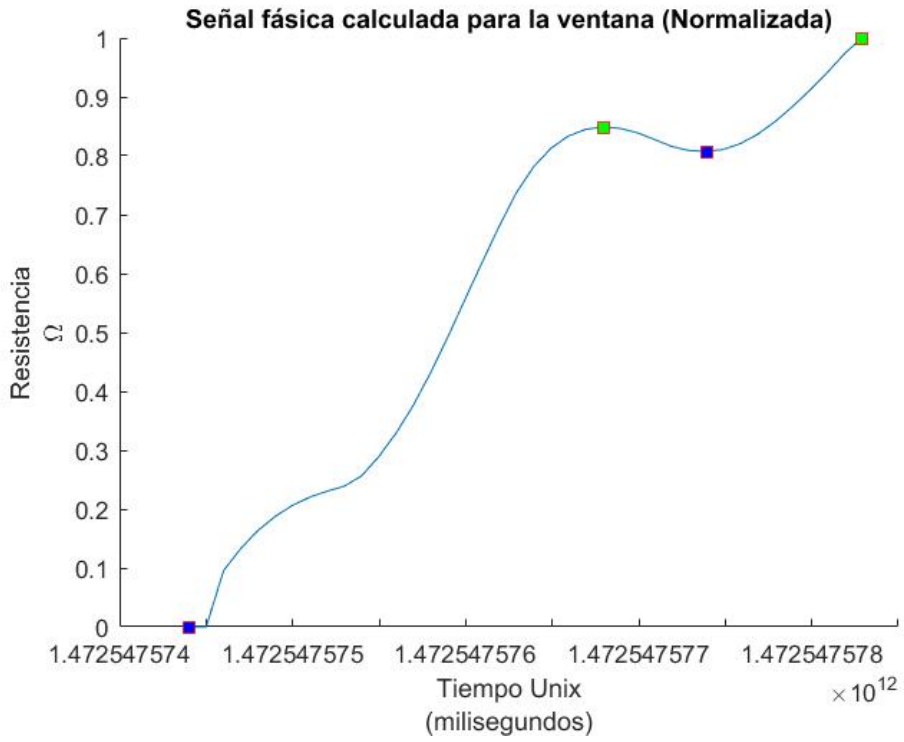


Figura 6.16: Ejemplo de la detección de peaks en la señal fásica de una ventana de tiempo

Metodología de clusterización

Para cada conjunto entrenamiento-prueba utilizado, se realiza una clusterización de las ventanas temporales de los registros del conjunto de entrenamiento. Posteriormente las ventanas del conjunto de prueba serán asignadas a los clusters almacenados.

Se utilizan los métodos de "Hard clustering", K-mean y K-medoid (véase 4.2), con eliminación de variables ("Backward selection"). En cada iteración es eliminada la variable que implique mayor mejora en las métricas de comparación. Así, se comparan los resultados al clusterizar los datos utilizando $\{2, \ldots, Nvar\}$ variables.

Se decide la utilización de algoritmos de selección de variables, ya que así se resuelven

dos problemas: por un lado se disminuye la complejidad computacional requerida para la asignación de nuevas ventanas de tiempo al cluster correspondiente. Al mismo tiempo, al disminuir la cantidad de variables, se disminuye la redundancia en la información, mejorando los resultados de partición.

Debido a que el método de eliminación iterativa de variables se debe realizar para cada fold, se decide utilizar las métricas SC, S y XB para comparar las mejoras alcanzadas al eliminar cada una de las variables. Esto debido a que, a diferencia de las métricas de DI y ADI, estos cálculos no son tan complejos computacionalmente. Sin embargo, dada la definición del índice de Xie y Beni, en el caso de Kmedoid siempre se tendrá $XB = \infty$, utilizándose sólo las métricas SC y S.

El algoritmo 1 se ejecuta para generar la eliminación recursiva de variables, entregando como resultado las variables eliminadas en cada iteración. Como se hace uso de métodos de hard clustering, la elección del modelo final se realiza con la comparación de métricas DI y ADI, que resulta mejor para estos casos, pero que es computacionalmente costosa para su utilización directa en las iteraciones de la eliminación iterativa de variables.

Algorithm 1: Backward feature selection en clusters

Function *BSClusters (K, method, paramCluster, dataCluster, c, rutaGuardado)*

$VarEliminar \leftarrow \{\ \}$; ▷ `Almacenar variables a eliminar en cada iteración`

$Nvar \leftarrow$ número de variables totales;

dataCluster $\leftarrow$ Reemplazar Missing Values por cero;

for $Nelim = 0 : (Nvar - 2)$ **do** ▷ `Eliminación iterativa de variables`

dataClusterAux $\leftarrow$ EliminarVar(dataCluster,VarEliminar);

$SC = \infty,\ S = \infty,\ XB = \infty$;

if $Nelim = 0$ **then**

GuardarResultados (ClusterKfold (K, method, paramCluster, dataClusterAux), rutaGuardado);

else

for $V_{(\mathrm{i})} = V_{(1)}, \ldots, V_{(Nvar-Nelim)}$ **do** ▷ `Escoger posibles variables`

dataAux2 $\leftarrow$ EliminarVar(dataClusterAux,$V_{(\mathrm{i})}$);

$[SC_{Nelim}, S_{Nelim}, XB_{Nelim}] \leftarrow$ ClusterKfold (K, method, paramCluster, dataAux2);

if $(SC_{\mathrm{i}} \leq SC) \wedge (S_{\mathrm{i}} \leq S) \wedge (XB_{\mathrm{i}} \leq XB)$ **then** ▷ `Mejor variable a eliminar`

$VarToEliminar \leftarrow V_{(\mathrm{i})}$;

$SC = SC_{\mathrm{i}},\ S = S_{\mathrm{i}},\ XB = XB_{\mathrm{i}}$;

$VarEliminar \leftarrow VarEliminar.Add(VarToEliminar)$;

GuardarResultados (ClusterKfold (K, method, paramCluster, dataClusterAux), rutaGuardado);

Resultados clusterización

Para comparar resultados se utilizan las métricas explicadas en la sección 4.2.3 (DI y ADI), que resultan mejores indicadores para la elección de cluster con métodos de hard clustering en conjunto con las métricas calculadas en la eliminación iterativa (SC, S, XB).

Se comparan resultados obtenidos con $\{2, \ldots, 5\}$ clusters, ya que un mayor o menor número de clusters implicaría una cantidad de estados mentales que hace intratable las siguientes etapas.

Para cada número de clusters, se prueban diferentes configuraciones (número de variables). A cada configuración se le asigna un "puntaje" proporcional a los valores de DI y ADI (ya que valores mayores son mejores), inversamente proporcional a SC, S y XB (valores menores son mejores) e inversamente proporcional al número de variables utilizadas (ya que se busca un menor procesamiento). Los pesos de estas métricas a los puntajes se realiza tal que DI y ADI presenten una mayor importancia que las otras métricas y permitiendo visualizar resultados de acuerdo a lo que el investigador haría de manera manual. Así, el proceso es automatizado obteniendo los gráficos 6.5 y 6.6.

Tabla 6.5: Métricas K-Means diferente número de clusters

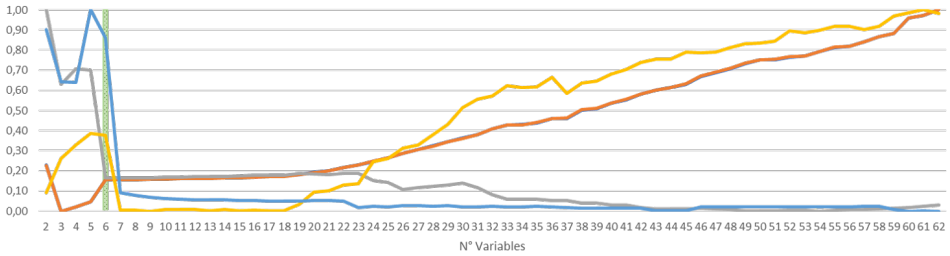
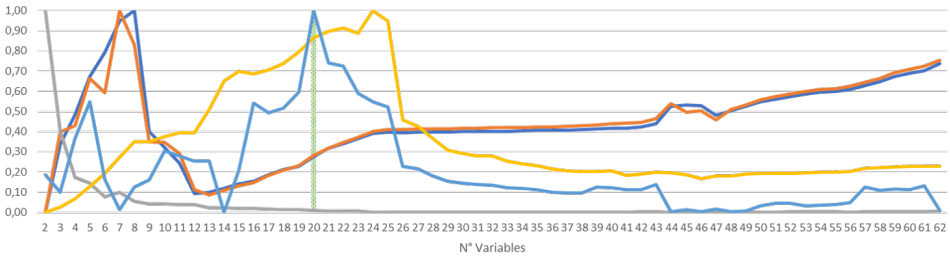
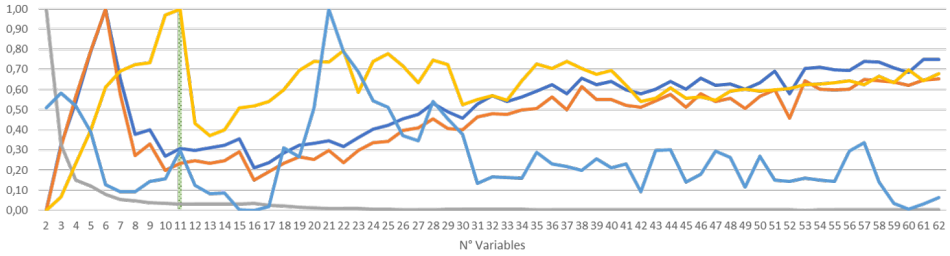
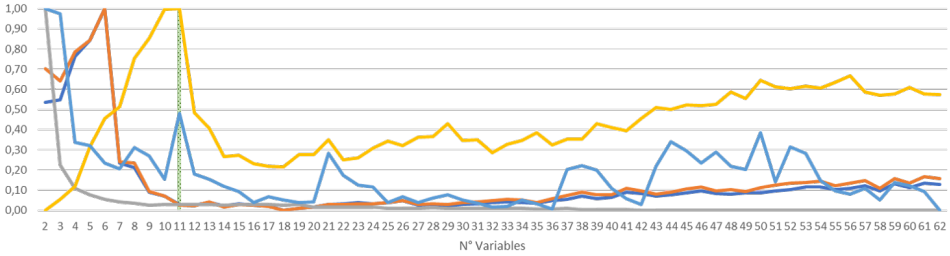
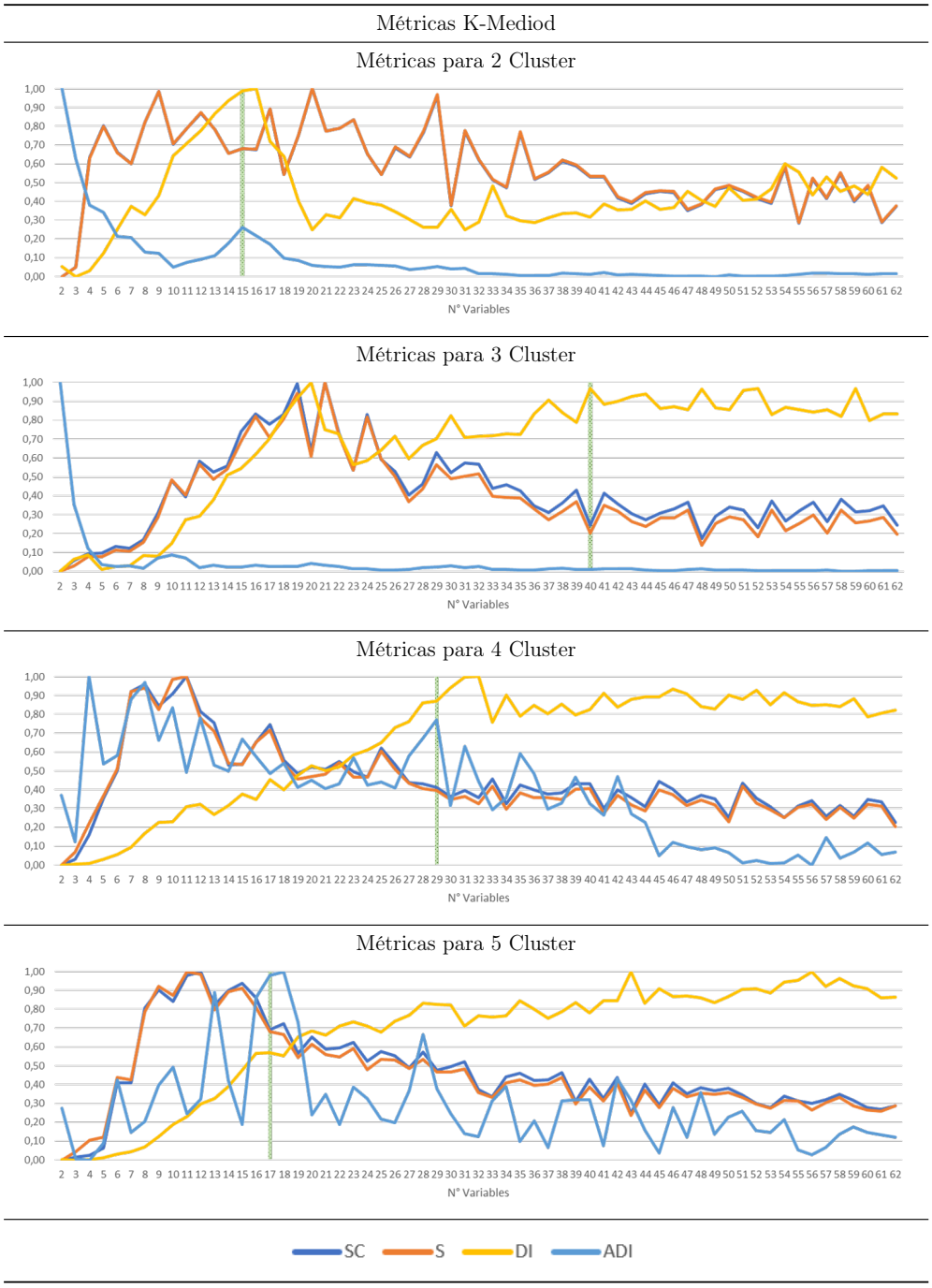
Métricas K-Means
Métricas para 2 Cluster

Métricas para 3 Cluster

Métricas para 4 Cluster

Métricas para 5 Cluster



Tabla 6.6: Métricas K-Mediod diferente número de clusters



En los gráficos anteriores, las líneas verdes corresponden al número de variables escogidos dado el mejor puntaje (véase sección 8.2 en anexos). De esta manera, los mejores resultados para cada número de cluster se presentan en las tablas 6.7 y 6.8 para los casos Kmean y Kmediod respectivamente. Se resaltan los cluster con mejor métrica DI en cada caso.

Tabla 6.7: Resultados clusterización K-Mean

Kmean						
N clusters	**N variables**	**SC (desv. est.)**	**S (desv. est.)**	**XB (desv. est.)**	**DI (desv. est.)**	**ADI (desv. est.)**
2	6	0,5570 (0,0131)	0,0003 (0,0000)	2,1805 (0,0186)	0,0195 (0,0018)	0,0565 (0,0199)
3	20	0,7601 (0,0056)	0,0006 (0,0000)	1,8844 (0,0182)	0,1748 (0,0138)	0,0023 (0,0004)
4	11	0,6406 (0,0282)	0,0004 (0,0000)	2,3431 (0,0791)	0,0674 (0,0136)	0,0002 (0,0001)
5	11	0,5660 (0,0307)	0,0004 (0,0000)	2,5323 (0,1307)	0,0770 (0,0185)	0,0003 (0,0001)

Tabla 6.8: Resultados clusterización K-Mediod

Kmediod						
N clusters	**N variables**	**SC (desv. est.)**	**S (desv. est.)**	**XB (desv. est.)**	**DI (desv. est.)**	**ADI (desv. est.)**
2	6	0,5603 (0,0612)	0,0003 (0,0000)	inf -	0,0948 (0,0130)	0,0751 (0,0172)
3	20	0,2328 (0,1307)	0,0001 (0,0001)	inf -	0,0546 (0,0081)	0,0031 (0,0017)
4	11	0,2787 (0,1041)	0,0002 (0,0001)	inf -	0,0497 (0,0036)	0,0035 (0,0035)
5	11	0,3649 (0,1760)	0,0002 (0,0001)	inf -	0,0311 (0,0054)	0,0029 (0,0060)

Finalmente se escoge la utilización del modelo Kmean con tres clusters y 20 variables, listadas en la tabla 6.9. Se utilizan las señales de GSR (fásica), ECG, PPG, HR, EEG, EyeTracker (pestañeos). Es interesante notar que en ciertos casos se prefiera la clusterización con el mínimo número de señales a utilizar, lo que puede ser considerado en otro tipo de implementaciones.

Finalmente, es importante notar que la desviación estándar de las métricas entre conjuntos de entrenamiento-prueba es pequeña, sin embargo realizando una descomposición de variables utilizadas cada modelo de cluster, en sus dos componentes principales y graficando los resultados (imegenes 6.17, 6.18) se observa una segmentación similar en todas las particiones entrenamiento.

Tabla 6.9: Variables usadas Kmean

Señal	Variables
Phasic GSR	$avg\_amplitud$, $ultimo$, $avg\_riseTime$, $avg\_fallTime$ $avg\_riseLevel$, $avg\_fallLevel$
ECG	$median\_ECG$
PPG	muP_PPG, $stdP\_PPG$
HR	std_HR
EEG	$F3\_Theta\_fase$, $F4\_Theta\_fase$, $F7\_Alpha\_fase$ $F7\_Theta\_fase$, $F8\_Alpha\_fase$, $P7\_Alpha\_fase$ $P7\_Theta\_fase$, $P8\_Alpha\_fase$, $P8\_Theta\_fase$
Fijaciones y Pestañeos	$Blink\_N$

Tabla 6.10: Variables usadas Kmediod

Señal	Variables
Phasic GSR	n_peaks, $avg\_riseTime$, $avg\_fallLevel$
ECG	$median\_ECG$, $ECGMAD$
PPG	$stdP\_PPG$, TM_PPG
HR	$mean\_HR$
ST	$median\_ST$
EEG	$F3\_Alpha\_fase$, $F4\_Theta\_fase$, $F8\_Theta\_fase$
Pupila	$range\_Pupil$, vel_mean_Pupil
Fijaciones y Pestañeos	Fix_N

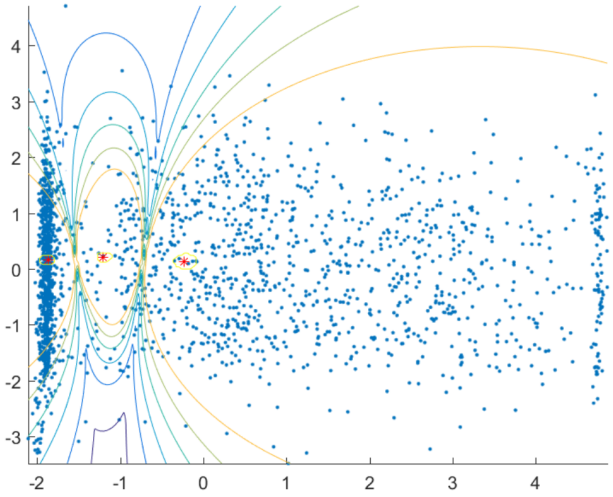


Figura 6.17: Visualización clusters generados con Kmean fig. 1.

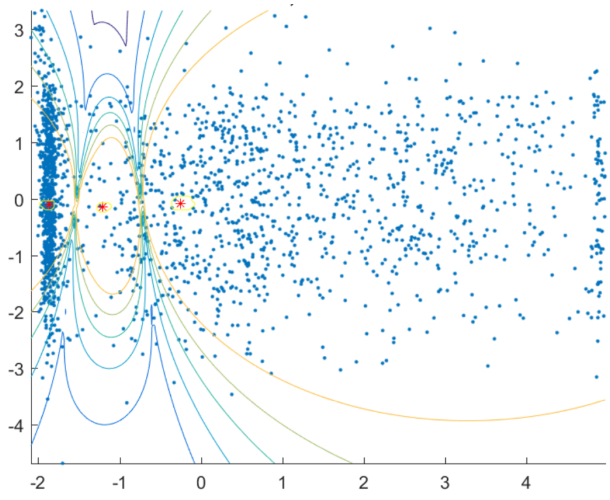


Figura 6.18: Visualización clusters generados con Kmean fig. 2.

Modelo de sacadas

Para cada cluster generado con los datos de entrenamiento se caracteriza las probabilidades de realizar diferentes sacadas. Posteriormente para los datos "nuevos" (conjunto de prueba) se considera las propiedades de las sacadas del cluster en los cuales son clasificados.

Distribución conjunta de dirección y amplitud de sacadas por cluster

Las sacadas realizadas en un estímulo en dos dimensiones, como es el caso de la página web, se pueden caracterizar por un ángulo $\phi \in [0°, 360°)$ que representa una dirección y, una amplitud d, que generalmente es medido en grados. El cálculo de estas medidas se basan en la posición de las fijaciones realizadas por el usuario y no directamente de las sacadas.

En la figura 6.19 se muestra que, dada dos fijaciones, la amplitud es medida como el cambio angular entre ellas, lo que se transforma a píxeles para mejor adaptación al estímulo (página web). En la figura 6.20 se ve que, dada dos fijaciones, con un sistema de coordenadas con origen en la primera fijación, la dirección de la sacada (ϕ) se obtiene como el ángulo entre el eje $X = 0°$ y la segunda fijación.

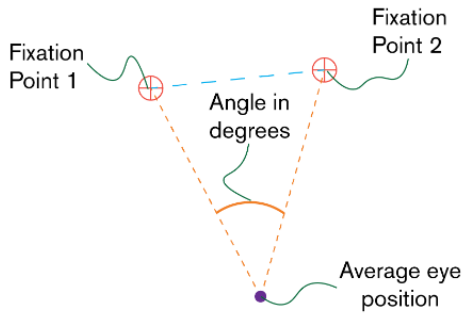


Figura 6.19: Definición amplitud de sacada.

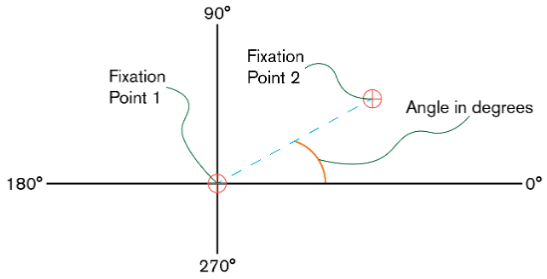


Figura 6.20: Definición dirección de sacada.

Así, para cada ventana temporal se registra mediante el EyeTracker los datos de amplitud y dirección de sacadas. Por lo tanto, se busca caracterizar las probabilidades de realizar sacadas de amplitud d y dirección ϕ, a priori en cada cluster, lo que se realizará de forma conjunta $P_c(\mathrm{d}, \phi)$.

La medición en grados mide la percepción del tamaño del estímulo más que el tamaño real. Si por ejemplo, el usuario se aleja del monitor, entonces lo verá más pequeño, por lo cual un movimiento de amplitud 100 px. requiere un movimiento visual de menos grados que cuando se encuentra cerca de la pantalla. En la imagen 6.21 se observa la relación entre los movimientos visuales medidos en grados y en píxeles.

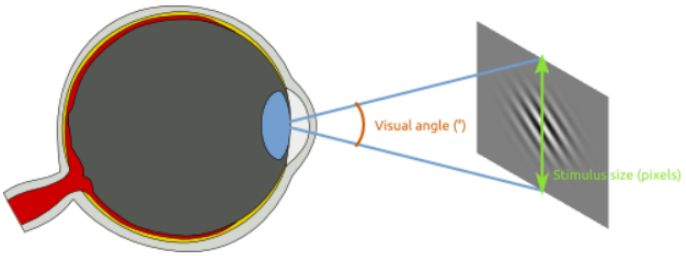


Figura 6.21: Relación entre ángulos visuales y estímulo en píxeles

El EyeTracker registra la distancia de cada ojo al monitor (mm.). Dado que durante la experimentación se les indicó a los usuarios que permanezcan sentados a una distancia cómoda para ellos de la pantalla y eviten movimientos durante la navegación, se considera el promedio entre las distancias de ambos ojos, medida en centímetros, como la distancia del participante al monitor durante toda su navegación. Para mayor precisión en ambientes donde los usuario presentan movimiento, se puede calcular la distancia promedio en cada ventana de tiempo.

Teniendo los datos de resolución y tamaño verticales de la pantalla, los cuales son $h = 27$ cm y $r = 1024$ px. respectivamente, más la distancia promedio del participante al monitor d, se puede calcular la cantidad de píxeles presentes en un grado a partir de la fórmula:

$$Pixel\_por\_grado = \left(\frac{180 \cdot \arctan2(0{,}5 \cdot h, \mathrm{d})/\pi}{0{,}5 \cdot r} \right)^{-1}$$

Posteriormente, para cada cluster se utilizan los datos de amplitud y dirección de sacada, para calcular la distribución conjunta siguiendo el método utilizado en [73]. Esto se realiza mediante un método de distribuciones no paramétrico de estimación de densidad con Kernel Gaussiano, con la optimización de los dos parámetros de ancho de banda necesarios d y h, según el método expuesto en [19].

$$P_c(\mathrm{d}, \phi) = \frac{1}{n} \sum_{\mathrm{i}} K_h(\mathrm{d} - \mathrm{d_i}, \phi - \phi_\mathrm{i})$$

La estimación se realiza según la ecuación anterior, donde K_h es un kernel gaussiano de dos dimensiones, $\mathrm{d_i}$ y ϕ_i son los datos de amplitud y dirección entre los distintos pares de fijaciones realizados en las ventanas de tiempo del cluster al cual se le hace la estimación de probabilidades.

Para aplicar el método anterior, se establece una grilla de $n \times n$, con $n = 2^8$. Al mismo tiempo, se establecen los límites $[0°, 360°)$ para los posibles valores de ϕ, y $[0, 170]$ px. para la amplitud (rango del 70 % de los casos en cada partición, en cada cluster).

Por lo tanto, para cada conjunto de entrenamiento se estima de densidad conjunta para las sacadas según cada cluster, como en el caso presentado en las imágenes 6.22, 6.23 y 6.24 correspondientes a la primera partición. Se observa en los tres gráficos la acumulación de densidad en sacadas de movimiento horizontal hacia la derecha. Este comportamiento es normal en estímulos de páginas web.

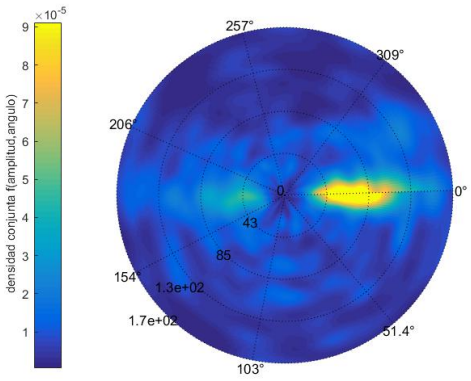


Figura 6.22: Distribución conjunta de dirección y amplitud de sacadas cluster 1, primera partición.

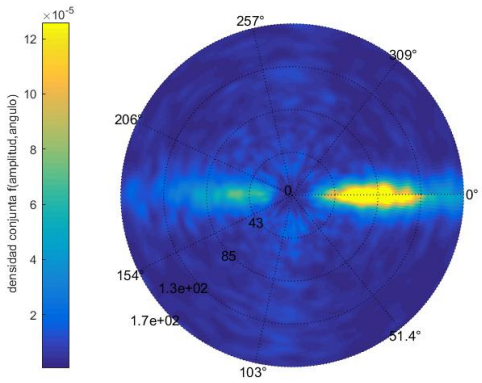


Figura 6.23: Distribución conjunta de dirección y amplitud de sacadas cluster 2, primera partición.

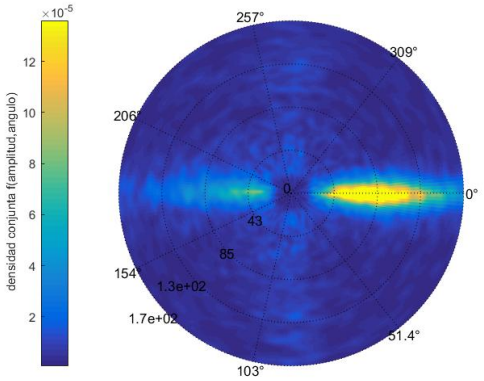


Figura 6.24: Distribución conjunta de dirección y amplitud de sacadas cluster 3, primera partición.

Cálculo a priori probabilidades de transición

Para el estímulo se genera una grilla de dimensiones 1220×1280 píxeles de ancho y alto respectivamente, en divisiones cuadradas de 20 píxeles, es decir, un total de 3904 cuadrantes. Este tamaño permite un total de cuadrantes manejable por los algoritmos usados posteriormente, e identificar las AoI presentes en la página web.

En la imagen 6.25 cada cuadrante presenta un color correspondiendo a algún área de interés. Se utiliza la función cajón superior en los cuadrantes donde se presentan fracciones de AoI. Por esta razón, las áreas correspondientes a los banners (AoI4 y AoI5) pierden la separación. Lo mismo ocurre entre las componentes "Noticias 6" y "Noticias 7" en el AoI3. Sin embargo, la definición permite una buena aproximación al estímulo real.

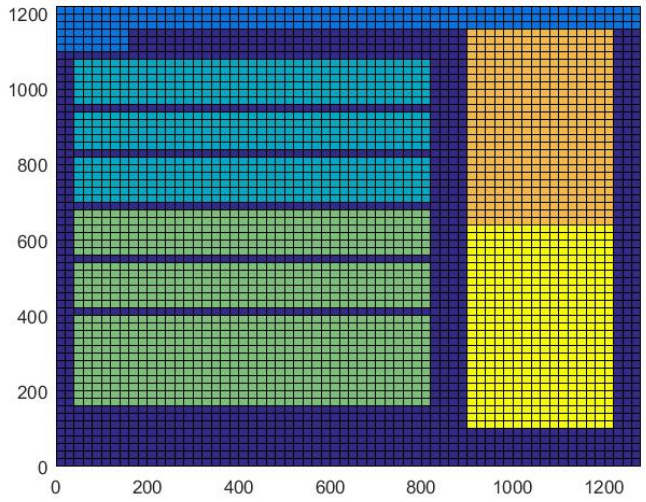


Figura 6.25: Grilla representando el estímulo visual y sus diferentes áreas de interés definidas.

La transición entre dos áreas de interés se modela como una secuencia de sacadas realizadas desde un AoI a otro, pasando por zonas white-space. Dada la fijación en algún cuadrante, se calculan las probabilidades de transición a los cuadrantes vecinos presentes dentro de una circunferencia de amplitud máxima 170 píxeles utilizando la distribución conjunta calculada.

Se genera una nueva grilla, en coordenadas cartesianas, que comprendan un cuadrado con los límites $[-170, 170]$ en los ejes X e Y, esta se transforma a coordenadas polares, se interpolan los valores presentes en la grilla polar (figura 6.27), y se vuelve a transformar a coordenadas cartesianas (figura 6.28). Se observa que esta nueva distribución no puede ser utilizada directamente en la grilla de la página web, por lo que se genera una grilla con cuadrantes de 20 píxeles por lado, como se muestra en la imagen 6.29.

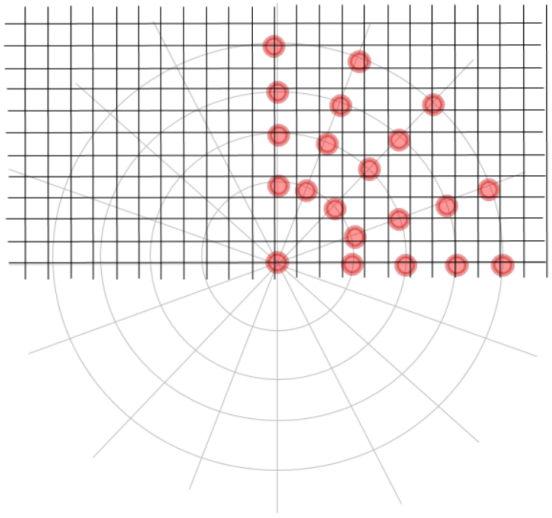


Figura 6.26: Transformación de coordenadas polares a cartesianas

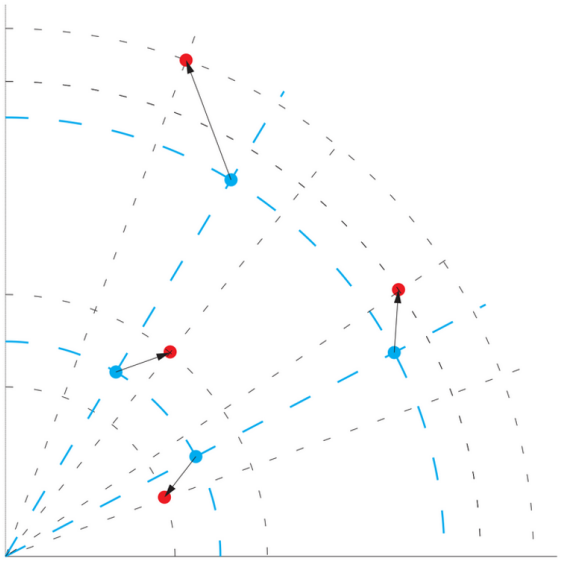


Figura 6.27: Interpolación entre grillas en coordenadas polares.

Posteriormente, se ubica cada cuadrante del estímulo al centro de la distribución de probabilidades de transición generada como grilla cuadrada, conociendo las probabilidades de transición entre cuadrantes, y posibilitando el cálculo de probabilidad de transición entre AoI's distantes.

El gráfico 6.30 representa los cuadrantes i, j, k, todos conectados. Los valores de $0 \leq p \leq 1$ representan las probabilidades de transición entre nodos. Para calcular el camino de mayor probabilidad para la transición desde el nodo i al nodo k, se debe considerar dos posibles caminos: $P_{\mathrm{i},k} = p_{\mathrm{i},k}$ y $P_{\mathrm{i},j,k} = p_{\mathrm{i},j}p_{j,k}$ donde P representa la probabilidad total de realizar dicho camino. Existen otros camino posibles, por ejemplo considerar un loops entre los nodos i, j antes de ingresar a k, sin embargo, su probabilidad será menor. Entonces lo que se busca es resolver el problema de encontrar la secuencia de mayor probabilidad.

El gráfico anterior corresponde al problema abordado, donde cada nodo es un cuadrante del estímulo y se busca la ruta de mayor probabilidad para alcanzar otros cuadrantes, sin realizar una transición directa necesariamente.

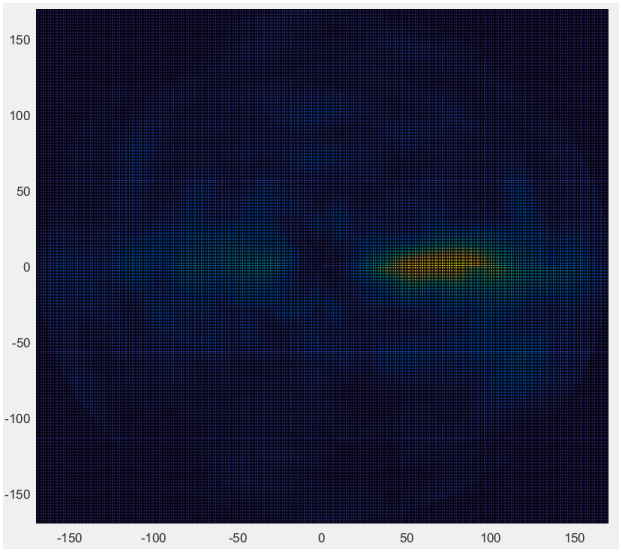


Figura 6.28: Interpolación densidad en coordenadas cartesianas.

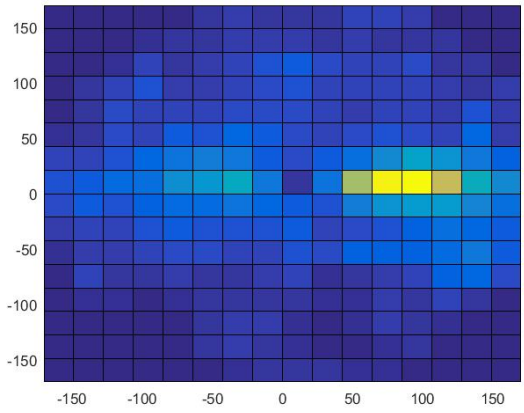


Figura 6.29: Densidad en coordenadas cartesianas simplificada.

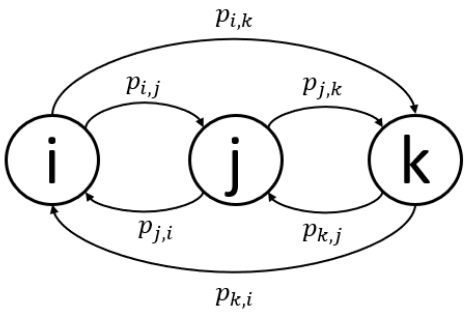


Figura 6.30: Grafo con probabilidades de transición.

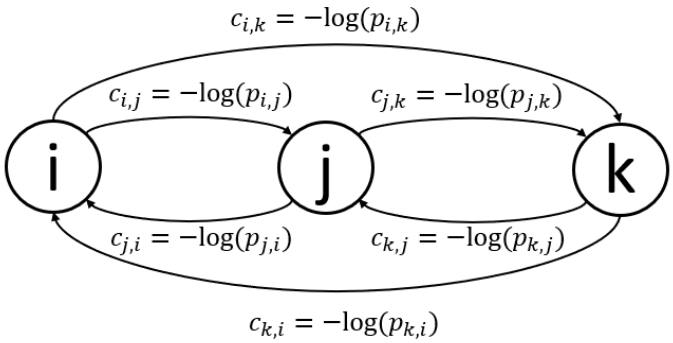


Figura 6.31: Grafo con costos de transición.

En la figura 6.31 se aplica la función menos logaritmo a los valores de los arcos. Dado que las probabilidades son menor a 1, los valores de $log(p)$ serán negativos. Valores más pequeños de p entregan un valor menor. Los caminos antes señalados tendrán costos $C_{i,k} = c_{i,k} = p_{i,k}$ y $C_{i,j,k} = c_{i,j} + c_{j,k} = -log(p_{i,j}) - log(p_{j,k}) = -log(p_{i,j}p_{j,k})$ los cuales al ser minimizados permitirán la maximización de las probabilidades correspondientes a esos caminos. Por lo tanto, el problema puede ser resuelto por métodos de cálculo de camino mínimo, con lo cual se utiliza el algoritmo de Dijkstra.

Resolviendo lo anterior para cada cluster, se obtiene el valor de camino mínimo entre cada nodo y los nodos de los AoI definidos. En las imágenes 6.32, 6.33 y 6.34 se presentan ejemplos de camino, entre dos nodos para cada cluster. Se obtienen resultados diferenciados entre grupos.

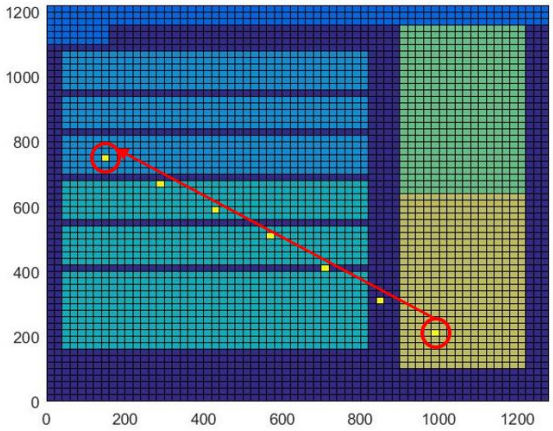


Figura 6.32: Camino de máxima probabilidad entre dos puntos cluster 1.

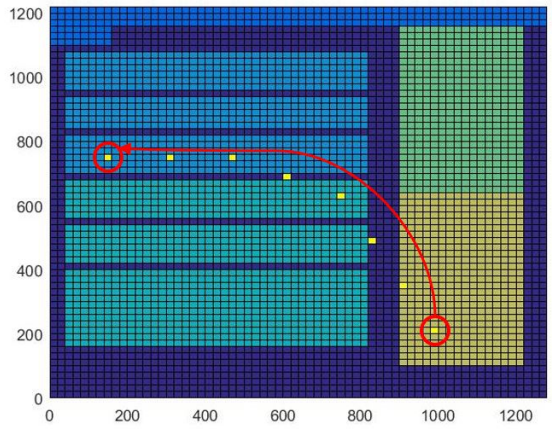


Figura 6.33: Camino de máxima probabilidad entre dos puntos cluster 2.

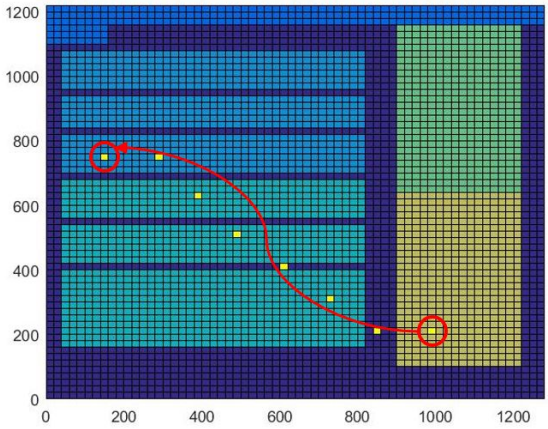


Figura 6.34: Camino de máxima probabilidad entre dos puntos cluster 3.

6.3.6. Etapa 3: Generación de caminos visuales.

Características EEG relacionadas a movimiento ocular

Según lo planteado en 5.3.1, en la predicción de caminos visuales se utiliza, para cada paso predicho, las variables independientes $EEG(v)$, relacionadas a los mecanismos internos del cerebro en relación a las sacadas. A continuación se detalla el procedimiento para calcular dichas características.

Para comprender qué electrodos son útiles en la caracterización de las sacadas, se utilizan dos enfoques: uno computacional, en el que se prueban diferentes características extraídas de diferentes electrodos, y uno basado en teoría de la "neurociencia cognitiva".

En la "Etapa dos del modelo" se utilizan las señales Alpha y Theta registradas en los electrodos $\{F3, F4, F7, F8, P7, P8\}$ para la caracterización de estado mental. Estas señales nos son utilizadas en el cálculo de $EEG(v)$, para evitar redundancia. Por lo tanto, en el enfoque computacional los electrodos usados son $\{AF3, AF4, FC5, FC6, T7, T8, O1, O2\}$.

El enfoque teórico se basa en estudios encontrados en el área de neurociencia cognitiva. En la mayoría de los estudios de atención con registro de neuronas individuales se han centrado en el área intraparietal, especialmente el surco intraparietal (IPS) y el área intraparietal lateral (LIP), región involucrada en los movimientos oculares sacádicos y en la atención viso-espacial.

En [16] evidencian que la actividad en LIP proporciona un mapa de relevancia o prioridad (idea similar a saliency maps), utilizado por el sistema oculomotor como un objetivo sacádico. El sistema visual utiliza este mapa para determinar el lugar de atención.

Según [p. 607][78], el área IPS forma parte de una red de áreas que median movimientos sacádicos de los ojos. Recibe aferencias de varias áreas visuales. En el mono macaco se encuentra en la pared lateral del surco intraparietal, de ahí su nombre. Sin embargo, en los humanos se localiza en el surco intraparietal medial.

El IPS posee las regiones: anterior, lateral, ventral, caudal y medial. Además, se identifican las siguientes áreas:

- **LIP** y VIP: involucrado en la atención visual y movimientos oculares sacádicos.
- VIP y MIP: control visual de alcanzar y señalar.
- AIP: control visual de agarrar y manipular movimientos de la mano
- CIP: percepción de la profundidad de la estereopsis.

Como el encefalograma sólo permite mediciones a nivel de cortex, no es posible medir directamente las señales del LIP. Por lo tanto, usando el estándar 10-20 de EEG, se buscan los electrodos que permitan una medición del IPS y sus áreas descritas.

Según lo expuesto en [46] utilizar el electrodo $P3$ alcanza principalmente las áreas de Brodmann (región de la corteza cerebral definida con base en su citoarquitectura, es decir, disposición de las células que constituyen la corteza cerebral) AB-40 y en menor grado AB-7 del lóbulo parietal inferior y posterior en la región del IPS. Sin embargo, no se posee la

medición de dicho electrodo, por lo cual, este enfoque queda propuesto para otras posibles implementaciones.

Para cada electrodo a usar se aplica un filtro pasa alto de frecuencia de corte de 0.5 Hz y un filtro pasa bajos de frecuencia de corte de 40 Hz, además de un filtro de Hampel, como pre-procesamiento de las señales, de forma similar a lo expuesto en la sección 6.3.5.

Para el enfoque computacional se propone la utilización de características extraídas de cada uno de los electrodos. Un posible enfoque es el análisis de frecuencias espectrales. Para el enfoque según la teoría de neurociencia cognitiva, se utiliza un solo canal ($P3$), y su poder espectral puede ser analizado a partir de análisis de Fourier o transformadas de Wavelet. Según lo expuesto en [1], este método no es adecuado para predecir interdependencia de los datos en diferentes canales.

Un primer método dentro del enfoque computacional es usar el método propuesto en [1] para la generación de características. Se seleccionan características relacionadas al poder espectral para caracterizar cada una de las bandas de frecuencias: delta (1-4 Hz), theta (4-8 Hz), alpha (8-12 Hz), beta (12-25 Hz), gamma (sobre 25 Hz). Este proceso es desarrollado con la herramienta EEGLAB [32]. Este método será denominado **Análisis Espectral de EEG** en las secciones siguientes.

Un segundo método corresponde a la utilización de las señales de cada electrodo para generar características desde la señal cruda. Este método será denominado **Análisis señal cruda EEG** en las secciones siguientes.

Primero, se aproxima la señal total a los últimos 20 segundos de navegación, anteriores a la próxima ventana de tiempo. Para el primer método, se realiza lo siguiente:

Al analizar el poder espectral de los 8 canales utilizados, se observa una superposición de señales. Para evitar esto, se realiza un análisis de componentes independientes (ICA, por sus siglas en inglés).

El método ICA, asume que si se tienen N señales observadas X_{i}, donde en este caso $N = 8$ electrodos, cada señal es una mezcla lineal de N señales de fuente independientes S_{i}.

$$X = AS$$

Como la matriz A, correspondiente a $N \times N$ factores desconocidos, se utilizan algoritmos que estiman la matriz de separación W para la estimación de las señales de fuente.

$$S = WX$$

Posteriormente las 8 componentes son reducidas a una única componente, mediante el análisis de componentes principales (PCA), entregando mejor ajuste en la lectura de actividades de espectro de potencia para diferentes frecuencias.

Finalmente, se separa la componente generada en los rangos de frecuencia a considerar. Para cada rango, se calcula:

- Porcentaje de potencia presente en cada banda respecto al total. Esto entregará indicadores de que bandas de frecuencia se encuentran más activas al comienzo de una nueva

ventana de tiempo.

- Potencia espectral promedio para cada banda.

Para el segundo método se utilizan las señales de los electrodos preprocesadas para calcular las siguientes características:

- Potencia dentro de la ventana de tiempo.
- Fase de la señal dentro de una ventana de tiempo.

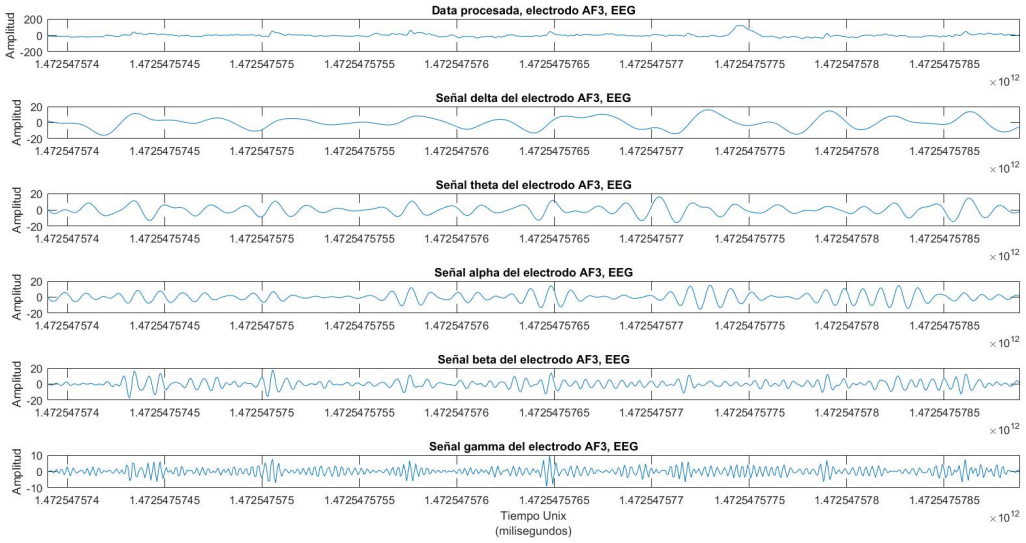


Figura 6.35: Bandas señal EEG obtenidas para el electrodo AF3, para alguna ventana de tiempo.

Cada método es desarrollado de manera separada y los resultados de cada modelo son comparados más adelante.

Predicción camino visual

La "predicción" de caminos visuales se basa en la generación de secuencias a partir de una caracterización de los datos procesados del usuario. Las secuencias, corresponden a los caminos visuales simples y/o temporales. En las etapas anteriores se generaron los procesamientos de datos para la obtención de indicadores de intención de visita a cada área de interés y las probabilidades de transición entre AOI's según el estado mental del usuario al principio de cada ventana.

Los modelos de redes neuronales recurrentes, con sus variaciones (ver sección 4.3.1), permiten generar secuencias a partir de datos entregados como inputs. De esta manera, en la etapa de entrenamiento, el modelo aprende a utilizar los datos entregados como semilla para el cálculo de probabilidades en cada posición de la secuencia. Esto es generalizado para el conjunto de testeo, donde se entregan datos del usuario para la generación de la secuencia de Áreas de interés en la próxima ventana de tiempo.

Existen dos conjuntos de variables usadas en la predicción en una ventana de tiempo, las que dependen de la posición y las que se mantienen constantes durante la ventana.

Las variables que durante toda la ventana de tiempo se mantienen constantes son la intención de visita, calculada una vez antes de comenzar la ventana temporal, las variables $EEG(v)$ y la secuencia real presentada en la ventana anterior, la cual es conocida. Estas se integran dentro del estado oculto inicial H_0 de la red neuronal. Al igual que lo realizado en las arquitecturas encoder-decoder.

En la imagen 6.36 se observa gráficamente lo mencionado. El estado oculto inicial H_0 se calcula a partir de la intención de visita obtenida en la etapa 1, I, las variables $EEG(v)$ y la secuencia observada en la etapa anterior SA, calculada a partir de sus elementos $S_1, \ldots, S_L$, donde L corresponde al largo de la secuencia de la ventana anterior.

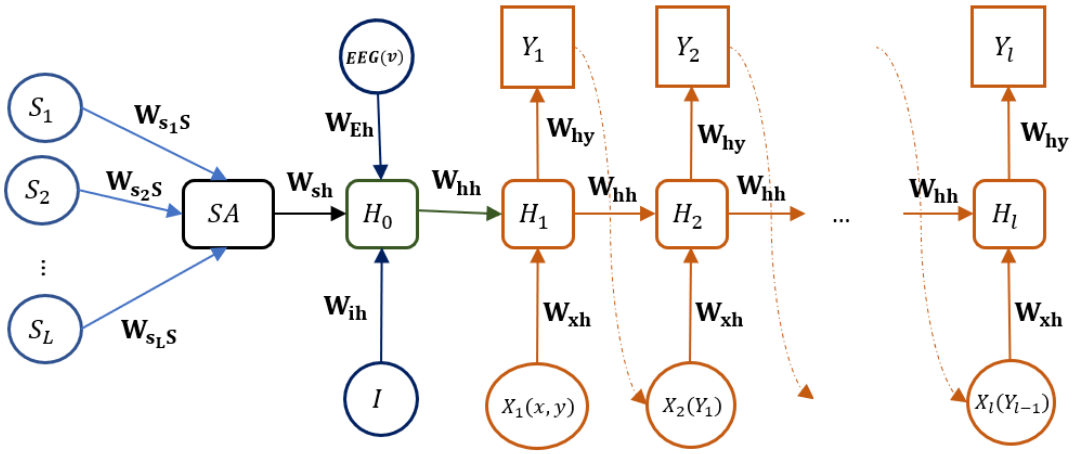


Figura 6.36: Diagrama de red neuronal recurrente con estimación de estado inicial.

En la imagen se observa que para cada uno de los l pasos temporales (donde l es el largo de la secuencia a predecir), se entrega un input que dependerá de la predicción realizada en la etapa anterior. Se observa además que para el primer paso temporal, el input dependerá del nodo en el cual se encuentra la mirada del usuario al comenzar la ventana de tiempo $X_1(x, y)$.

La red RRN de la figura 6.36 queda formalmente representada por las ecuaciones de red neuronal 6.3, estado inicial 6.4 y función de pérdida 6.5 aplicada para la estimación de parámetros de la red $\{b, W_{hh}, W_{xh}, c, W_{hy}, \mathrm{e}, s_{\mathrm{i}}, \mathrm{d}, W_{sh}, W_{\mathrm{ih}}\}$. Los términos $\mathrm{ee}g_1, \ldots, \mathrm{ee}g_Q$ representan las Q variables $EEG(v)$ utilizadas.

Red neuronal recurrente ($t = 1, \ldots, l$):

$$a^{(t)} = b + W_{hh} h^{(t-1)} + W_{xh} x^{(t)} \tag{6.3a}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right) \tag{6.3b}$$
$$o^{(t)} = c + W_{hy} h^{(t)} \tag{6.3c}$$
$$\hat{y}^{(t)} = softmax(o^{(t)}) \tag{6.3d}$$

Estado oculto inicial (H_0):

$$EEG_a = f + \mathrm{ee}g_1 W \mathrm{eg} g_{1h}, \ldots, \mathrm{ee}g_Q W \mathrm{eg} g_{Qh} \tag{6.4a}$$
$$EEG = \tanh\left(EEG_a\right) \tag{6.4b}$$
$$SA_a = \mathrm{e} + s_1 S_{1h}, \ldots, s_L S_{Lh} \tag{6.4c}$$
$$SA = \tanh\left(SA_a\right) \tag{6.4d}$$
$$a^{(0)} = \mathrm{d} + W_{\mathrm{eh}} EEG + W_{sh} SA + W_{\mathrm{ih}} I \tag{6.4e}$$
$$h^{(0)} = \tanh\left(a^{(0)}\right) \tag{6.4f}$$

Loss Function:

$$\begin{aligned} L(\{x_1, \ldots, x_l, s_1, \ldots, s_L, I\}\{y_1, \ldots, y_l\}) &= \sum_t L^{(t)} \\ &= -\sum_t \log\left(y^{(t)} | \{x_1, \ldots, x_t, s_1, \ldots, s_t, I\}\right) \end{aligned} \tag{6.5}$$

El modelo RNN anterior puede ser programado manualmente. Para entrenar la red (estimar parámetros a partir de los datos de entrenamiento), se utiliza el método back propagation throw time (BPTT), adaptado a la estimación del estado oculto inicial. Esto se hace estimando el gradiente de cada parámetro de forma recurrente. Para cada uno de los N nodos de estado oculto se calcula el gradiente $\nabla_N L$ a partir del gradiente calculado para el nodo siguiente. Estos cálculos se encuentran en la sección 8.3.

Lo anterior permite un mejor entendimiento del modelo. Por otro lado, la utilización del framework TensorFlow para la ejecución de modelos RNN realiza un cálculo para la estimación de gradientes de manera automática. Al mismo tiempo facilita la creación de modelos mas complejos, por ejemplo, al integrar una mayor cantidad de capas ocultas (véase redes apiladas 4.3.2).

Un problema de las redes RNN es que pueden presentar "overfitting" en la etapa de entrenamiento, reduciendo sus habilidades de predicción. Es decir, se sobre-ajustan los resultados del conjunto de entrenamiento (la red estima bien los resultados en este conjunto) pero estos resultados no son generalizables al conjunto de testeo.

Existe el método de dropout, un método de regularización en las conexiones entre las entradas y las unidades de las celdas LSTM-GRU (lo que no se utiliza en esta implementación), o entre estas últimas y la capa de salida, donde se excluyen de forma probabilísticas de la activación y actualización de parámetros, mientras se entrena la red. Para ello se establecen los valores de probabilidad de mantención de dropout (donde 1 significa que no se agrega dropout).

Además de lo anterior, se utiliza "gradient clipping" (recorte de gradiente) de valor 5. Esto evita que al realizar backpropagation a través del tiempo, los gradientes exploten (cuando existen multiplicaciones con valores mayores a 1) o desaparezcan (cuando se multiplican por valores menores a 1), manteniendo sus valores dentro de un intervalo controlado.

Grandes conjuntos de variables obligan a la utilización de más recursos y algunas veces generan que la ejecución de los algoritmos demoren más tiempo. Otro inconveniente es que muchos algoritmos de machine learning muestran un decaimiento del desempeño cuando el número de variables es significativamente mayor al óptimo [66]. Por lo tanto, en algunas aplicaciones es deseable considerar un subconjunto de variables lo que mejora los costos computacionales, costo de los sensores y evita el ruido provocado por la mayor cantidad de puntos considerados en los algoritmos usados [71].

En este caso, la utilización de las variables $EEG(v)$ (véase 6.3.6) puede estar lejos de la cantidad óptima de variables a considerar en cada modelo. Por esta razón, los modelos propuestos en esta sección serán tratados con métodos de selección de variables explicados en la sección 4.3.6. En particular se utiliza "Principal Feature Analysis" para la selección de un subconjunto de variables antes del procesamiento de las redes neuronales propuestas, y "Fordward and Backward Selection" para la selección de variables dentro de este nuevo subconjunto en relación a cada modelo.

Considerando lo anterior se ejecutan los diferentes modelos. En primer lugar, se ejecuta el modelo considerando los **caminos visuales simples**. En este caso se requiere escoger un largo l fijo para cada secuencia. Para ello, se calculan los caminos visuales simples para la totalidad de datos de manera ex-post, según el procedimiento expuesto en 6.3.2. Así, se obtiene la tabla 6.11.

Tabla 6.11: Tabla de frecuencia largos de secuencias en ventanas de tiempo

Largo	0	1	2	3	4	5
N° ventanas	52	872	666	515	388	210
% acumulado	0,0181	0,3213	0,5529	0,7319	0,8668	0,9398

Largo	6	7	8	9	10	11
N° ventanas	95	50	20	3	4	1
% acumulado	0,9729	0,9903	0,9972	0,9983	0,9997	1,0000

El 97 % de los casos pueden ser representados con un largo $l = 6$. Así, se considera que el largo del camino visual real de la ventana y de la ventana anterior tendrán el mismo largo $l = L = 6$. Las secuencias de largo superior se truncan a los primeros 6 elementos, y las largo inferior, se completa la secuencia de 6 elementos con ceros al final.

Para cada ventana de tiempo se genera una distribución de probabilidad de AOI visitado por el usuario con la red neuronal recurrente, en cada paso temporal $t \in \{1, \ldots, l\}$. Con esto, es posible genera una muestra para la secuencia en dicha ventana. En un primer enfoque se generan caminos visuales simples utilizando el muestreo de "argmax".

Los caminos visuales muestreados son generados escogiendo el AOI con mayor probabilidad en cada paso temporal $t \in \{1, \ldots, l\}$. De esta manera, en cada modelo las secuencias generadas son comparadas con los caminos reales. Para ello, se proponen las siguientes métricas.

Cantidad de desigualdades: Dada dos secuencias de mismo largo $\{x_1, \ldots, x_l\}$ y $\{y_1, \ldots, y_l\}$ se calcula como la cantidad de desigualdades elemento a elemento, es decir:

$$NotEqual = \sum_{\mathrm{i}=1}^{l} \mathbb{1}(x_{\mathrm{i}} \neq y_{\mathrm{i}})$$

Esta es una métrica de computo simple, sin embargo no entrega información acerca de la distancia entre los elementos.

Distancia de Levenshtein: La distancia de Levenshtein, creada por el científico ruso Vladimir Levenshtein, también conocida como distancia de edición, es el número mínimo de operaciones requeridas para transformar una cadena de caracteres en otra. Las operaciones permitidas son la inserción, eliminación o sustitución de un carácter.

A modo de ejemplo, la distancia de Levenshtein entre las cadenas de caracteres "casa" y "calle" es de 3, necesitándose las operaciones:

- casa $\rightarrow$ cala (sustitución de 's' por 'l')
- cala $\rightarrow$ calla (inserción de 'l' entre 'l' y 'a')
- calla $\rightarrow$ calle (sustitución de 'a' por 'e')

Esta medida representa un costo unitario por cada operación realizada, por lo que cadenas más similares presentaran una distancia menor. La distancia de Levenshtein es útil en procesos como la revisión ortográfica de palabras y en este caso será usado para comparar las secuencias de áreas de interés. En primera instancia, se evalúan los modelos utilizando las métricas de función de pérdida, distancia de Levenshtein y cantidad de desigualdades. La primera métrica entrega una idea de la calidad de las probabilidades generadas por el modelo en cada paso temporal. Las últimas dos, entregan una evaluación de la calidad de las secuencias generadas como predicciónes a caminos visuales simples en cada ventana.

Primero, se calcula el modelo para la segmentación de la página web en 6 AOI's. Se ejecuta el modelo usando validación cruzada de 10-folds. Cada cierto número de iteraciones en el proceso de entrenamiento, se evalúa el modelo en los conjuntos de testeo. El modelo se entrena con las variables de "Análisis espectral" y utilizando todas las variables definidas, se prueban diferentes configuraciones, mostradas en la tabla 6.12.

La tabla 6.12 muestra para cada configuración los valores de la función de pérdida en el conjunto de entrenamiento y de testeo. Además se muestran los valores de "distancia de Levenshtein" y "cantidad de desigualdades" ("non equal") para la comparación entre los caminos visuales simples reales y los generados por el modelo.

Tabla 6.12: Modelos camino visual simple, Analisis Espectral, 6 AOI's

			Todas las variables							
			N° Neuronas							
Modelo	**N° Capas**	**Dropout P(keep)**	**128 [TrainL - TestL - DL - NE]**				**256 [TrainL - TestL - DL - NE]**			
RNN	*1*	*1*	0,104 (0,017)	3,543 (0,408)	2,296 (0,084)	0,390 (0,014)	0,015 (0,004)	5,032 (0,585)	2,299 (0,112)	0,388 (0,018)
		0,8	0,242 (0,025)	2,550 (0,334)	2,202 (0,094)	0,374 (0,016)	0,058 (0,017)	3,686 (0,403)	2,146 (0,142)	0,363 (0,023)
	2	*1*	0,013 (0,003)	5,072 (0,463)	2,360 (0,108)	0,400 (0,02)	0,001 (0,001)	5,504 (0,397)	2,296 (0,086)	0,391 (0,016)
		0,8	0,129 (0,018)	2,777 (0,278)	2,173 (0,096)	0,367 (0,018)	0,029 (0,007)	3,910 (0,482)	2,140 (0,125)	0,361 (0,021)
	3	*1*	0,004 (0,003)	5,367 (0,331)	2,283 (0,097)	0,388 (0,016)	0,001 (0,001)	5,166 (0,398)	2,289 (0,158)	0,388 (0,027)
		0,8	0,097 (0,011)	2,793 (0,211)	2,155 (0,068)	0,364 (0,013)	0,037 (0,005)	3,401 (0,325)	2,131 (0,13)	0,359 (0,022)
LSTM	*1*	*1*	0,091 (0,025)	3,621 (0,581)	2,280 (0,148)	0,388 (0,025)	0,009 (0,005)	5,575 (0,57)	2,314 (0,126)	0,395 (0,023)
		0,8	0,173 (0,017)	3,095 (0,389)	2,198 (0,096)	0,373 (0,017)	0,026 (0,009)	4,846 (0,566)	2,177 (0,103)	0,368 (0,017)
	2	*1*	0,008 (0,004)	5,497 (0,586)	2,287 (0,124)	0,389 (0,022)	0,001 (0,001)	5,814 (0,613)	2,195 (0,121)	0,372 (0,02)
		0,8	0,068 (0,01)	3,591 (0,253)	2,180 (0,097)	0,369 (0,016)	0,017 (0,005)	4,556 (0,533)	2,176 (0,083)	0,368 (0,016)
	3	*1*	0,004 (0,002)	5,102 (0,463)	2,242 (0,083)	0,380 (0,016)	0,001 (0,001)	5,264 (0,52)	2,206 (0,127)	0,374 (0,021)
		0,8	0,053 (0,007)	3,534 (0,468)	2,186 (0,11)	0,370 (0,018)	0,015 (0,004)	4,329 (0,563)	2,188 (0,07)	0,370 (0,013)
GRU	*1*	*1*	0,051 (0,016)	4,827 (0,791)	2,431 (0,156)	0,413 (0,026)	0,004 (0,003)	6,468 (0,717)	2,367 (0,109)	0,403 (0,019)
		0,8	0,111 (0,023)	3,701 (0,571)	2,287 (0,134)	0,389 (0,024)	0,014 (0,005)	5,415 (0,552)	2,252 (0,109)	0,381 (0,018)
	2	*1*	0,003 (0,002)	5,535 (0,493)	2,353 (0,111)	0,400 (0,02)	0,000 (0,0001)	5,585 (0,426)	2,337 (0,134)	0,397 (0,023)
		0,8	0,038 (0,01)	3,911 (0,474)	2,251 (0,082)	0,381 (0,014)	0,006 (0,003)	5,042 (0,566)	2,200 (0,115)	0,372 (0,02)
	3	*1*	0,001 (0,001)	5,218 (0,471)	2,283 (0,114)	0,390 (0,021)	0,00002 (0,00002)	5,032 (0,434)	2,254 (0,132)	0,382 (0,023)
		0,8	0,026 (0,006)	3,822 (0,312)	2,182 (0,085)	0,369 (0,014)	0,011 (0,003)	4,289 (0,363)	2,195 (0,095)	0,370 (0,016)

Se observa de estos modelos que en general la función de pérdida calculada en el conjunto de entrenamiento es menor para una mayor cantidad de neuronas, es decir, los resultados mejoran en los casos de 256 neuronas y al tener una mayor cantidad de capas (profundidad del modelo). Pese a lo anterior, los resultados de la función de pérdida calculada en los conjunto de testeo tienden a aumentar al aumentar el número de neuronas de 128 a 256.

De la tabla de resultados de las diferentes configuraciones se puede concluir que en general, dado que el modelo con 256 neuronas por capa presenta valores más bajos en la función de pérdida de test, se prefiere un modelo de 128 neuronas. Al mismo tiempo, aumentar el número de capas se mejora el proceso de entrenamiento, por lo que se prefiere el modelo de 3 capas. Sin embargo, es importante considerar que para la consideración de un mayor número de capas, es necesaria la integración de una probabilidad de mantención de dropout, en este caso de 0.8, lo que permite que la generalización del aprendizaje del modelo en el conjunto de test, no empeore por el mayor número de capas.

De los tres tipos de celdas probados en los modelos, se observa que las celdas RNN normales entregan las funciones de pérdida más altas en los conjuntos de entrenamiento (peor desempeño) que las celdas LSTM y GRU, siendo estas últimas las de mejor desempeño en el entrenamiento. Sin embargo, las celdas RNN tienden a entregar una mejor generalización de lo aprendido en el proceso de entrenamiento en los conjuntos de testeo.

Finalmente, se observa que las métricas de "distancia de Levenshtein" y "non equal" no tienden a mejorar demasiado en las diferentes configuraciones. Por lo tanto, el modelo no cumple con lo buscado.

El modelo anterior debe ser tratado contra el sobre entrenamiento, es decir, permitir que el modelo pueda generalizar lo aprendido durante su fase de entrenamiento en los conjuntos de test.

Se busca evitar el sobre entrenamiento al disminuir la cantidad e información suministrada a la red. En la tabla 6.13 se prueba que para la configuración más básica (RRN con una capa) la selección de variables por método PFA, manteniendo tres variables. Las funciones de pérdida de los modelos en el conjunto de testeo mejoran a los de los modelos anteriores (sin selección de variables). Sin embargo, de los caminos sampleados, no se obtienen mejoras en cuanto a las métricas de similitud respecto a los caminos visuales reales. Por lo tanto, se muestra que la selección de variables para el estado inicial es fundamental, pero no se prueban nuevas configuraciones ya que no se logra mejoras significativas.

Tabla 6.13: Modelos camino visual simple, Análisis Espectral, 6 AOI's, PFA de tres variables

			Principal feature analysis							
			N° Neuronas							
Modelo	**N° Capas**	**Dropout P(keep)**	**128 [TrainL - TestL - DL - NE]**				**256 [TrainL - TestL - DL - NE]**			
RNN	***1***	***1***	0,16 (0,029)	2,95 (0,319)	2,28 (0,135)	0,39 (0,024)	0,04 (0,013)	4,14 (0,556)	2,28 (0,13)	0,39 (0,022)
		0,8	0,31 (0,027)	2,21 (0,248)	2,19 (0,09)	0,37 (0,017)	0,12 (0,023)	3,12 (0,444)	2,17 (0,131)	0,37 (0,022)

Finalmente, se prueban configuraciones para la predicción en un contexto de 4 AOI's. Dado lo aprendido en los modelos anteriores, se prueban configuraciones con probabilidades de mantención de dropout de 0.8, y utilizando el modelo básico (RNN de una capa) se presentan los resultados luego de seleccionar variables del análisis espectral. En primer lugar se prueba el modelo con la totalidad de variables, y sus resultados se muestran en la tabla 6.14.

Tabla 6.14: Modelos camino visual simple, Análisis Espectral, 4 AOI's

			Todas las variables							
			N° Neuronas							
Modelo	**N° Capas**	**Dropout P(keep)**	**128** **[TrainL - TestL - DL - NE]**				**256** **[TrainL - TestL - DL - NE]**			
RNN	***1***	***0,8***	0,129 (0,025)	2,620 (0,399)	2,084 (0,145)	0,354 (0,025)	0,018 (0,007)	4,186 (0,735)	2,102 (0,109)	0,357 (0,021)

De la tabla 6.14 se observa que el modelo presenta resultados similares a los del caso de 6 AOI's. Nuevamente el aumentar la cantidad de neuronas provoca la perdida de generalización del aprendizaje en los conjuntos de datos de testeo. Por esta razón se utiliza la configuración con la selección de tres variables por el método de PFA. Los resultados se muestran en la tabla 6.15.

Tabla 6.15: Modelos camino visual simple, Análisis Espectral, 4 AOI's, PFA de tres variables

			Principal feature analysis							
			N° Neuronas							
Modelo	**N° Capas**	**Dropout P(keep)**	**128** **[TrainL - TestL - DL - NE]**				**256** **[TrainL - TestL - DL - NE]**			
RNN	***1***	***0,8***	0,162 (0,031)	2,431 (0,381)	2,044 (0,117)	0,345 (0,02)	0,061 (0,026)	3,265 (0,378)	2,049 (0,1)	0,348 (0,018)

De la tabla 6.15 se observa nuevamente un comportamiento similar al modelo de 6 AOI's. La selección de variables mejora la función de pérdida calculada en los conjuntos de testeo, lo que explica una mejor generalización del aprendizaje adquirido por el modelo, cuando se ingresa un menor número de variables a la estimación del estado inicial de la red. Sin embargo, no se logra la suficiente mejorías del modelo.

Finalmente se decide explorar un nuevo método de selección de variables. Para ello se corre el modelo con la selección de variables con el método backward selection. No se prueba el método de forward selection ya que se esperan resultados similares. Dado el tiempo de ejecución de este modelo se utiliza directamente la configuración con mayor número de neuronas para probar su factibilidad. Los resultados se muestran en la tabla 6.16

De la tabla 6.16 se observa que este método mejora la generalización de los resultados en el conjunto de test. Esto se aprecia porque la función de pérdida de este conjunto es menor que cuando se hace selección de variables mediante PFA. Sin embargo, al mismo tiempo se observa que la función de pérdida en el conjunto de entrenamiento aumenta. Dada la complejidad computacional de este modelo (debe probar todas las combinaciones de variables EEG en el modelo), se concluye que los resultados de esta modelación no logran resultados satisfactorios en cuanto a la predicción de caminos visuales simples en ventanas de tiempo.

Tabla 6.16: Modelos camino visual simple, Análisis Espectral, 4 AOI's, Backward selection

			Backward Selection			
			N° Neuronas			
Modelo	**N° Capas**	**Dropout P(keep)**	**256** **[TrainL - TestL - DL - NE]**			
RNN	***1***	***0,8***	0,275 (0,006)	1,944 (0,253)	1,870 (0,138)	0,319 (0,026)

Dado que lo anterior se desarrolla con caminos visuales simples, se espera que un escenario más complejo, como el caso de los caminos visuales temporales, presenten peores resultados. Para evidenciarlo, se ejecuta el modelo con 4AOI's, escogiendo un parámetro $q = 200$ ms. De esta manera, el camino visual temporal corresponde a una secuencia de largo 25.

Al ejecutar un modelo de 128 neuronas con todas las variables EEG en los primeros dos folds (ilustraciones 6.37 y 6.38), observamos que aunque el modelo aprende a minimizar la función de pérdida en los conjuntos de entrenamiento, no es capaz de generalizarlo a los conjuntos de testeo. Al mismo tiempo, no se logran mejoras en las métricas de distancia de levenshtein y not-equal a lo largo de las iteraciones.

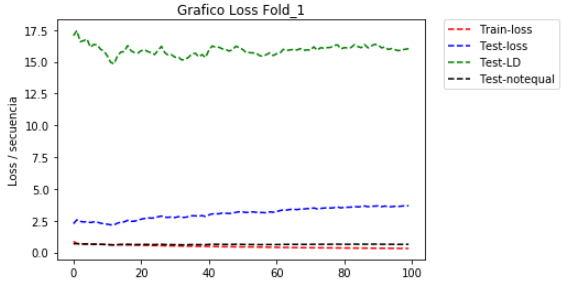


Figura 6.37: Función de perdida modelo de predicción caminos visuales temporales, fold 1

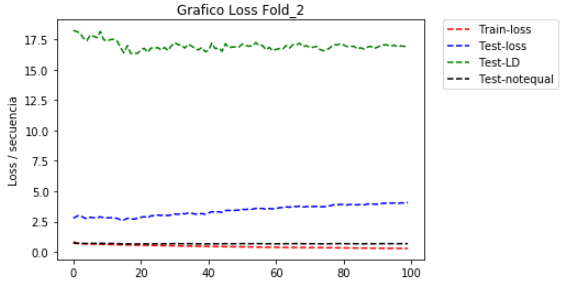


Figura 6.38: Función de perdida modelo de predicción caminos visuales temporales, fold 2

A pesar de lo anterior, es posible perfeccionar los modelos antes descritos para el caso de caminos visuales simples, probando la utilización de variables EEG, con el enfoque de señal cruda descrito en secciones anteriores. Sin embargo, dado que estas corresponden a una mayor cantidad de variables, la selección de una configuración adecuada y de las variables a considerar en la estimación del estado inicial cobra vital importancia, donde lo aprendido hasta este punto puede ser de gran utilidad.

Al mismo tiempo, mejoras en las métricas de LD y NE por si solas no necesariamente indican un buen desempeño del modelo, aunque si permiten una evaluación rápida de los resultados. En caso de lograr mejores valores, es importante considerar la métrica de **Scanmatch**, la cual permite entender mejor la cercanía entre los resultados reales y los predichos, agregando costos en función de la distancia y similitud de distintas AOI's.

Método Scanmatch: El método ScanMatch [30] permite obtener una medida de distancia que dependa de la similitud entre las diferentes áreas de interés, y es útil cuando el orden de las fijaciones es importante para caracterizar el comportamiento de los usuarios.

Para evaluar el modelo de esta forma, será necesaria la caracterización de la matriz de distancia entre diferentes AOI's, y del valor de penalización de gaps. Para ello es importante considerar la distancia entre AOI's en relación a sus características, similitud semántica y distancia en coordenadas dentro del estímulo, lo cual requiere de un desarrollo más detallado planteado para trabajo futuro.

6.4. Modelo predicción de fijaciones: segundo escenario

6.4.1. Idea general y esquema de solución

Un segundo enfoque aplicado es la utilización de la información contenida a nivel píxel a píxel, según lo expuesto en 2.2.1. En dicha sección, la formulación original corresponde a la predicción de secuencias de coordenadas referidas a las fijaciones que realiza el usuario en una ventana de tiempo.

En el estudio [10] nombrado en la sección de estado del arte, se realiza una predicción de este tipo en videos. Sin embargo, el modelo no predice directamente las coordenadas (x, y) de las fijaciones en el estímulo, sino que predice mapas de saliencia, que pueden ser usados como densidades de probabilidad para muestrear dichas secuencias. Estas se calculan a partir de características del estímulo a través del tiempo y, por lo tanto, no dependen del historial de "navegación" del usuario. El modelo del estudio calcula las densidades de probabilidad a partir de mezclas gaussianas, de las coordenadas.

Se utiliza la misma metodología que en el estudio anterior, es decir, predecir mapas de saliencia con los cuales se pueda crear las secuencias de coordenadas como una muestra. Sin embargo, no se consideran directamente las características temporales del estímulo, sino que estas se integran implícitamente en las características que presentan los movimientos oculares de los usuarios.

En este caso, para lograr la generación de puntos de fijación, se realizan las siguientes simplificaciones:

- En vez de predecir distribuciones respecto a las fijaciones, se hace en función de los datos de mirada, es decir, todos los puntos registrados por el EyeTracker (fijaciones, sacadas y puntos indefinidos), separando las probabilidades de movimiento y de caracterización de cada punto en cada una de las clases.
- A diferencia de predecir mapas de saliencia para las coordenadas, se calculan las distribuciones como mezclas gaussianas para los **movimientos** en cada coordenada, Δx y Δy dentro del sitio.
- Se modifica el enfoque de ventanas de tiempo usado en el modelo con AOI's, para mejorar los modelos usados en este escenario.
- No se espera predecir los movimientos dinámicos del estímulo (despliegue de menús).

Las simplificaciones anteriores permiten generar resultados que pueden ser tratados para la generación de secuencias de fijaciones, según el planteamiento original. En particular, conociendo el punto inicial, es posible generar las distribuciones a nivel coordenadas a partir de lo calculado a nivel diferencial en cada eje. De aquí en adelante los resultados serán evaluados según considerando lo anterior.

En la ilustración 6.39 se presenta el esquema de solución del modelo. Se estructuran las señales y se crean los conjuntos de validación cruzada al igual que en el modelo anterior. Estos datos estructurados son entregados a un Script principal de Python, con Tensorflow, en el cual se corre el modelo final.

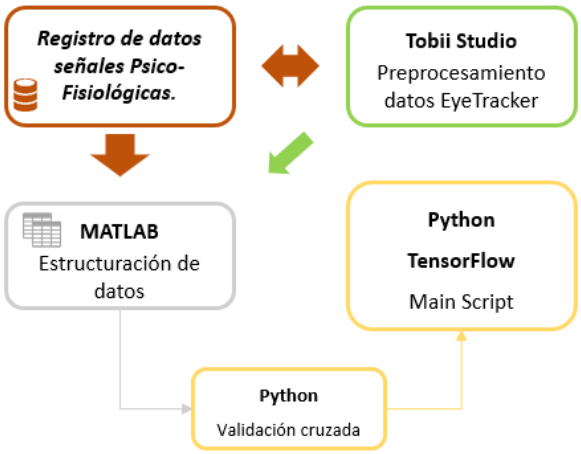


Figura 6.39: Esquema de la solución.

6.4.2. Representación del estímulo

Para comprender en más detalle el problema, en la imagen 6.40 se muestra un registro completo de los movimientos oculares de un usuario en la página web. Aunque se muestra una página estática, en la realidad, los elementos dinámicos de esta hacen que no necesariamente las coordenadas se correspondan con las áreas de interés definidas semánticamente y que se muestran en el fondo.

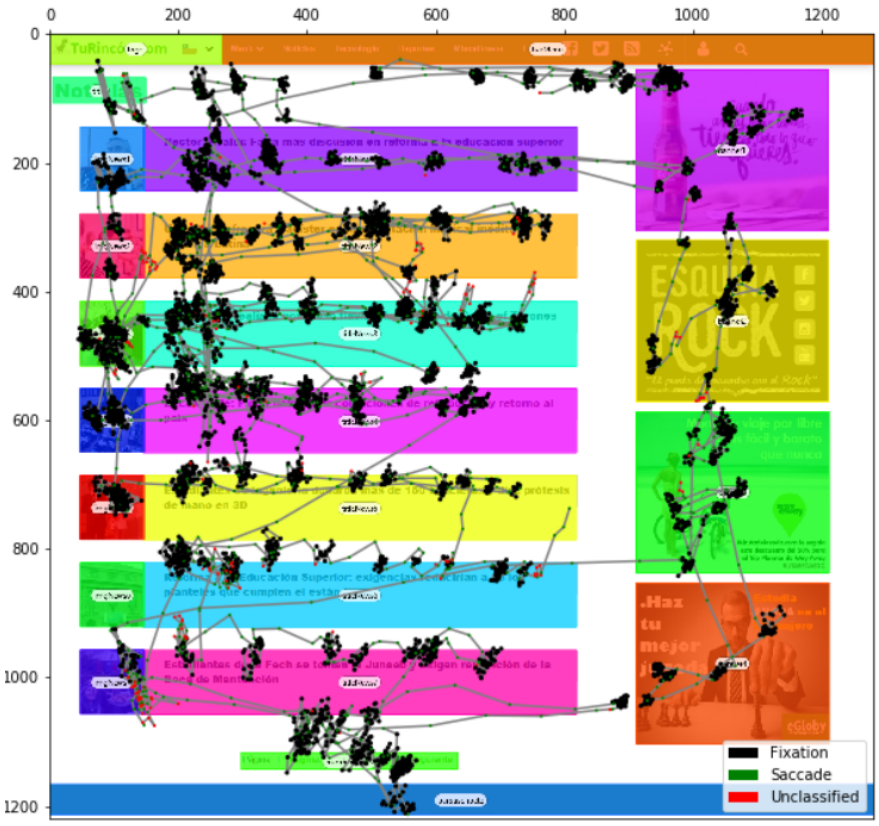


Figura 6.40: Ejemplo de registro completo de datos de mirada para un usuario

Como se puede ver en el gráfico, los datos de mirada se caracterizan por sus coordenadas (x,y) en la pantalla, siendo la esquina superior izquierda de la ventana, el origen. Además, cada uno de los puntos registrados pertenece a una de las tres clases: fijación, sacada o indeterminado, representadas con los colores negro, verde y rojo en el gráfico. Las líneas grises corresponden a la unión de los puntos discretos en el tiempo.

En la realidad, la clasificación de puntos de la mirada en una de las tres categorías anteriores se basa en algoritmos ya validados por el EyeTracker, en el cual se agrupa un conjunto de puntos en lo que es una fijación, por ejemplo. En en este caso, por lo tanto, un conjunto de puntos clasificados pueden pertenecer a la misma fijación. La clasificación de los puntos de los datos de mirada se escapa de este trabajo y se utilizan los algoritmos administrados por el aparato de medición.

6.4.3. Representación de los datos

De los aprendizajes de los modelos usados en el primer escenario, se desprende el hecho de que las ventanas de tiempo de predicción demasiada extensas presentan demasiada incertidumbre, dificultando el ajuste de las redes a los datos de entrenamiento y haciendo difícil su generalización a los datos de test.

Por la razón anterior, se establecen dos parámetros: τ y Δ. El primero corresponde al tiempo de la ventana predecesora a la ventana de tiempo a ser predicha, es decir, la ventana de tiempo de los datos de entrada en el modelo. El segundo parámetro corresponde al largo de la ventana de predicción. Por lo tanto, dado un tiempo inicial t_o se obtiene la "ventana de información" $(t_o - \tau)$ y la "ventana de predicción" $(t_o + \Delta)$.

Para cada ventana de información o predicción se poseen datos de la mirada, los cuales al ser medidos por el EyeTracker con una frecuencia de 125 Hz, cada ventana queda representada por un arreglo de vectores de la forma $x_{\mathrm{i}} = [x^1, x^2, p^1, p^2, p^3]$. Cada uno de estos vectores corresponde a un punto de la mirada, donde (x^1, x^2) corresponden a las coordenadas horizontal y vertical, respectivamente, del punto en la página web, p^1, p^2, p^3 corresponden a la clase fijación, sacada o indefinido, respectivamente, con el valor 1 para la clase activa y 0 para el resto.

6.4.4. Creación conjuntos entrenamiento y test

El primer paso corresponde a la elección de los conjuntos de datos de entrenamiento y testeo. Se divide la cantidad de usuarios en los conjuntos de entrenamiento y testeo para evitar el entrenamiento de la red con información de usuarios y la predicción de los mismos.

Para poder evaluar la predicción realizada por los modelos, se utiliza un enfoque de evaluación k-fold. Por lo tanto se realiza la misma asignación de usuarios a cada fold que en el escenario anterior (véase tabla 6.2).

Para generar ventanas de entrenamiento y test, se crea un tercer parámetro de tiempo δ. Por lo tanto, para un usuario perteneciente a un fold se itera partiendo en el tiempo $t_o = max(\tau, \delta)$ generando las ventanas de información y predicción. Luego se actualiza $t_o = t_o^{(\mathrm{i}-1)} + \delta$, repitiendo este proceso de manera iterativa hasta alcanzar el final de la muestra

para el usuario. La imagen 6.41 esquematiza este proceso en tres iteraciones para un usuario.

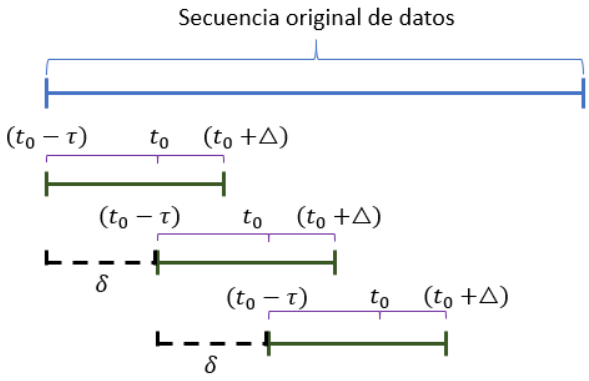


Figura 6.41: Creación de ventanas de información y predicción para un usuario.

6.4.5. Planteamiento formal del modelo

Lo que se busca en una primera instancia es la estimación de distribuciones de probabilidad para los datos de la mirada del usuario. Para ello, se modelan los "movimientos" de la mirada como una mezcla de distribuciones gaussianas bivariadas de parámetros $\mu = (\mu_{x_1}, \mu_{x_2})$, $\sigma = (\sigma_{x_1}, \sigma_{x_2})$ y ρ, correspondientes a la media, desviación estándar y correlación respectivamente. Además, se busca obtener información de la clase a la cual pertenece cada punto, lo cual se modela con una distribución de probabilidad multinoulli (Bernoulli de 3 categorías).

La idea general del modelo es usar una arquitectura que reciba datos presentes en la ventana de información y prediga en cada paso temporal los parámetros de las distribuciones antes mencionadas. Para ello, los datos de coordenadas deben ser llevados a los vectores de desplazamiento $\mathrm{d}x = [\mathrm{d}x^1, \mathrm{d}x^2, p^1, p^2, p^3]$, donde p^1, p^2, p^3 siguen siendo las mismas variables pero $\mathrm{d}x^1, \mathrm{d}x^2$ corresponden a la diferencia en las coordenadas horizontal y vertical de un punto respecto al punto anterior.

El modelo general se piensa como una arquitectura encoder-decoder, en la cual se codifican los datos de la ventana de información en un vector de contexto de largo fijo, para luego decodificarlo en la predicción de las distribuciones de probabilidad. Para lograr generar este resultado, se utilizan las denominadas **mixture density networks (MDN)** en los resultados de redes recurrentes LSTM. En la ilustración 6.42 se muestra lo descrito.

En la ilustración, cada paso temporal genera outputs $(\pi)_t$ correspondientes a los pesos de las S distribuciones gaussianas a ser mezcladas y $(\mu, \sigma, \rho)_t$ correspondientes a las medias y desviaciones estándar de dichas distribuciones. De esta forma, cada output es un vector y_t, que consiste para cada distribución normal $j = 1, \ldots, S$ en la mezcla, en su media $\mu_j = (\mu^j_{x_1}, \mu^j_{x_2})$, desviación estándar $\sigma_j = (\sigma^j_{x_1}, \sigma^j_{x_2})$, la correlación ρ_j y peso π_j. Además, genera los valores para la aplicación de la función softmax que permiten obtener una distribución categórica para la clase de cada punto, $\mathrm{e}_t = (p_1, p_2 \text{ y } p_3)_t$.

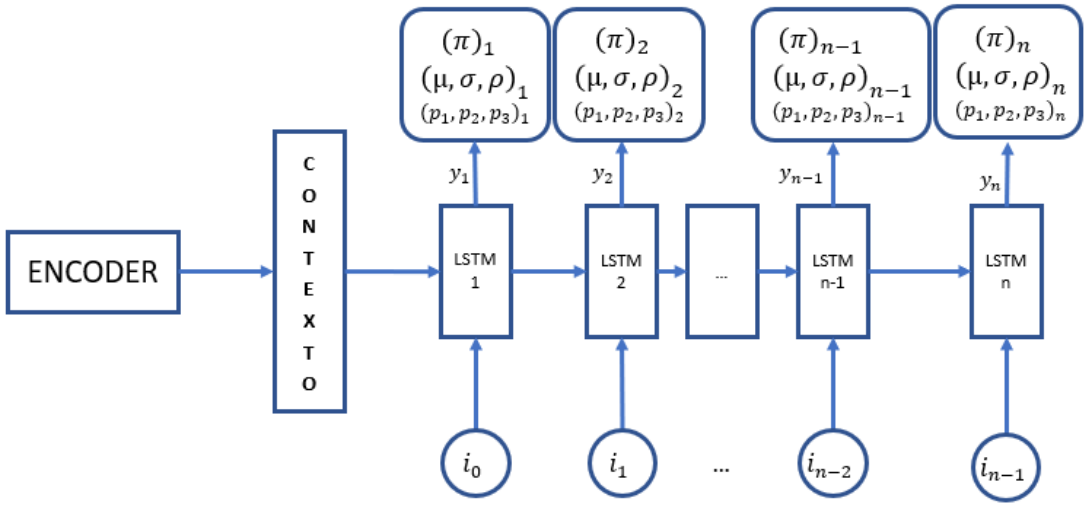


Figura 6.42: Arquitectura general encoder-decoder modelo gaze data

Los outputs entregados por el decoder son predichos por la red de la forma:

$$\hat{y}_t = (\hat{\mathrm{e}}_t, \{\hat{\pi}_t, \hat{\mu}_t, \hat{\sigma}_t, \hat{\rho}_t\}_{j=1}^{M}) = LSTM(\mathrm{i}_{t-1})$$

Dada una cantidad S de distribuciones gaussianas en la mezcla, la red genera un total de $6S+3$ parámetros por paso temporal, ya que μ y σ presentan dos coordenadas (S pesos π, $2S$ valores para μ, $2S$ valores para σ, S valores para ρ y tres para e). Estos output son procesados segun las ecuaciones 6.6 para obtener valores dentro de los rangos correctos.

$$\mathrm{e}_t^{\mathrm{i}} = \frac{exp(\hat{\mathrm{e}}_t^{\mathrm{i}})}{\sum_{\mathrm{i}=1}^{3} exp(\hat{\mathrm{e}}_t^{\mathrm{i}})} \qquad \Longrightarrow \mathrm{e}_t^{\mathrm{i}} \in (0,1), \quad \sum_{\mathrm{i}} \mathrm{e}_t^{\mathrm{i}} = 1 \tag{6.6a}$$

$$\pi_t^j = \frac{exp(\hat{\pi}_t^j)}{\sum_{\mathrm{i}=1}^{S} exp(\hat{\pi}_t^{\mathrm{i}})} \qquad \Longrightarrow \pi_t^j \in (0,1), \quad \sum_{j} \pi_t^j = 1 \tag{6.6b}$$

$$\mu_t^j = \hat{\mu}_t^j \qquad \Longrightarrow \mu_t^j \in \mathbb{R} \tag{6.6c}$$

$$\sigma_t^j = exp(\hat{\sigma}_t^j) \qquad \Longrightarrow \sigma_t^j > 0 \tag{6.6d}$$

$$\rho_t^j = tanh(\rho_t^j) \qquad \Longrightarrow \rho_t^j \in (-1,1) \tag{6.6e}$$

Finalmente, dado los parámetros, la probabilidad de que un punto pertenezca a una clase esta dada por e_t y la distribución para los movimientos de la mirada en $t+1$, x_{t+1} se pueden calcular dado el output anterior y_t mediante las fórmulas:

$$P(\mathrm{d}x_{t+1}|y_t) = \sum_{j=1}^{S} \pi_t^j \mathcal{N}(\mathrm{d}x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j) \tag{6.7}$$

Donde:

$$\mathcal{N}(\mathrm{d}x|\mu,\sigma,\rho) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[\frac{-Z}{2(1-\rho^2)}\right] \tag{6.8a}$$

$$Z = \frac{(\mathrm{d}x^1-\mu_1)^2}{\sigma_1^2} + \frac{(\mathrm{d}x^2-\mu_2)^2}{\sigma_2^2} - \frac{2\rho(\mathrm{d}x^1-\mu_1)(\mathrm{d}x^2-\mu_2)}{\sigma_1\sigma_2} \tag{6.8b}$$

La implementación de los modelos se realiza con el framework Tensorflow. En la sección 8.5 se entregan los cálculos de las derivadas necesarias para ejecutar el método de back-propagation. Sin embargo, Tensorflow trabaja de manera interna para entrenar el modelo, por lo cual dicha formulación no es programada. En particular la programación realizada en este trabajo se basa fuertemente en los códigos del estudio [43], el cual ha sido adaptado a nuestro problema, permitiendo generar modelos con diferentes arquitecturas para la codificación de datos de la ventana de información y decodificación en resultados para las ventanas de predicción.

6.4.6. Procesamiento de datos

Dado que cada ventana de tiempo queda representado por un arreglo de vectores, el primer tratamiento requerido es la interpolación de datos en aquellos vectores en los que se obtienen valores perdidos por el EyeTracker, debido a que en la toma de datos pueden existir pérdidas por pestañeos del usuario, u otros artefactos.

Para la interpolación de los datos se separan los arreglos de datos de las coordenadas x^1 y x^2 y en cada una se realiza una interpolación lineal entre los datos correctamente adquiridos en los intervalos en los que se encuentran pérdidas de datos. A estos puntos generados se les asigna una clase de dato indefinido. En la imagen 6.43 se observan los datos de una ventana de tiempo y luego de la interpolación, los resultados se muestra en 6.44. Se observa la existencia de un mayor número de puntos rojos entre los puntos verdes y negros.

Figura 6.43: Datos de mirada en una ventana de tiempo.

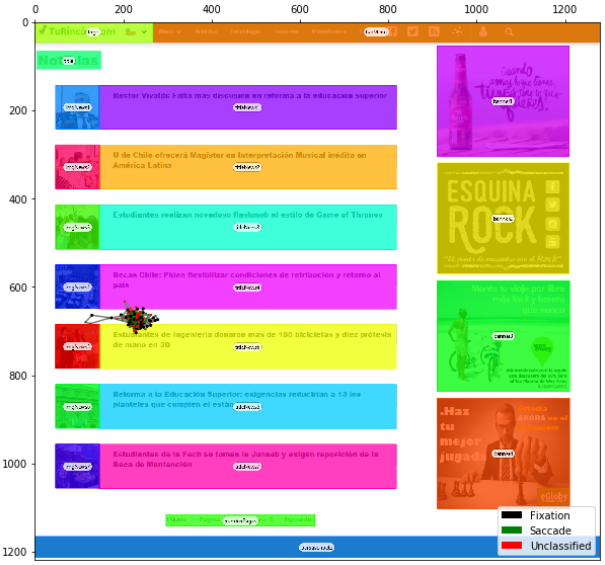


Figura 6.44: Datos de mirada interpolados para la misma ventana

Posteriormente, dado el largo de las secuencias de datos en las ventanas de información M y predicción N, se corrigen los largos para las muestras que presenten desalineación con esta cantidad de datos en la secuencia. Para ello se recortan los datos sobrantes o se agregan las ultimas coordenadas registradas en cada eje al final de la secuencia, en el caso de falta de datos.

Dado que el modelo en realidad no busca la predicción de distribuciones de coordenadas en las que el usuario posa su mirada, sino que del desplazamiento de esta, el segundo procesamiento es el cálculo de $\mathrm{d}x^1, \mathrm{d}x^2$ como la diferencia entre cada coordenada en los datos del paso temporal i y la coordenada del paso i − 1. Además, se aplica la división por un factor $f = (f_{\mathrm{d}x_1}, f_{\mathrm{d}x_2})$ calculadas como la desviación estándar en los conjuntos de entrenamiento y almacenadas para ser aplicados en el procesamiento de los conjuntos de test. Este factor se aplica para corregir la desviación estándar de los desplazamiento dejándolo con valor 1 en el conjunto de entrenamiento. No se hacen ajustes en la media ya que esta es cercana a cero en la mayoría de los casos.

6.4.7. Implementación del modelo

En esta sub-sección se implementa el modelo planteado, denominado "Modelo Gaze data - Gaze data". Posteriormente se utilizan otras arquitecturas para utilizar lo aprendido a nivel píxel para la predicción de secuencias de AOI y del próximo AOI en la sub-sección 6.4.8.

Modelo Gaze Data - Gaze Data

En un primer modelo utilizado corresponde a una arquitectura Encoder-Decoder, mostrado en la ilustración 6.45.

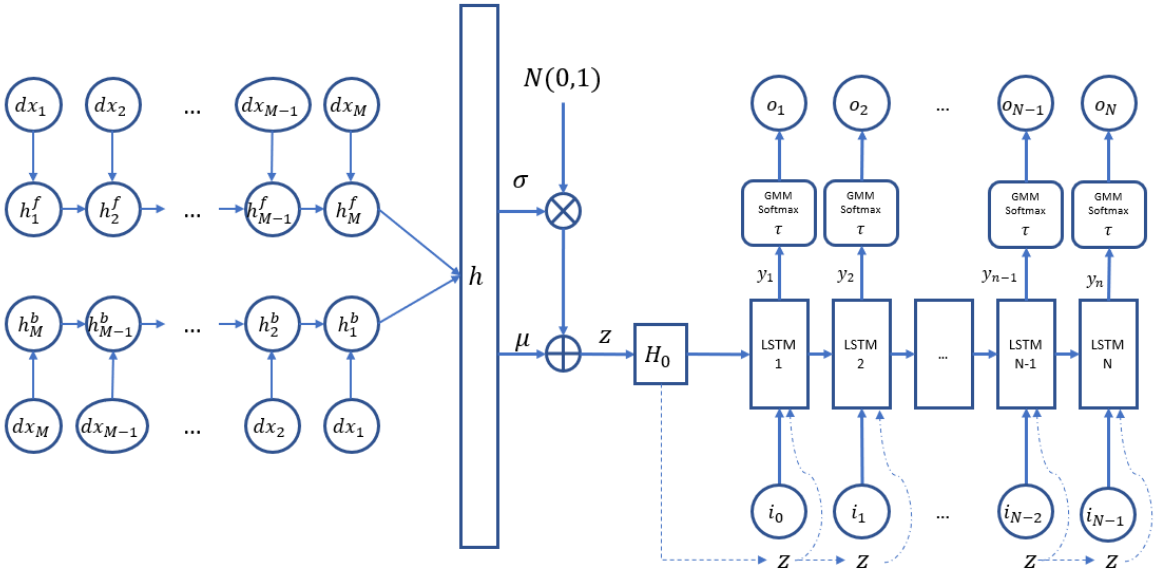


Figura 6.45: Arquitectura encoder-decoder primer modelo gaze data

En la parte izquierda de la ilustración se muestra una red LSTM bidireccional que recibe los vectores $\{\mathrm{d}x_1, \ldots, \mathrm{d}x_M\}$, donde cada $\mathrm{d}x_\mathrm{i}$ es un vector de la forma, $\mathrm{d}x_\mathrm{i} = [\mathrm{d}x^1, \mathrm{d}x^2, p^1, p^2, p^3]$. Esta red se basa en la arquitectura propuesta en el estudio [43] usada para aprender patrones de dibujos.

Esta red presenta una cantidad de M pasos temporales que son codificados en el vector h. Este vector es utilizado para generar una distribución gaussiana de media μ y desviación estándar σ, con tamaño como parámetro del modelo. Esta distribución captura los patrones

de movimiento de los usuarios en cuanto a su mirada. Esta normal es utilizada para generar el estado oculto inicial del decoder H_0 a partir de la función de activación $tanh()$ aplicada a un muestreo de la distribución gaussiana estimada z.

El decodificador de la red, corresponde a una red LSTM en la que en el primer paso temporal se le entrega un desplazamiento nulo ($\mathrm{d}x^1 = 0$ y $\mathrm{d}x^2 = 0$), y $p_j = 1$ para la clase j activa en el último punto entregado en el encoder (para mantener continuidad de información), en formato $\mathrm{d}x_\mathrm{i}$, concatenado con un muestreo z de la distribución gaussiana generada por el codificador.

La concatenación con la distribución gaussiana generada por el encoder permite entregar información directamente a cada paso acerca de la "forma" en que los movimientos están siendo ejecutados por el usuario. Esto se representa en la imagen con las lineas punteadas a cada paso temporal t y a las celdas LSTM.

Es importante notar que este modelo es idéntico a lo expuesto en [43], adaptado al problema presente a partir de la definición de los valores $[p^1, p^2, p^3]$ y una pequeña modificación en la función de pérdida, ya que en este caso, las ventanas poseen un largo fijo y por lo cual las redes del encoder y decoder, también.

Para entrenar el modelo, se entregan los datos de la ventana τ y de la ventana δ. De esta manera en cada paso temporal de la red en el decodificador recibe los valores correctos para su estimación de parámetros. Aunque en la realidad, la red de decodificación, en cada paso temporal, recibe la estimación realizada en el paso anterior.

Antes de generar cada output temporal o_t se agrega un parámetro de temperatura τ el cual permite controlar el nivel de aleatoriedad de las distribuciones de probabilidad generadas. Este parámetro es aplicado a los parámetros de las redes gaussianas y a los parámetros de la distribución categóricas según las ecuaciones 6.9. Este valor esta en el rango $(0, 1)$ y $\tau \to 0$ implica un modelo completamente determinista.

$$\hat{\mathrm{e}}_t \to \frac{\hat{\mathrm{e}}_t}{\tau}, \hat{\pi}_t \to \frac{\hat{\pi}_t}{\tau}, \hat{\sigma_1}^2 \to {\sigma_1}^2\tau, \hat{\sigma_2}^2 \to {\sigma_2}^2\tau \tag{6.9}$$

Para evitar el sobre-entrenamiento de la red, se agregan a las celdas del encoder y del decoder dropout, ambas con probabilidad de mantención de 0.9.

La función de pérdida para este primer modelo esta dada por la ecuaciones 6.10. La pérdida de reconstrucción L_R busca maximizar la log-verosimilitud de la distribución generada en los datos de entrenamiento. Esta compuesta por la perdida de los términos de desviación $(\mathrm{d}x_1, \mathrm{d}x_2)$ y la pérdida por los términos de clase perteneciente en cada punto e_t, presentes en las ecuaciones 6.10a,6.10b. El segundo corresponde a la divergencia de Kullback Leibler medida como la diferencia entre la distribución del vector latente z y un vector gaussiano IID

con media cero y desviación estándar unitaria, mostrado en la ecuacion 6.10c

$$L_R = -\frac{1}{N_{train}} \sum_{\mathrm{i}=1}^{N_{train}} \log\Big(\sum_{j=1}^{S} \pi_t^j \mathcal{N}(\mathrm{d}x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j\Big) \tag{6.10a}$$

$$- \frac{1}{N_{train}} \sum_{\mathrm{i}=1}^{N_{train}} \sum_{j=1}^{3} p_j \log\left(\hat{p}_j\right) \tag{6.10b}$$

$$L_{KL} = -\frac{1}{2N_z}(1 + \hat{\sigma} - \mu^2 - exp(\sigma)) \tag{6.10c}$$

$$TotalLoss = L_R + w_{KL} L_{KL} \tag{6.10d}$$

La función de pérdida total 6.10d está dado por L_R más L_{KL} ponderado por w_{KL}, parámetro que le otorga mayor o menor importancia a este ultimo término.

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds. Se aplica 10 iteraciones, en las cuales para cada una se deja un fold como datos de testeo y el resto como datos de entrenamiento. Para cada fold, se generan batches para los datos de entrenamiento, cada uno con una cantidad de muestras dadas como parámetro. El entrenamiento se realiza por un número definido de epochs (o iteraciones) y en cada iteracion, el modelo recibe el siguiente batch, repitiendo el primer bacth cada vez que se alcanza el último. Además en cada iteración se aplica un decay rate de 0.999 a una tasa de aprendizaje de 0.001, y un clip de gradiente de 1.

En primera instancia, se corre el modelo con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 1$ segundo y $\delta = 3$ segundos, con los demás parámetros:

- tamaño de los batches : 50
- probabilidad dropout encoder : 0.9
- probabilidad dropout decoder : 0.9
- w_{KL} : 0.01
- número de mezclas : 20
- epochs : 1000
- número de neuronas encoder: 128
- número de neuronas decoder: 128
- tamaño vector z: 128
- temperatura : 1

Se puede apreciar que en cada fold se tiene un comportamiento bastante similar. En las imagen 6.46 se observa el costo de reconstrucción en el proceso de entrenamiento de la red a través de las iteraciones. Se puede ver su descenso desacelerado. Por otro lado, en la figura 6.47 se ve que el costo KL empieza a aumentar rápidamente para luego converger a un valor cercano a 3. Como se escoge un valor de w_{KL} pequeño, el costo total sigue la tendencia del costo de reconstrucción.

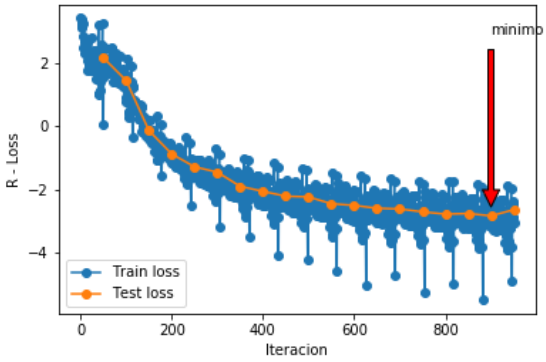


Figura 6.46: Costo de reconstrucción por epoch, modelo 1

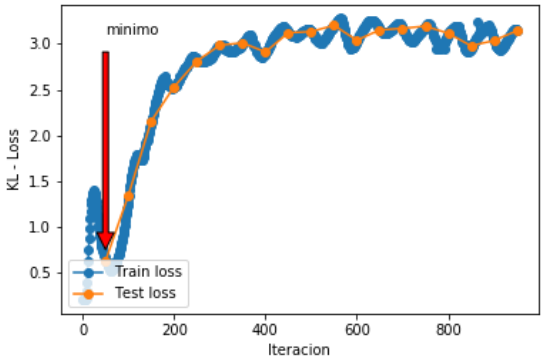


Figura 6.47: divergencia de Kullback Leibler por epoch, modelo 1.

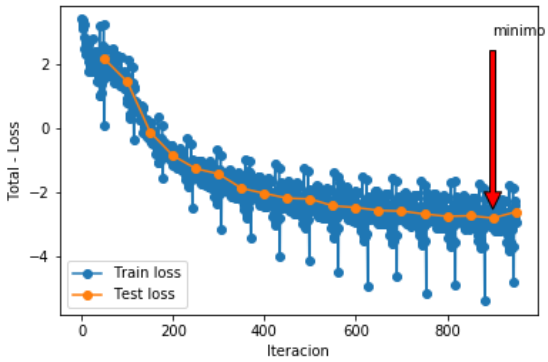


Figura 6.48: Costo total por epoch, modelo 1

Posteriormente se ejecuta el mismo modelo, variando el número de neuronas en el encoder y en el decoder a 256. Sin embargo, los resultados presentan la misma tendencia. Al aplicar el modelo en el conjunto de testeo se obtienen las métricas en promedio luego de las k iteraciones:

Modelo de 128 neuronas:

- L_R entrenamiento: -2.87
- L_{KL} entrenamiento: 2.96
- L_{Total} entrenamiento: -2.78
- L_R testeo: -2.90
- L_{KL} testeo: 2.97
- L_{Total} testeo: -2.87

Modelo de 256 neuronas:

- L_R entrenamiento: -4.08
- L_{KL} entrenamiento: 3.05
- L_{Total} entrenamiento: -3.99
- L_R testeo: -3.62
- L_{KL} testeo: 3.01
- L_{Total} testeo: -3.59

Se puede observar que aunque ambos modelos se logran entrenar correctamente, el modelo con más neuronas permite mejores resultados, es decir, la función de pérdida se ve minimizada. Sin embargo, el tiempo de ejecución de cada modelo en una muestra de testeo supera el tiempo Δ. A pesar de esto, se prefiere el uso del modelo de 256 neuronas. Con este modelo

entrenado, es posible generar muestreos de puntos para una ventana de tiempo. Para ello se siguen dos métodos principales:

En primer lugar, se genera una distribución para los movimientos de la mirada en t dado el punto real en $t-1$. Al saber el punto original en cada paso anterior, se espera que haya menos dispersión en los resultados. En la imagen 6.49 se observa una ejemplo siguiendo este método en una ventana de tiempo.

Posteriormente se prueba un enfoque de predicción en la totalidad de la ventana de predicción dado los datos de la ventana de información. Aquí, las distribuciones calculadas en un tiempo t están dado con el input correspondiente a una muestra generada de la distribución calculada en el tiempo $t-1$. En particular, para la distribución en $t-1$ se utiliza el método greedy para la selección de una muestra $\mathrm{d}x_{\mathrm{i}}^{(t)}$ a partir de la máxima probabilidad. Se espera que este método al poseer resultados basados en el muestreo en cada paso temporal, presente mayor aleatoriedad. En la imagen 6.50 se muestra un ejemplo para la misma ventana de tiempo que en la imagen 6.49 pero utilizando este método.

Figura 6.49: Distribuciones generadas a partir de puntos reales para una ventana de tiempo.

Figura 6.50: Distribuciones generadas a partir de puntos muestreados para una ventana de tiempo.

En ambos métodos, las distribuciones calculadas en cada paso temporal son agregadas para obtener una distribución correspondiente a la totalidad de la ventana de predicción. Además, los resultados son llevados nuevamente a coordenadas de la página a través de las transformaciones inversas a las realizadas en el procesamiento de datos (aplicar el factor sobre desplazamientos en la mirada).

Para evaluar estas distribuciones se utilizan métricas de mapa de calor. Para ello, en cada ventana de tiempo de predicción se generan 4 imágenes. En 6.51 se observa la página web en su totalidad en negro, y sobre esta puntos blancos correspondientes a las fijaciones llevadas a cabo en dicho periodo de tiempo por el usuario. En la imagen 6.52 se agregan los demás

puntos de mirada, registradas por el EyeTracker, es decir, se agregan las sacadas y los puntos indefinidos.

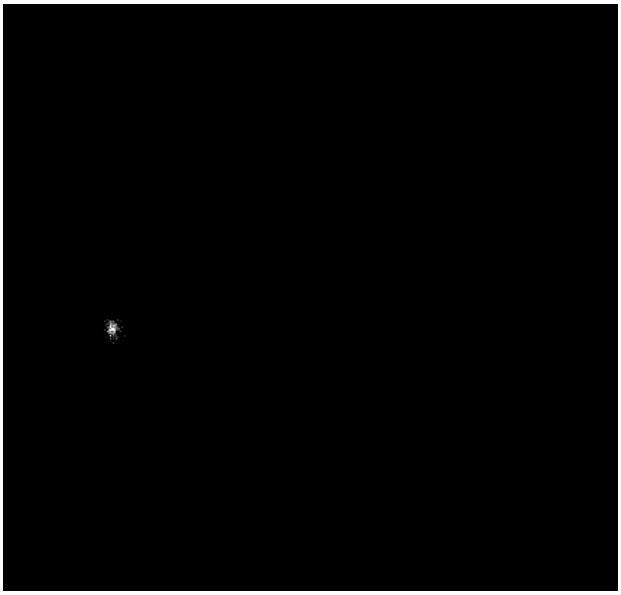


Figura 6.51: Fijaciones realizadas en una ventana de tiempo de predicción (ground truth).

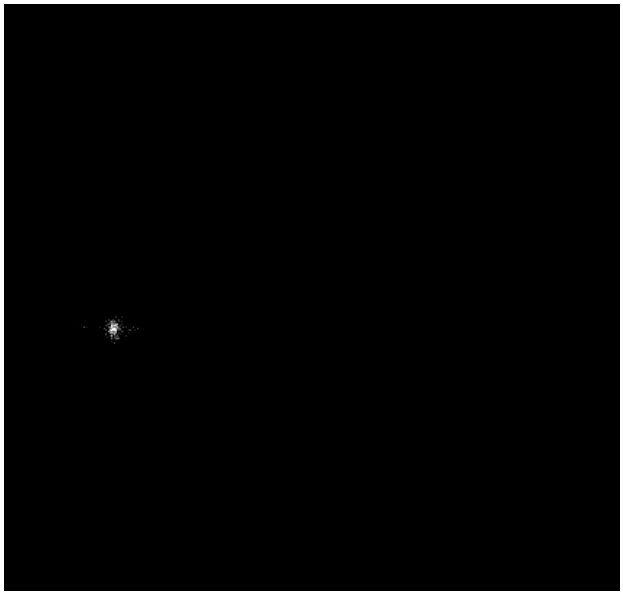


Figura 6.52: Datos de mirada (fijaciones, sacadas y puntos indefinidos) para una ventana de tiempo.

Figura 6.53: Distribución de probabilidades creadas a partir de puntos reales para una ventana de tiempo de predicción.

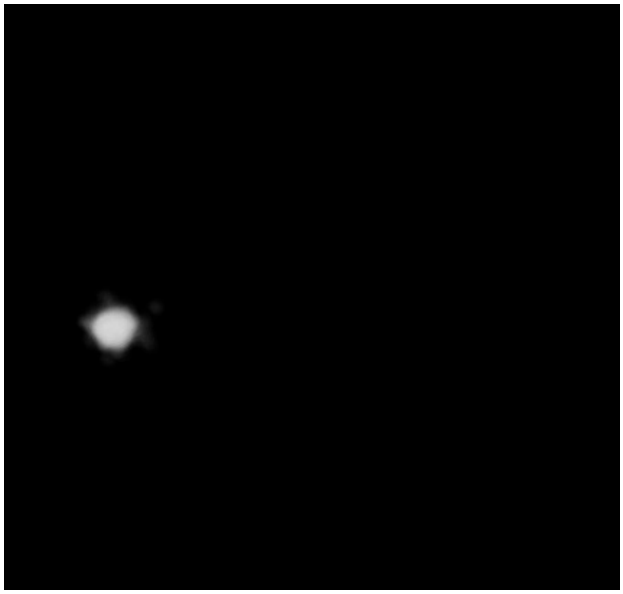


Figura 6.54: Distribución generada a partir de puntos muestreados para una ventana de tiempo.

Las distribuciones mostradas en las figuras 6.50 y 6.49 son llevadas a las mismas escalas de colores, y se muestran en las imágenes 6.53 y 6.54 respectivamente. Posteriormente, las

imágenes son re-dimensionadas para aliviar la complejidad computacional. Finalmente estas imágenes son tratadas con las métricas programadas en [21].

Las métricas de evaluación se utilizan en estudios como [18, 56], al igual que lo realizado en [10]. Son definidas a continuación, donde S representa los resultados de saliencia obtenidos (ilustraciones 6.53 y 6.54), y G representa un mapa de saliencia generado a partir de los datos de fijaciones de los usuarios (ilustraciones 6.51 y 6.52):

Area Under Curve ROC (AUC): Es el área bajo la curva ROC (Receiver Operating Characteristic). La curva ROC es usada para la evaluación de clasificadores mediante diferentes valores de threshold. El saliency map generado es usado como clasificador, donde para cada punto del estímulo con un valor de saliency mayor al threshold se le asigna una fijación. Luego se comparan estos resultados con las fijaciones reales.

Variando los valores de threshold se va generando la curva, al graficar las tasas de "false positives" versus "true positive". La fracción de área bajo la curva es la medida utilizada. La predicción perfecta corresponde a un valor de $AUC = 1$.

Existe una variante, llamada **AUC-Judd**. En esta, para valor de threshold, la tasa de "true positives" se calcula como el ratio entre los valores "true positives", es decir, valores del mapa de saliencia que con valor mayor al threshold en los píxeles categorizados como fijaciones, y el total de fijaciones. La tasa de "false positive" es el ratio entre "false positives" y el total de pixeles en el mapa de saliencia con valores superiores al threshold, pero que no son considerados como fijaciones.

Otra variante corresponde al denominado **AUC-Borji**, el cual usa un muestreo aleatorio uniforme de los píxeles como negativos y define los valores del saliency map sobre el threshold en estos pixeles como falsos positivos.

Normalized Scanpath Saliency (NSS): Es una correspondencia entre los mapas de saliencia y los datos reales (ground truth), calculado como la saliencia promedio normalizada en las posiciones de las fijaciones [22]. A diferencia de los valores de AUC, los valores absolutos de saliencia son usados en el cálculo de esta métrica.

NSS es sensible a los falsos positivos y a transformaciones monotónicas (transformar un conjunto de números en otro que preserve el orden del conjunto original). Por otro lado, dado que los valores de saliencia son restados, es una métrica invariante a las transformaciones lineales tales como los contrastes. Dado un mapa S y un mapa binario de fijaciones G^f, se calcula:

$$NSS(S, G) = \frac{1}{N} \sum_{\mathrm{i}} \bar{S}_{\mathrm{i}} \times G^f_{\mathrm{i}}$$

Donde N es la cantidad de píxeles considerados como fijaciones, i indica un píxel particular, y se ha normalizado el mapa original S para que tenga una media cero y desviación estándar de 1:

$$N = \sum_{\mathrm{i}} G^f_{\mathrm{i}} \qquad y \qquad \bar{S} = \frac{S - \mu(S)}{\sigma(S)}$$

Un valor de cero indica que el mapa es completamente aleatorio, un valor positivo indica una correlación entre los puntos de fijaciones reales y la saliencia calculada y un valor negativo

indica que la saliencia de los puntos reales es pequeña (mal resultado). un valor positivo c, corresponde a que las fijaciones caen en sectores del mapa de saliencia con valor c veces sobre el promedio.

Linear Correlation Coefficient (CC): La correlación lineal mide la fuerza en la relación lineal entre dos variables

$$CC(G,S) = \frac{\sum_{x,y}(G(x,y)-\mu_G)(S(x,y)-\mu_S)}{\sqrt{\sigma_G^2\sigma_S^2}}$$

Donde μ y σ corresponden a la media y desviación estándar de de los valores en los mapas. CC estén en el rango [-1,1] y para valores cercanos a 1/-1 existe una mayor relación enetre ambas variables.

Similarity Score (SIM): Es una medida de que tan similar son dos distribuciones. Después de que cada distribución es escalada para que sume 1, se suman los valores mínimos para cada punto en las distribuciones. Un valor de 1 indica que ambas distribuciones son iguales. Un valor de 0 indica que las distribuciones no se superponen y son completamente diferentes.

$$S = \sum_{\mathrm{i},j} \mathrm{mín}(P_{\mathrm{i},j}, Q_{\mathrm{i},j}) con \sum_{\mathrm{i},j} P_{\mathrm{i},j} = \sum_{\mathrm{i},j} Q_{\mathrm{i},j} = 1$$

Finalmente, los resultados promedio dentro de las muestras de los 10 folds evaluados usando validación cruzada se muestran en la tabla 6.17. Para la construcción de esta tabla, se han calculado las métricas AUC-Judd, AUC-Borji y NSS a partir de la comparación entre los datos de los mapas de datos de mirada (véase 6.52) y las distribuciones generadas a partir de puntos muestreados(véase 6.54) y a partir de los datos de fijaciones (véase 6.51) con dicha distribución. Al mismo tiempo, para evaluar la diferencia entre las distribuciones generadas a partir de una muestra de puntos(6.54) y las generadas a partir de conocer los puntos reales dentro de la ventana de tiempo de predicción (6.53) se han usado las métricas de CC y SIM.

Tabla 6.17: Resultados métricas Saliency map para modelo de 128 neuronas en encoder.

	NSS		**AUC Judd**		**AUC Borji**		**CC**	**SIM**
Fold	**gaze-pred**	**fixation-pred**	**gaze-pred**	**fixation-pred**	**gaze-pred**	**fixation-pred**	**pred-dist**	**pred-dist**
1	11,4802	17,3195	0,9811	0,9995	0,9063	0,9987	0,8761	0,6907
2	12,0851	16,8782	0,9899	0,9995	0,9619	0,9991	0,9176	0,7395
3	11,7547	17,2196	0,9751	0,9997	0,9298	0,9992	0,8581	0,6789
4	11,0441	17,7417	0,9492	0,9997	0,8844	0,9994	0,7871	0,5788
5	11,7460	16,9889	0,9856	0,9997	0,9375	0,9991	0,8585	0,6805
6	11,4116	18,6434	0,9426	0,9997	0,8820	0,9988	0,8685	0,6640
7	11,3963	18,1097	0,9611	0,9997	0,9003	0,9991	0,8664	0,6510
8	11,9202	17,5664	0,9923	0,9995	0,9462	0,9993	0,8737	0,6611
9	11,7353	17,3290	0,9901	0,9996	0,9326	0,9993	0,9019	0,7294
10	11,2118	16,5682	0,9638	0,9995	0,9347	0,9989	0,8377	0,6329
Promedio	11,579 (0,325)	17,436 (0,611)	0,973 (0,018)	1,000 (0,000)	0,922 (0,027)	0,999 (0,000)	0,865 (0,035)	0,671 (0,046)

Es importante notar que las métricas anteriores en general son utilizadas para comparar mapas de saliencia con mapas de ground truth de las fijaciones llevadas a cabo por los usuarios. Sin embargo, en este trabajo se han utilizado para la comparación a nivel "gaze data", es decir, se consideran una aglomeración de puntos relacionados a una fijación como un conjunto de puntos independientes en vez de un sólo punto. Por esta razón, la figura de ground truth presenta una mayor cantidad de puntos que lo comúnmente presentado en la literatura.

De las métricas se puede apreciar lo siguiente. Primero, al comparar los mapas de saliencia generado a partir de los puntos reales, lo que se está haciendo es utilizar los verdaderos puntos para generar distribuciones de los siguientes movimientos alrededor de estos. Por lo tanto, por definición, sus métricas de NSS, AUC Judd y AUC Borji, tenderán a ser buenas (por definición los puntos caerán dentro de las zonas con más saliencia). Por esta razón no se agregan dichas métricas en los resultados reportados.

Por otro lado, las distribuciones generadas a partir de la predicción de puntos en la ventana de predicción generan un nuevo punto en cada paso temporal (el cual no necesariamente corresponde a un punto real del usuario), alrededor de este se generan distribuciones de movimiento y se escoge un nuevo punto para repetir el procedimiento. Se espera que a lo largo de la secuencia, la distribución sea más dispersa, lo que se evidencia al comparar las imágenes 6.54 con 6.53. Sin embargo, al ser una ventana de tiempo pequeña, la suma total de los movimiento debe generar una distribución de probabilidades donde su masa se presente bastante cercana a los puntos reales. Siguiendo esta idea, las métricas de NSS y AUC nos demuestran los buenos resultados obtenidos. Se observa que las métricas usando los datos de mirada tienden a ser menores que al considerar sólo los puntos catalogados como fijaciones, debido a que los puntos de indefinición o de sacadas tienden a presentar más dispersión.

Aunque los resultados de saliencia son buenos, es bueno comparar los mapas generados con los mapas de saliencia creados a partir de puntos reales. De la métrica CC se observa una fuerte correlación entre ambas distribuciones (superior al 80 %). De la métrica de similitud se observa que en términos de histograma, los mapas comparados píxel a píxel tienden a ser similares (SIM mayor a 0.5), sin embargo, se reporta una menor correlación píxel a píxel.

En conclusión, en ventanas de tiempo de predicción pequeñas, en este caso de un segundo, los mapas de saliencia generados a partir de estimación de desplazamientos de la mirada tienden a ser representativos de los datos de mirada reales. Por lo tanto se espera que la información adquirida por el modelo sea de utilidad para estimar las probabilidades de que un usuario visite cada una de las áreas de interés definidas.

Para finalizar, es importante notar las ventajas y debilidades de este modelo. Se puede apreciar que el modelo logra calcular distribuciones en ventanas de tiempo de menor tamaño. Sin embargo, se observa mayor divergencia al querer calcular la distribución en una ventana de predicción completa. Al darle poco peso al factor de divergencia KL en la función de pérdida, esta tiende aumentar, sin embargo se mantiene en un error acotado. Por lo tanto se espera que al variar el término w_{KL} a un mayor valor, la red intente mejorar estos resultados en su proceso de entrenamiento.

Los resultados de este modelo dan a pensar en que este tipo de información estimada

puede ser de mayor utilidad en la predicción de áreas de interés, por lo cual en la siguiente sección se utilizará este enfoque para la estimación de resultados similares a los buscados en el escenario uno.

6.4.8. Modelos Gaze Data para la predicción de AOI's

Modelo predicción de AOI's temporal

Para predecir las áreas de interés alcanzadas por el usuario en una ventana de tiempo de predicción, primero se definen estas al igual como se hizo en el primer escenario, tal cómo se muestra en la figura 6.1. Notar que como se está trabajando a escala más reducida, se busca la continuidad de los datos y por lo tanto no se elimina el AOI_6 correspondiente a la ventana de registro. Al mismo tiempo, no se hace la simplificación a 4 AOI's. Siguiendo el ejemplo anterior, en el cual se hacen predicciones en diferentes pasos temporales, en esta ocasión se busca la predicción de alcances de AOI's en tiempos discretos. Dado que se quiere conservar la estructura del modelo anterior, en un primer intento se desarrolla una arquitectura que permita predecir los movimientos de la mirada y al mismo tiempo los alcances de los AOI's.

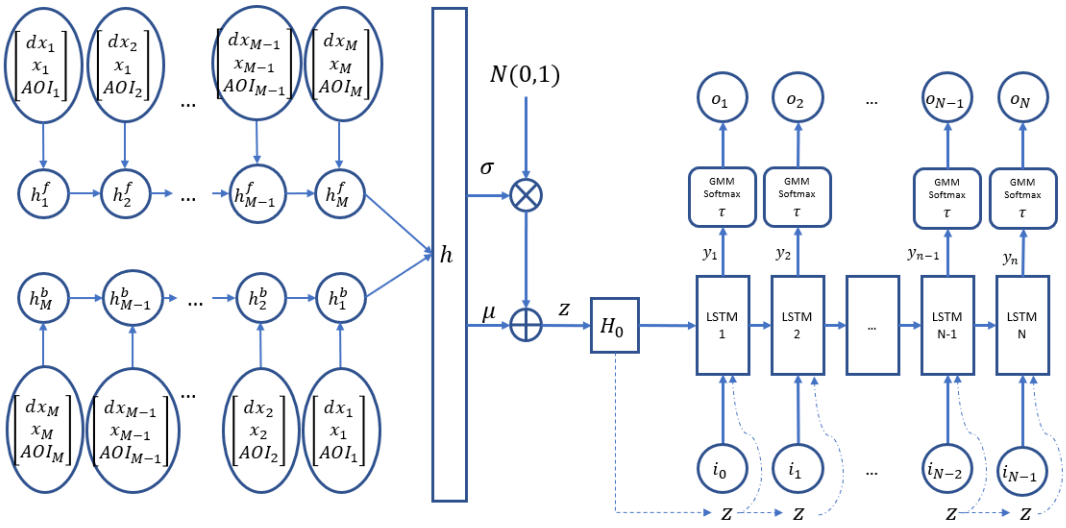


Figura 6.55: Arquitectura predicción AOI's temporal.

En este modelo, se puede observar que los inputs para el encoder corresponden a la concatenación de los vectores $\mathrm{d}x_\mathrm{i}$ más el vector AOI_i que es la versión one-hot del AOI alcanzado en dicho paso temporal. Es decir, es un vector de largo 7, correspondiente a los 6 AOI's definidos más el cero. El vector presenta soló ceros a excepción de un 1 en el área de interés alcanzada. Si no se alcanza ninguna, el 1 corresponde al AOI simbólico cero. Además se entrega como input las coordenadas x_i directamente.

El razonamiento detrás de los inputs escogidos es, en primer lugar, entregar información de los patrones de movimiento del usuario ($\mathrm{d}x_\mathrm{i}$), pero además las zonas en las cuales se encuentra posando la mirada el usuario (x) y al mismo tiempo si estos movimientos han implicados fijaciones en los AOI's. Se espera que esta información pasada revele los patrones

de movimiento futuros y las coordenadas en las que se encuentra el usuario, lo cual puede ser usado indirectamente para conocer en mayor detalle las AOI's en las cuales podría haber fijaciones.

La segunda diferencia corresponde al decoder, el cual en cada paso temporal entrega un output que permite encontrar los parámetros de las distribuciones gaussianas para los movimientos de la mirada, pero también los parámetros $\hat{a}$ que permiten encontrar una distribución para el AOI alcanzado en el paso temporal a través de la formula 6.11.

$$a_t^{\mathrm{i}} = \frac{exp(\hat{a}_t^{\mathrm{i}})}{\sum_{\mathrm{i}=1}^{N_{AOI}+1} exp(\hat{a}_t^{\mathrm{i}})} \implies a_t^{\mathrm{i}} \in (0,1), \quad \sum_{\mathrm{i}} a_t^{\mathrm{i}} = 1 \tag{6.11}$$

Por lo tanto, además de lo anterior, se agrega un nuevo término a la función de pérdida, según la formula 6.12.

$$L_{AOI} = -\frac{1}{N_{train}} \sum_{\mathrm{i}=1}^{N_{train}} \sum_{j=1}^{N_{AOI}+1} A_j \log(a_j) \tag{6.12a}$$

$$TotalLoss = L_R + w_{KL} L_{KL} + L_{AOI} \tag{6.12b}$$

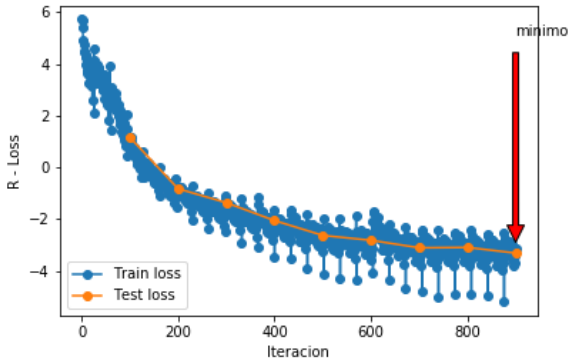


Figura 6.56: Costo de reconstrucción por epoch, modelo 2

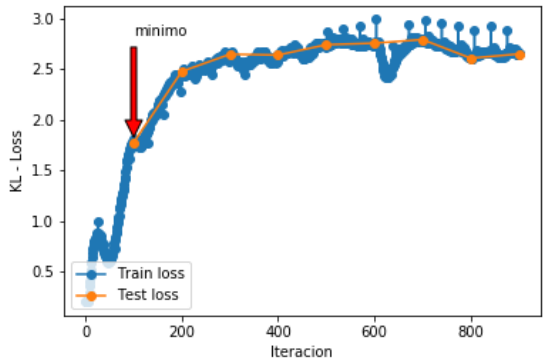


Figura 6.57: divergencia de Kullback Leibler por epoch, modelo 2.

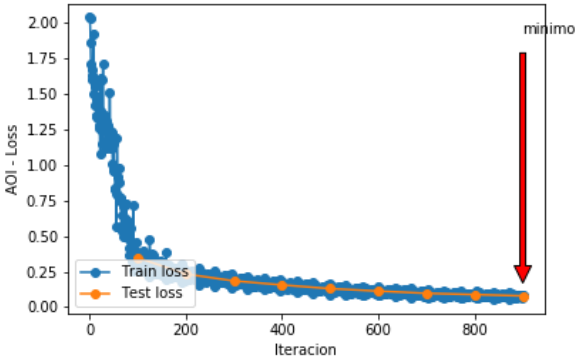


Figura 6.58: Costo softmax AOI por epoch, modelo 2

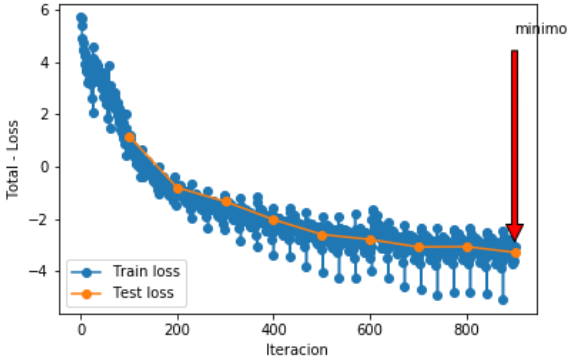


Figura 6.59: Costo total por epoch, $L_{Total} = L_R + w_{KL} L_{KL} + L_{AOI}$, modelo 2

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds. En primera instancia, se corre el modelo con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 1$ segundo y $\delta = 3$ segundos, con los demás parámetros:

- tamaño de los batches : 100
- decay rate: 0.999
- tasa de aprendizaje: 0.001
- clip de gradiente: 1
- probabilidad dropout encoder : 0.9
- probabilidad dropout decoder : 0.9
- w_{KL} : 0.01
- número de mezclas : 20
- epochs : 1000
- número de neuronas encoder: 256
- número de neuronas decoder: 256
- tamaño vector z: 128
- temperatura : 1

Se puede apreciar que en cada fold se tiene un comportamiento bastante similar. En la imagenes 6.56 y 6.58 se observa que los costos de los datos de desplazamiento de la mirada y de la predicción de AOI's disminuye a lo largo de las iteraciones de entrenamiento. Sin embargo, al igual que en el modelo anterior la divergencia de KL aumenta hasta un valor cercano a 3.

Aplicando el modelo a los conjuntos de test de los diferentes folds, se obtienen las siguientes métricas promedio:

- L_R entrenamiento: -3.41
- L_{KL} entrenamiento: 2.68
- L_{AOI} entrenamiento: 0.08
- L_{Total} entrenamiento: -3.33
- L_R test: -3.33
- L_{KL} test: 2.68
- L_{AOI} test: 0.07
- L_{Total} test: -3.31

De los resultados, se puede concluir que el modelo logra aprender los patrones de predicción de AOI's en los datos de entrenamiento, de manera lenta. Al mismo tiempo, dado que las funciones de pérdida tanto en los conjuntos de entrenamiento como de testeo parecen cercanos, se tiende a pensar que el modelo es generalizable a este último conjunto. Para asegurar esto, se prueba el modelo en la generación de muestras a partir de datos de diferentes ventanas de información.

En este modelo, solo se requiere la estimación de un resultado, por lo cual la generación de muestras se puede hacer de dos maneras: una selección del AOI que maximice la probabilidad según el modelo (método argmax) o la selección a partir de la distribución de probabilidad según la distribución sobre las posibles alternativas calculadas por la red.

El método de muestreo a partir de las distribuciones se puede aplicar de diferentes maneras, por lo cual, su medidas puede quedar representada por el valor L_{AOI} reportado anteriormente.

En conclusión el aprendizaje del modelo es generalizable al conjunto de testeo, sin embargo, no es suficiente para obtener buenos resultados en las predicciones de la ventana de tiempo de predicción. Por lo tanto, en un siguiente modelo se aplica este mismo procedimiento a la predicción de sólo el siguiente AOI.

Debido a lo anterior, se piensa también que los datos de entrada (ventana de información) no aportan lo suficiente para una predicción favorable, por lo cual más adelante se intenta obtener mejores resultados agregando datos de EEG a la red.

Modelo predicción del próximo AOI

Este modelo se basa en el anterior, sin embargo, los outputs pasan a ser sólo el próximo AOI visitado por el usuario en la ventana de tiempo de predicción. No se busca la predicción de las distribuciones de movimiento de los datos de mirada porque el primer modelo ejecutado demuestra buenos resultados en este ámbito. Por lo tanto, en la ilustración 6.60 se muestra que el decoder solo utiliza la función softmax para la generación de distribuciones sobre el próximo AOI. Es decir, la función de pérdida está dada por la ecuación 6.13. Se mantiene el término L_{KL} ya que aun se utilizan datos de movimiento de la mirada en el encoder, lo que permite aproximar una distribución gaussiana z entregada como input en el decoder.

$$TotalLoss = L_{AOI} + w_{KL}L_{KL} \tag{6.13}$$

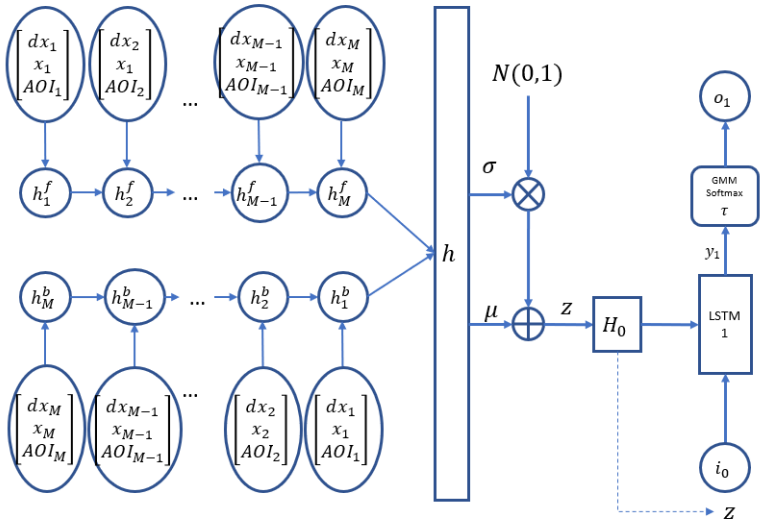


Figura 6.60: Arquitectura predicción AOI's temporal.

Se define el próximo AOI como aquel AOI que es alcanzado en la ventana de predicción en primer lugar, pero que no es el último alcanzado en la ventana de información. En caso de no existir se establece con el AOI simbólico 0. Por esta razón, para evitar la no existencia, se aumenta el tamaño de la ventana de predicción a 3 segundos.

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds, con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 3$ segundos y $\delta = 3$ segundos y los demás parámetros:

- tamaño de los batches : 100
- tasa de aprendizaje: 0.001
- decay rate: 0.999
- clip de gradiente: 1

- probabilidad dropout encoder : 0.9
- probabilidad dropout decoder : 0.9
- w_{KL} : 0.01
- epochs : 1000
- número de neuronas encoder: 256
- número de neuronas decoder: 256
- tamaño vector z: 128
- temperatura : 1

Se puede apreciar que en cada fold se tiene un comportamiento bastante similar. En este modelo se observa que el costo de softmax de AOI, al igual que en el caso anterior, no logra un entrenamiento deseable, donde los valores tienden a oscilar entre iteraciones, sin una disminución marcada. Por otro lado, manteniendo el mismo valor para w_{KL} se logra una disminución de los valores de L_{KL}, por lo cual, el z estimado presenta un comportamiento más similar a una distribución $\mathcal{N}(0, 1)$. Esto se puede ver gráficamente en 6.61, 6.62 y 6.63.

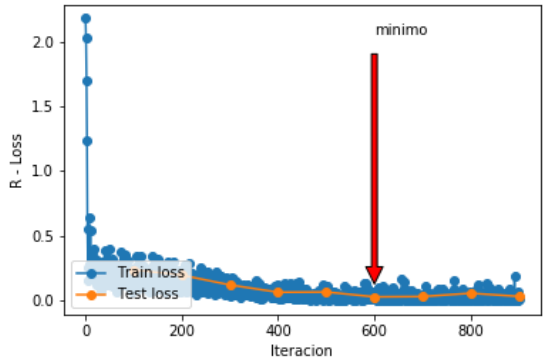


Figura 6.61: Costo softmax AOI por epoch, modelo 3

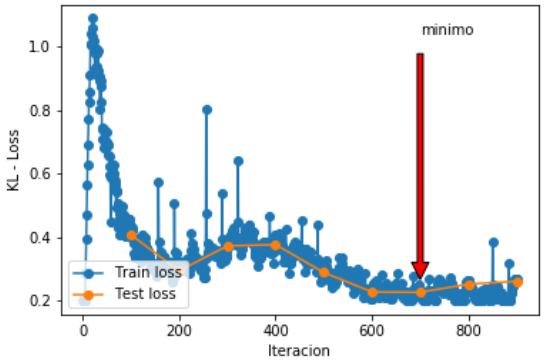


Figura 6.62: divergencia de Kullback Leibler por epoch, modelo 3.

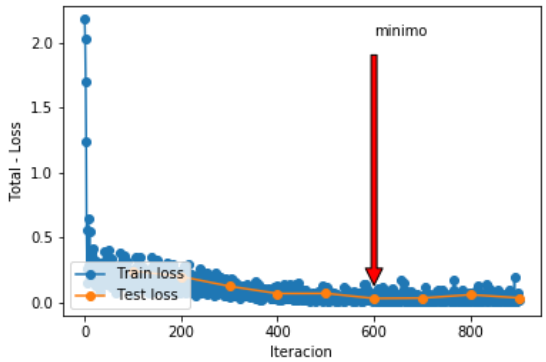


Figura 6.63: Costo total por epoch, modelo 3.

Las métricas promedio obtenidas al correr el modelo en los k-folds son las siguientes:

- L_{KL} entrenamiento: 0.27
- L_{AOI} entrenamiento: 0.01
- L_{Total} entrenamiento: 0.02
- L_{KL} test: 0.24
- L_{AOI} test: 0.02
- L_{Total} test: 0.03

De este modelo se puede apreciar lo siguiente. Dado que los resultados de L_{KL} tienden a disminuir a lo largo de las iteraciones, se piensa en alguna corrección a los modelos anteriores que permita este comportamiento independientemente de aumentar el valor de w_{KL}.

Dada la distribución de probabilidades calculada para una ventana de tiempo de predicción, se requiere de la elección de un AOI (muestreo). Para ello se pueden usar diferentes métodos. Para evaluar la calidad del modelo, en primer lugar se utiliza el método "argmax", es decir, considerar el próximo AOI como aquel que presente la mayor probabilidad de ocurrencia. Al realizar esto la métrica de accuracy para el conjunto de test resulta ser de 3,58 %. Por lo tanto, se concluye que el aprendizaje del modelo no es suficiente para la elección del siguiente AOI en la ventana de predicción.

Modelo Predicción próximo AOI con EEG

Posteriormente, para mejorar la información usada en el encoder, se agrega la información de EEG. La arquitectura se presenta en la imagen 6.64. Dado que la tasa de muestreo de los datos EEG es mayor, se debe agregar una segunda red de largo M_{e}, la cual entrega un estado oculto final h_{e} que es concatenado con z antes de aplicarle la función $tanh()$, generando H_0 para el decoder como la concatenación de los resultados entregados por las redes de EEG y los datos de mirada.

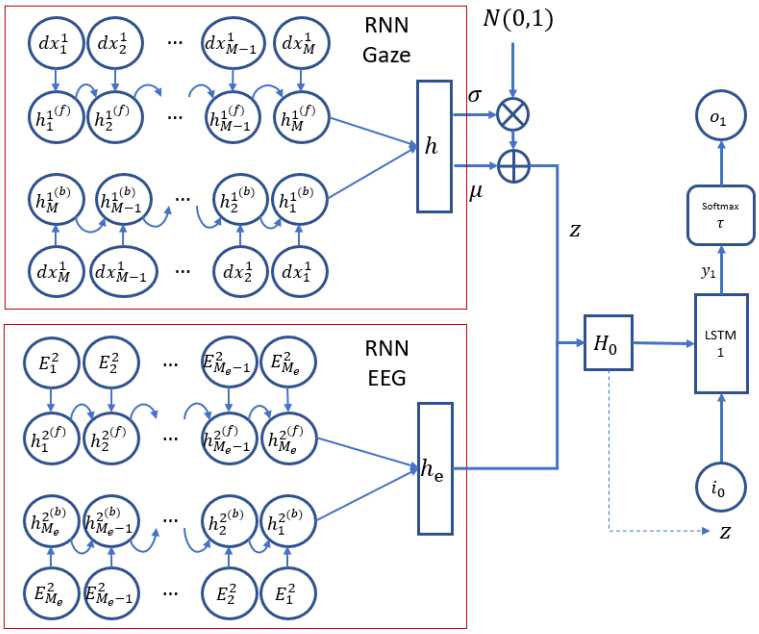


Figura 6.64: Arquitectura predicción AOI's temporal.

Los datos de EEG se entregan en forma de vectores E_{i} que constan de los valores tomados por cada electrodo, ajustados por la línea base del usuario.

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds, con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 3$ segundos y $\delta = 3$ segundos y los demás parámetros:

- tamaño de los batches : 100
- decay rate: 0.999
- tasa de aprendizaje: 0.001
- clip de gradiente: 1
- probabilidad dropout encoder Gaze : 0.9
- probabilidad dropout encoder EEG : 0.9
- probabilidad dropout decoder : 0.9
- w_{KL} : 0.01
- epochs : 1000
- número de neuronas encoder Gaze: 256
- número de neuronas encoder EEG: 256
- número de neuronas decoder: 512
- tamaño vector z: 128
- temperatura : 1

Se observa al correr el modelo en los conjuntos de entrenamiento y test, un comportamiento similar al modelo anterior. El costo de softmax de los AOI tiende a bajar a través de las iteraciones en el conjunto de entrenamiento, al igual que la divergencia KL. Sin embargo, los costos de softmax en el conjunto de testeo tienden aumentar, lo que implica un sobre-entrenamiento de la red. En las imágenes se destacan los valores menores de los costos en el conjunto de entrenamiento, lo cual ocurre en las primeras iteraciones.

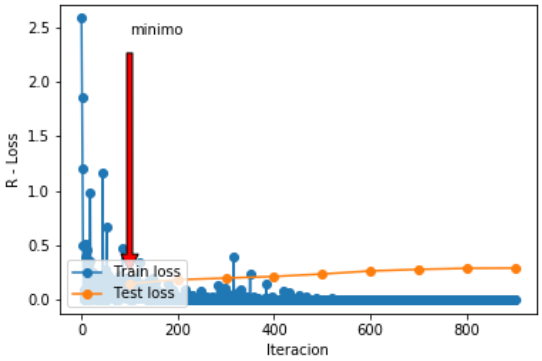


Figura 6.65: Costo softmax AOI por epoch, modelo 4

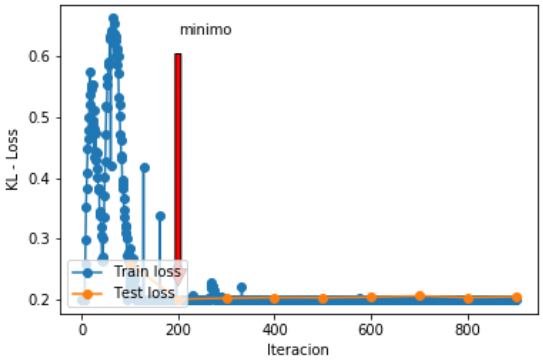


Figura 6.66: divergencia de Kullback Leibler por epoch, modelo 4.

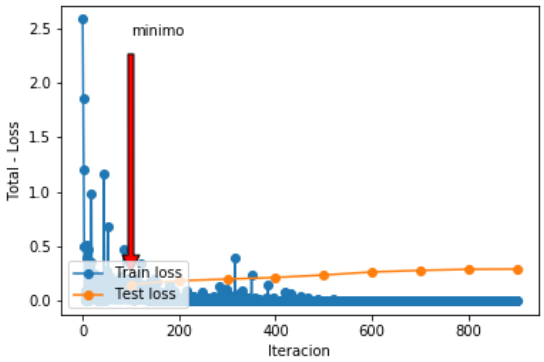


Figura 6.67: Costo total por epoch, modelo 4.

Al correr el modelo en los diferentes k-folds, se obtienen las siguientes métricas en prome-

dio:

- L_{KL} entrenamiento: 0.2
- L_{AOI} entrenamiento: 0.05
- L_{Total} entrenamiento: 0.05
- L_{KL} test: 0.23
- L_{AOI} test: 0.15
- L_{Total} test: 0.15

Para evitar el sobre-entrenamiento, se ejecuta el modelo nuevamente, pero disminuyendo el número de neuronas del encoder (ambas redes) y del decoder a 128. Además, se utilizan probabilidad de mantención de dropout de 0.8 en todas las redes.

En las ilustraciones 6.68, 6.69 y 6.70 se observa un ejemplo dentro de las k-fold iteraciones. Aunque el modelo presenta buen entrenamiento en los conjuntos de entrenamiento, en los conjuntos de testeo aún se presentan los problemas de poca genralización de los patrones aprendidos por la red.

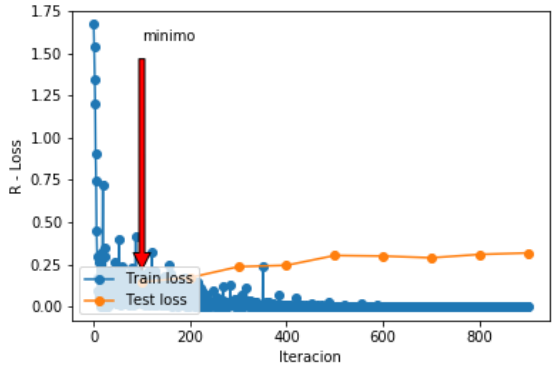


Figura 6.68: Costo softmax AOI por epoch, modelo 4, evitando sobre-entrenamiento

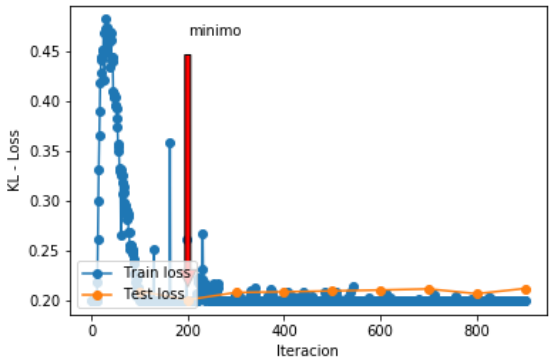


Figura 6.69: divergencia de Kullback Leibler por epoch, modelo 4, evitando sobre-entrenamiento

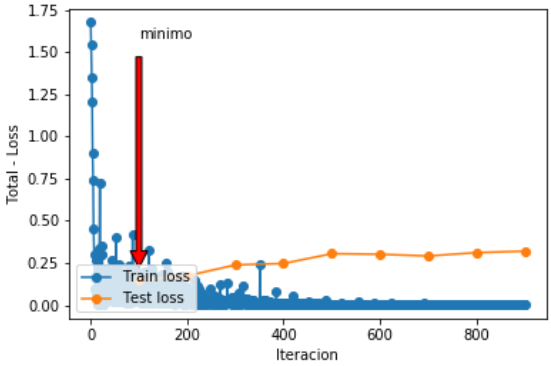


Figura 6.70: Costo total por epoch, modelo 4, evitando sobre-entrenamiento

Al correr el modelo en los diferentes k-folds, se obtienen las siguientes métricas en promedio:

- L_{KL} entrenamiento: 0.28
- L_{AOI} entrenamiento: 0.058
- L_{Total} entrenamiento: 0.061
- L_{KL} test: 0.28
- L_{AOI} test: 0.14
- L_{Total} test: 0.14

En conclusión, este modelo tiende a demorar demasiado en testear cada muestra del conjunto de testeo y, sin embargo, las ganancias en la disminución de costo en el conjunto de entrenamiento a lo largo de las iteraciones (se observa una mayor concentración cercana a cero, con menos oscilaciones) no son generalizables para el conjunto de testeo, aún cuando se utilicen métodos que permitan disminuir el sobre-entrenamiento de la red.

Modelo predicción próximo AOI con separación de datos de mirada

Respecto a la corrección del modelo para disminuir L_{KL} se propone la separación de los datos de mirada en diferentes redes, de manera que la red encargada de generar z sólo reciba información de las desviaciones de mirada, ajustadas por el factor para que su desviación estándar sea 1. En la imagen 6.71 se observa la nueva arquitectura, con los mismos datos de entrada que el modelo planteado en 6.4.8.

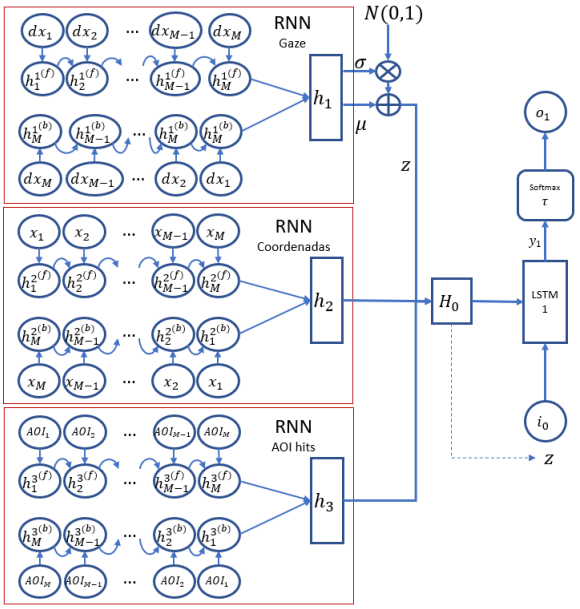


Figura 6.71: Arquitectura predicción próximo AOI con separación de datos encoder.

El estado inicial para el decoder se calcula a partir de la función $tanh()$ aplicada a la concatenación de z, h_2 y h_3. Se espera que el comportamiento del decoder respecto a L_{AOI} no cambie, ya que la información codificada es la misma. Por otro lado, z que le debería permitir mayor similitud a una distribución $\mathcal{N}(0,1)$, disminuyendo L_{KL}.

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds, con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 3$ segundos y $\delta = 3$ segundos y los demás parámetros:

- tamaño de los batches : 100
- decay rate: 0.999
- tasa de aprendizaje: 0.001
- clip de gradiente: 1
- probabilidad dropout encoder AOI : 0.9
- probabilidad dropout encoder gaze : 0.9
- probabilidad dropout encoder coordenadas : 0.9
- probabilidad dropout decoder : 0.9
- w_{KL} : 0.01
- epochs : 1000
- número de neuronas encoder AOI: 128
- número de neuronas encoder gaze: 256
- número de neuronas encoder coordenadas: 128
- número de neuronas decoder: 256
- tamaño vector z: 128
- temperatura : 1

Al entrenar los modelos en los k-folds, se observa un comportamiento bastante particular. Se observa que L_{KL} tiende a empezar con valores cercanos a la cota del modelo anterior (valor cercano a 0.2) pero presenta fuertes oscilaciones en algunas iteraciones. Este comportamiento se presenta de manera diferente en cada fold y puede ser debido al bajo valor de w_{KL}, el cual permite mayor holgura a la minimización de L_{KL}.

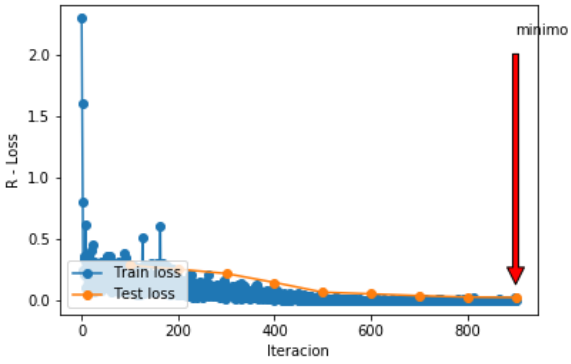


Figura 6.72: Costo softmax AOI por epoch, modelo 5

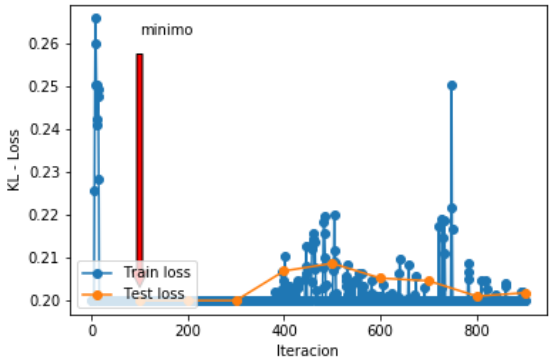


Figura 6.73: divergencia de Kullback Leibler por epoch, modelo 5.

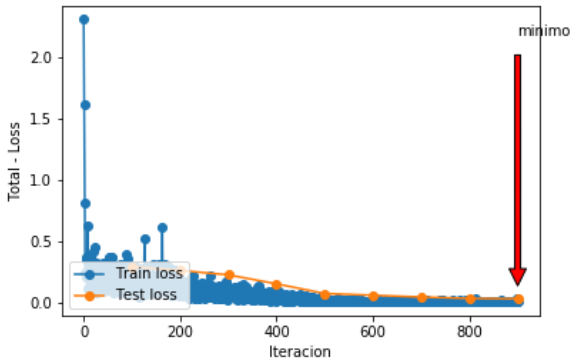


Figura 6.74: Costo total por epoch, modelo 5.

Al correr el modelo en los diferentes k-folds, se obtienen las siguientes métricas en prome-

dio:

- L_{KL} entrenamiento: 0.20
- L_{AOI} entrenamiento: 0.001
- L_{Total} entrenamiento: 0.007
- L_{KL} test: 0.20
- L_{AOI} test: 0.018
- L_{Total} test: 0.02

Se puede observar que este modelo presenta mejora es cuanto al tiempo de convergencia a los valores más bajos de L_{KL} que el modelo sin separación en el encoder. Al mismo tiempo L_{AOI} tiene una fuerte mejora en el conjunto de entrenamiento. Sin embargo, L_{AOI} del conjunto de test no logra capturar esas mejoras. Por lo tanto, se espera que el valor de accuracy no sea mucho mejor que los modelos anteriores.

Al igual que en el modelo presentado en 6.4.8, se generan muestras de próximos AOI's en las ventanas de tiempo de predicción, lo cual entrega un accuracy de 20 %, lo cual supera bastante al modelo anterior. Sin embargo, el método de selección del próximo AOI basado en escoger el área de mayor probabilidad no es suficiente.

Posteriormente, se ejecuta el mismo modelo, pero disminuyendo la probabilidad de mantención de dropout a 0.8 en todas las redes (encoder y decoder), de manera de evitar el sobre-entrenamiento de la red. Al ejecutar este modelo, no se logran mayores mejoras en el conjunto de entrenamiento, y se observan peores valores en los conjuntos de testeo.

- L_{KL} entrenamiento: 0.20
- L_{AOI} entrenamiento: 0.001
- L_{Total} entrenamiento: 0.007
- L_{KL} test: 0.20
- L_{AOI} test: 0.027
- L_{Total} test: 0.03

Conclusiones finales

Finalmente, dado los diferentes modelos anteriores, se puede concluir que esas arquitecturas no son lo suficientemente fuertes como para generar resultados generalizables a los conjuntos de testeo, por lo cual, las probabilidades puedes ser usadas pero no necesariamente escogiendo el próximo AOI en la ventana de predicción con el método de argmax, sino que se deben considerar diferentes alternativas de muestreo a partir de distribuciones categóricas o en su defecto, intentar mejorar los modelos con nuevas estructuras y datos de entrada.

6.5. Modelo predicción de estados mentales

6.5.1. Idea general y esquema de la solución

Posteriormente a los modelos de predicción de datos de la mirada y alcance de los AOI's, se generan modelos de predicción de diferentes caracterizaciones mentales del usuario, de manera que el sistema recomendador entregue información tanto de los movimientos oculares como de índices relacionados al estado mental de los usuarios.

La idea se basa en las ventanas de predicción descritas en la sección 6.4.4, en la cual se explica la generación de "ventanas de información" y "ventanas de predicción". Se busca generar un modelo similar a los planteados para la predicción de los movimientos en los datos de la mirada del usuario, pero que se puedan correr entregando resultados de manera independiente a los antes mencionados.

Para ello, se utilizan ciertas métricas como gold estándar, dependiendo del índice a estudiar, creando una etiqueta para cada ventana de tiempo de predicción del conjunto de entrenamiento a partir de métodos de clusterización.

Dado los clusters generados, se almacenan los centros de cada uno de ellos, de manera que para cada ventana de tiempo de predicción en los datos de testeo se pueda crear una etiqueta según el cluster más cercano creado con los datos de entrenamiento.

La métrica para evaluar la calidad de los clusters corresponde al puntaje de silhouette. Este es un método de validación e interpretación de consistencia dentro de los clusters. Es una métrica que entrega valores en el intervalo [-1,1] donde valores mayores indican que el objeto es fuertemente clasificado en su cluster y débilmente en los otros.

La métrica anterior se calcula punto a punto. Sea $a(\mathrm{i})$ la distancia promedio de el punto i a todos los demás puntos pertenecientes a su cluster. Sea $b(\mathrm{i})$ la distancia promedio más pequeña de i a los puntos pertenecientes a los otros clusters. Entonces se define el valor de silhouette para el punto i como:

$$s(\mathrm{i}) = \frac{b(\mathrm{i}) - a(\mathrm{i})}{max(a(\mathrm{i}), b(\mathrm{i}))}$$

El promedio de $s(\mathrm{i})$ sobre todos los puntos de un cluster es una medida de que tan estrechamente agrupados están los puntos dentro de un cluster. Y el promedio sobre todos los datos es una medida de que tan apropiadamente ha sido clusterizado los datos.

En las siguientes secciones se explican los conceptos a utilizar y los modelos planteados para cada uno de ellos.

El esquema de la utilización es el mismo que el usado para el modelo de predicción de fijaciones del segundo escenario, según la figura 6.39. El único cambio corresponde a la programación de los scripts principales llevados a cabo en TensorFlow.

6.5.2. Valencia y excitación

En primer lugar, se utiliza el denominado modelo plano de valencia y excitación, usado generalmente para el estudio de estados emocionales. Este plano consta de los ejes de valencia (valence) y excitación (arousal), tal como se muestra en la figura 6.75 presentada en el estudio [87].

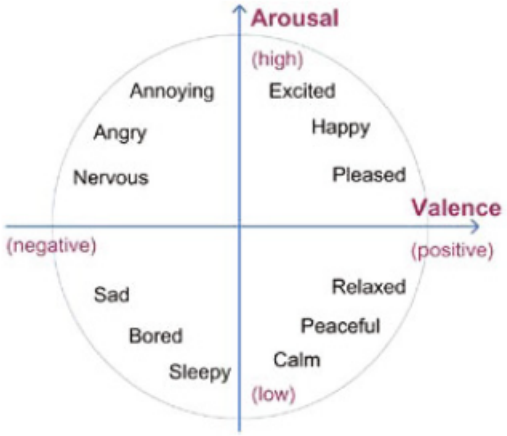


Figura 6.75: Estados emocionales y sus posiciones en el plano de valencia excitación.

Diferentes emociones son ubicadas dentro de este plano. La valencia define si la emoción es positiva o negativa y en que grado (cuadrantes derecho o izquierdo respectivamente) y la excitación define la intensidad de la emoción, yendo desde calmado (valor bajo) a emocionado (valor alto).

En el mismo estudio, se establecen las fórmulas para caracterizar los valores de valencia y excitación. Para ello, utilizan los electrodos AF3, AF4, F3 y F4 para su representación.

Excitación

Para el caso de la excitación, se extraen las bandas alfa [8-12] Hz y beta [12-30] Hz, mediante un filtro pasa-banda con una ventana de butter de orden 6.

La banda beta se asocia a estados de alerta o excitación mientras que la banda alfa se asocia a estados de relajación o inactivación. Por lo tanto, el ratio usado para indicar la excitación es $beta/alfa$. Este ratio es calculado para cada electrodo AF3, AF4, F3, F4, punto a punto.

En la siguientes ilustraciones 6.76, 6.77, 6.78, 6.79 se muestra el calculo del ratio para los cuatros electrodos en una ventana de predicción de ejemplo. Para hacer métricas comparables entre ventanas de predicción de diferentes usuarios, se normaliza la señal en el intervalo [-1,1].

Se puede observar que en general la señal se mantiene constante en torno a un valor con elevadas desviaciones puntuales. Se puede considerar que valores superiores a cero corresponden a estados de excitación y bajo cero, a relajación.

Posteriormente, cada ventana de tiempo de predicción es caracterizada por la media de sus ratios de Excitación en cada electrodo. Se espera que, dado el tiempo de la ventana, y la cantidad de muestras para cada electrodo, los peaks de la señal de ratios no afecten demasiado. Utilizando estas cuatro variables, se genera un indicador de una excitación alta o baja en la ventana de tiempo correspondiente.

Se lleva a cabo una clusterización de las ventanas de predicción en el conjunto de entrenamiento, utilizando dos clusters (relacionados a las dos categorías de excitación). Los resultados de cada cluster (centros) son almacenados para crear etiquetas de las ventanas de predicción en los conjuntos de testeo.

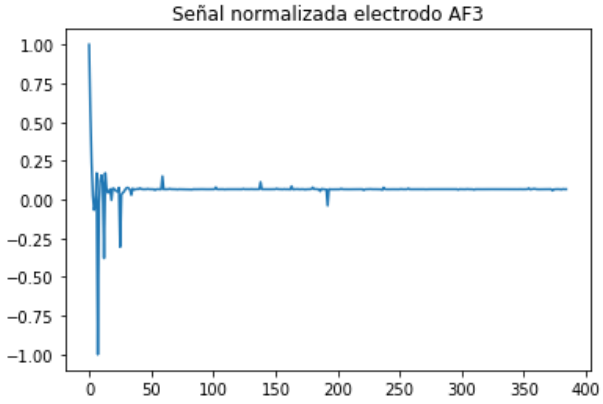


Figura 6.76: Señal normalizada electrodo AF3 para una ventana de predicción

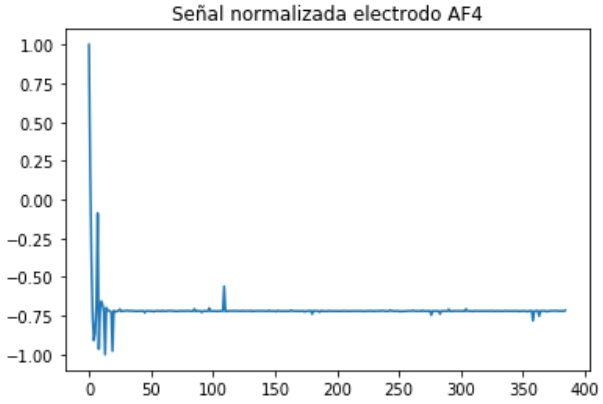


Figura 6.77: Señal normalizada electrodo AF4 para una ventana de predicción

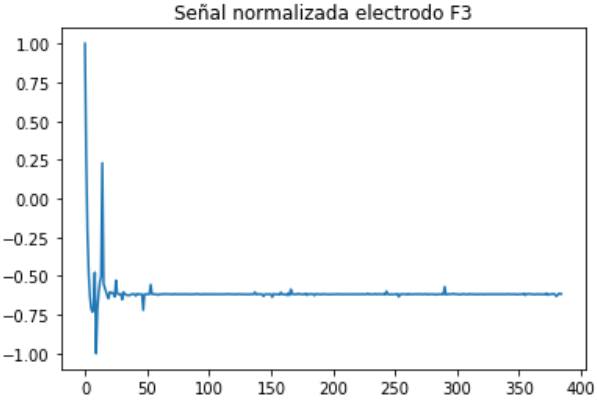


Figura 6.78: Señal normalizada electrodo F3 para una ventana de predicción

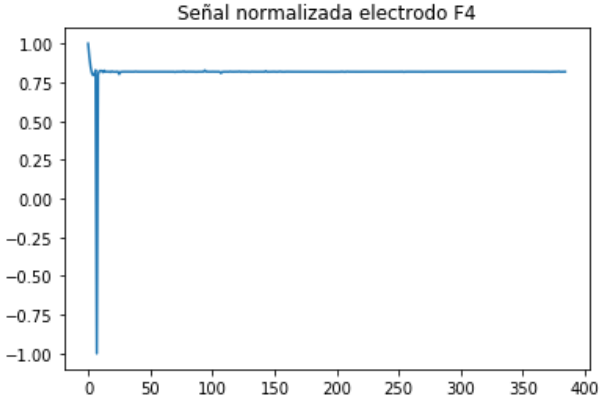


Figura 6.79: Señal normalizada electrodo F4 para una ventana de predicción

El modelo de cluster se realiza con K-means. En la ilustración 6.80 se muestra el cluster generado para las ventanas de predicción del conjunto de entrenamiento en el primer fold. El gráfico corresponde a la reducción de variables utilizando el método de PCA a dos dimensiones, mostrando una varianza explicada de 50 %.

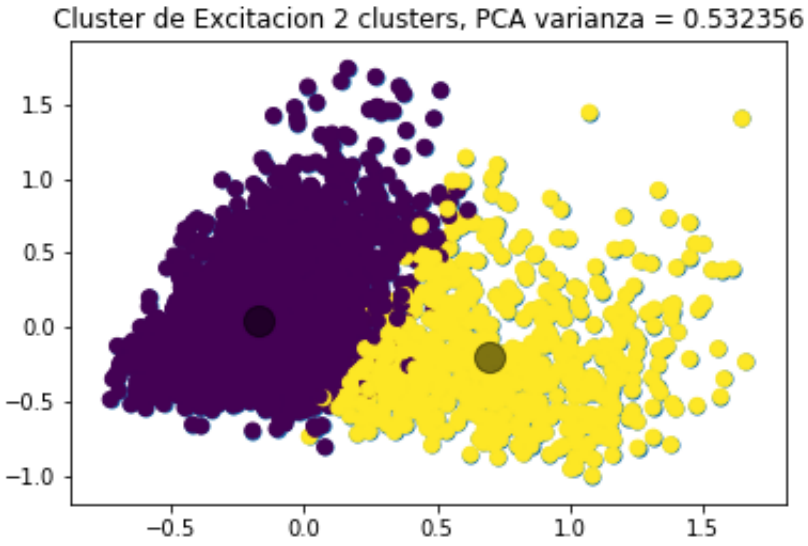


Figura 6.80: Clusters estados de valencia conjunto de entrenamiento, fold 1

En la tabla 6.18 se presenta la cantidad de ventanas de tiempo en cada categoría para conjunto de entrenamiento y de test. Se observa que en la mayoría de los casos las ventanas son categorizadas en la clase de alta excitación, siendo aproximadamente el 80 % de los casos en el conjunto de entrenamiento y 75 % en los conjuntos de test.

Tabla 6.18: Ventanas por clase, índice de excitación, conjuntos de entrenamiento y test, por cada fold.

		Fold											
Conjunto	**Etiqueta**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Promedio**	**Total**
Train	**Alta**	658	713	669	669	669	658	2764	669	658	658	879	8785
	(%)	(20 %)	(21 %)	(20 %)	(20 %)	(20 %)	(20 %)	(80 %)	(20 %)	(20 %)	(20 %)	(26 %)	(26 %)
	Baja	2660	2677	2615	2615	2615	2660	698	2615	2660	2660	2448	24475
	(%)	(80 %)	(79 %)	(80 %)	(80 %)	(80 %)	(80 %)	(20 %)	(80 %)	(80 %)	(80 %)	(74 %)	(74 %)
Test	**Alta**	87	63	92	92	92	87	147	92	87	87	93	926
	(%)	(26 %)	(24 %)	(25 %)	(25 %)	(25 %)	(26 %)	(75 %)	(25 %)	(26 %)	(26 %)	(28 %)	(28 %)
	Baja	253	205	282	282	282	253	49	282	253	253	239	2394
	(%)	(74 %)	(76 %)	(75 %)	(75 %)	(75 %)	(74 %)	(25 %)	(75 %)	(74 %)	(74 %)	(72 %)	(72 %)

El primer modelo corresponde a la predicción de la **excitación** del usuario en la ventana de predicción. Para ello se utiliza una red de encoder-decoder en la que la información de los electrodos AF3, AF4, F3 y F4 de la ventana de información, filtradas por pasa-banda y ajustadas por línea base, es codificada para que luego una red de decodificación entregue las probabilidades de ser una excitación alta o baja.

En la ilustración 6.81 se muestra un modelo de encoder-decoder que funciona como clasificador de secuencias. En el encoder, se codifican los datos de la ventana de información. Con esto, se genera un estado h con el cual se genera una ponderación $b_h + W_h h$ para la generación de un resultado softmax, $\hat{E}$, es decir, las probabilidades de que la ventana de predicción pertenezca a una de las dos clases de excitación. La función de pérdida para el

modelo queda representado por la ecuación 6.14.

$$L_{Excitacion} = -\frac{1}{N_{train}} \sum_{i=1}^{N_{train}} \sum_{j=1}^{2} E_j \log\left(\hat{E}_j\right) \tag{6.14}$$

Este primer enfoque permite crear un clasificador sin la necesidad de generar features representativas, sino que se busca que la red aprenda a leer las señales directamente, con lo cual se espera un menor tiempo de ejecución por parte del sistema recomendador.

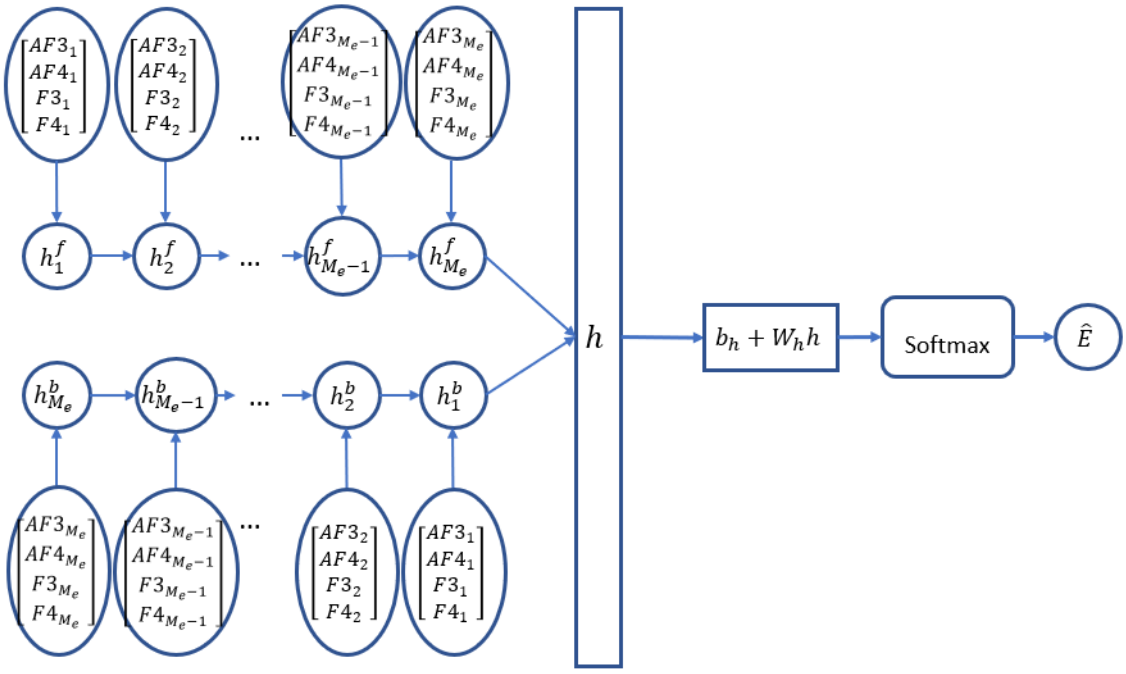


Figura 6.81: Arquitectura predicción estado de excitación en ventana de predicción.

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds, con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 3$ segundos y $\delta = 3$ segundos y los demás parámetros:

- tamaño de los batches : 100
- decay rate: 0.999
- tasa de aprendizaje: 0.001
- probabilidad dropout encoder : 0.7
- epochs : 1000
- número de neuronas encoder: 256

En la imagen 6.82 se observa el proceso de entrenamiento de la red en el primer fold, comportamiento compartido en las otras particiones. La función de pérdida disminuye en los conjuntos de entrenamiento. Sin embargo, este aprendizaje de la red no es generalizable a los conjuntos de test, lo que se ve en el aumento de la función de pérdida a lo largo de las iteraciones.

De las señales de excitación de los electrodos (figuras 6.76, 6.77, 6.78 y 6.79) se observa que para una ventana de tiempo, se tiende a tener valores positivos o negativos de manera

muy marcada. Sin embargo, el modelo no es capaz de aprender la tendencia de estas señales al cambiar de ventana temporal, es decir, desde la ventana de información a la ventana de predicción.

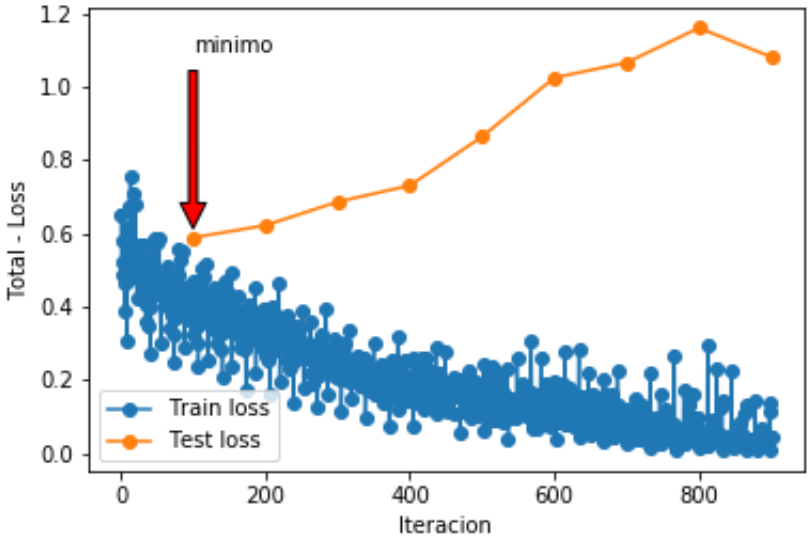


Figura 6.82: Función de pérdida a lo largo de las iteraciones, modelo Excitación.

Valencia

Para el caso de valencia, se mide comparando los niveles de activación entre dos hemisferios corticales. La inactivación del hemisferio frontal izquierdo se asocia a emociones negativas, y la inactivación del hemisferio frontal derecho se asocia a una idea de respuesta, con emoción positiva. Utilizando los electrodos F3 y F4, y sus bandas alpha (α) y beta(β), se puede caracterizar la valencia a partir de la siguiente fórmula valencia $= \alpha_{F4}/\beta_{F4} - \alpha_{F3}/\beta_{F3}$.

Para cada ventana de tiempo de predicción se genera una "señal de valencia", dada por la fórmula anterior. Para esto, la señal de cada electrodo en la ventana de predicción, F3 y F4, es ajustada por la línea base del usuario y luego se aplica un filtro pasa-banda con ventana de butter de orden 6, con rango (8-12) Hz para la señal alpha y (12-30) para la señal beta.

Dado lo anterior, cada ventana puede ser representada a partir de su señal de valencia. Para ello, se utiliza la valencia media dentro de la ventana para crear una de las categorías: positivo o negativo. Dada la naturaleza de la fórmula de valencia, valores mayores a cero corresponden a valencia positiva y valores menores a cero a valencia negativa.

A pesar de que teóricamente se puede clasificar cada ventana según la media de la valencia, se estudia la naturaleza de las ventanas de predicción en el conjunto de entrenamiento a partir de generación de clusters, donde cada ventana se representa por la valencia media y su desviación estándar.

En la ilustración 6.83 se observa la señal de valencia calculada para una ventana de predicción como ejemplo. Se puede observar que en general, dentro de una ventana, la valencia

se mantiene en valores neutros, con grandes desviación en ciertos instantes. Para hacer comparable los valores medios y desviación estándar, la señal es normalizada al intervalo [-1,1], tal como se muestra en la imagen 6.84.

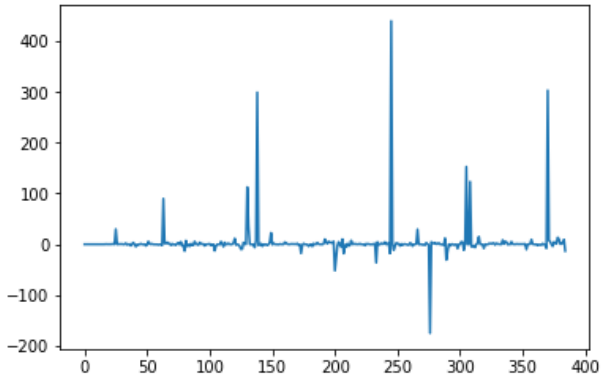


Figura 6.83: Señal de valencia cruda, ejemplo de una ventana

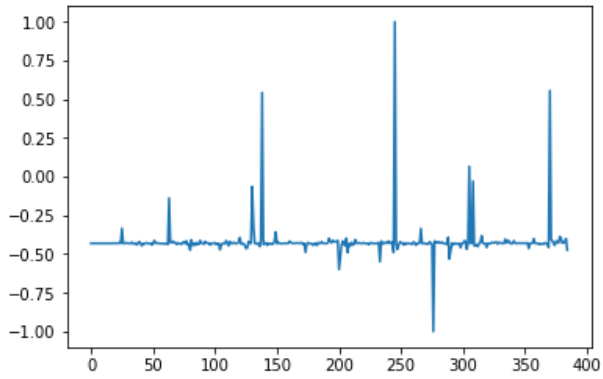


Figura 6.84: Señal de valencia intervalo [-1,1], ejemplo de una ventana

En la ilustración 6.85 se muestra el cluster generado para las ventanas de predicción del conjunto de entrenamiento en el primer fold. Se puede observar que los clusters son generados de acuerdo a lo anterior, es decir, se crean dos grupos, los de media mayor a cero y los de valor negativo, los que se pueden atribuir a las categorías valencia positiva y negativa, respectivamente.

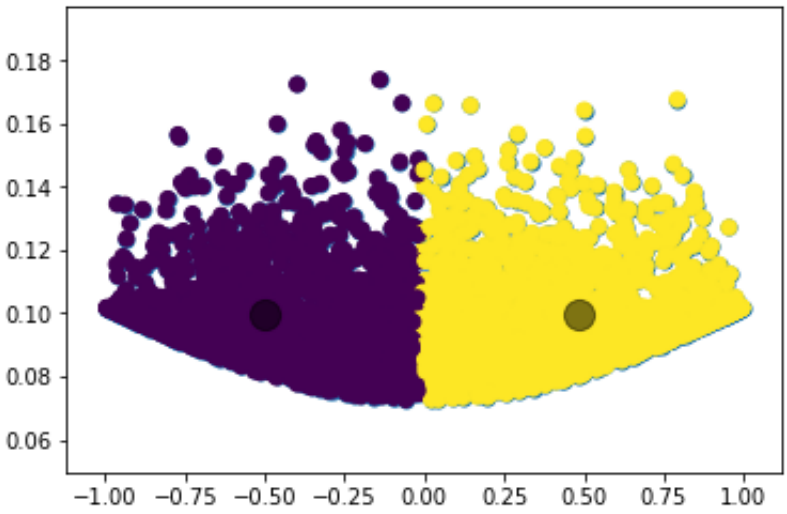


Figura 6.85: Clusters estados de valencia conjunto de entrenamiento, fold 1

Teniendo el modelo de clusterización calculado con el conjunto de entrenamiento, se procede a etiquetar las ventanas de predicción del conjunto de testeo, según el cluster más cercano a cada una de ellas.

Los modelos de predicción de valencia realizan la clusterización de los valores en las venta-

nas de entrenamiento para cada fold, guardan los resultados y entregan las etiquetas para los conjuntos de testeo. De esta forma, se genera la tabla 6.19, en la cual se muestra la cantidad de ventanas para cada clase en los conjuntos de entrenamiento y testeo, para cada fold.

Tabla 6.19: Ventanas por clase, índice de valencia, conjuntos de entrenamiento y test, por cada fold.

		Fold											
Conjunto	**Etiqueta**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Promedio**	**Total**
Train	**Positiva (%)**	1692 (51 %)	1728 (51 %)	1614 (49 %)	1614 (49 %)	1614 (49 %)	1692 (51 %)	1774 (51 %)	1614 (49 %)	1692 (51 %)	1692 (51 %)	1673 (50 %)	16726 (50 %)
	Negativa (%)	1626 (49 %)	1662 (49 %)	1670 (51 %)	1670 (51 %)	1670 (51 %)	1626 (49 %)	1688 (49 %)	1670 (51 %)	1626 (49 %)	1626 (49 %)	1653 (50 %)	16534 (50 %)
Test	**Positiva (%)**	183 (54 %)	147 (55 %)	169 (45 %)	169 (45 %)	169 (45 %)	183 (54 %)	101 (52 %)	169 (45 %)	183 (54 %)	183 (54 %)	166 (50 %)	1656 (50 %)
	Negativa (%)	157 (46 %)	121 (45 %)	205 (55 %)	205 (55 %)	205 (55 %)	157 (46 %)	95 (48 %)	205 (55 %)	157 (46 %)	157 (46 %)	166 (50 %)	1664 (50 %)

Se observa que en general, para cada uno de los fold, se mantiene el balance de clases entre las ventanas de alta y baja valencia, estando ambas en 50 % aproximadamente. Por lo tanto, el desempeño de un clasificador binario será comparado a una probabilidad base de 50 %.

En primer lugar se busca generar un clasificador que reciba como input la totalidad de datos de la ventana de información en cada muestra. De esta manera se prueba un modelo encoder-decoder, mostrado en la imagen 6.86.

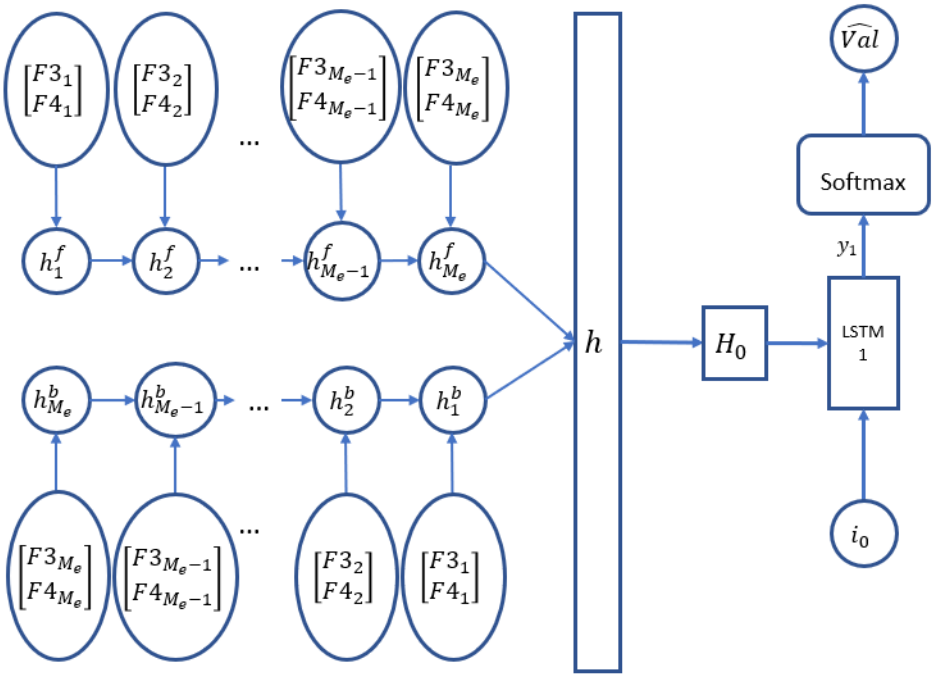


Figura 6.86: Arquitectura predicción estado de valencia en ventana de predicción.

La estructura de la imagen 6.86 consta de un codificador bidireccional, donde en cada paso temporal se entregan los valores adquiridos en los electrodos F3 y F4. Estos generan un

estado oculto inicial para el decoder, el cual consta de una celda LSTM, la cual entrega los valores $\hat{Val}$, correspondientes a las distribuciones de probabilidad a una valencia positiva o negativa.

Esta red es entrenada end-to-end, minimizando los costos de la función de pérdida 6.15, que corresponde a la log-verosimilitud de los resultados de clase para cada ventana de predicción.

$$L_{Valencia} = -\frac{1}{N_{train}} \sum_{\mathrm{i}=1}^{N_{train}} \sum_{j=1}^{2} Val_j \log\left(\hat{Val}_j\right) \tag{6.15}$$

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds, con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 3$ segundos y $\delta = 3$ segundos y los demás parámetros:

- tamaño de los batches : 100
- decay rate: 0.999
- tasa de aprendizaje: 0.001
- clip de gradiente: 1
- probabilidad dropout encoder AOI : 0.9
- probabilidad dropout decoder : 0.9
- epochs : 1000
- número de neuronas encoder: 256
- número de neuronas decoder: 256

En la imagen 6.87 se presenta el proceso de entrenamiento de la red para el primer fold. Se observa que a lo largo de las iteraciones, no se logra un aprendizaje constante del modelo, lo que se evidencia en que la función de pérdida en los conjuntos de entrenamiento presentan oscilaciones constantes, sin lograr una disminución considerable, luego de las 100 iteraciones.

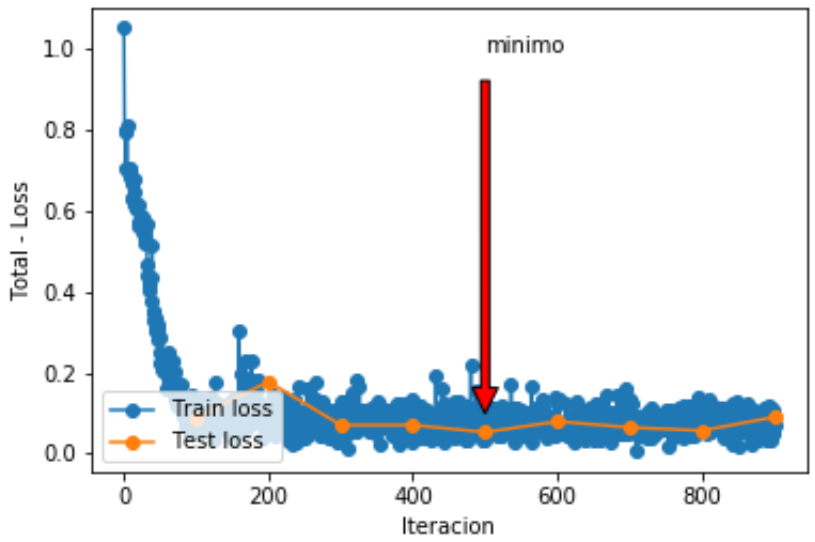


Figura 6.87: Función de pérdida a lo largo de las iteraciones, primer modelo valencia.

Los valores promedio de $L_{valencia}$ en los conjuntos de entrenamiento y test, son respectivamente 0.093 (0.0271) y 0.0584 (0.006), donde entre paréntesis se muestra la desviación

estándar. Al aplicar un muestreo de argmax en cada una de las ventanas, se compara la clase de valencia entregada por el modelo de clusters y la predicción hecha por el modelo, lo que entrega un accuracy promedio de 50 %, es decir, no logra mejora respecto a la clasificación básica (escoger siempre la clase mayoritaria).

En conclusión, el modelo no es capaz de aprender lo suficiente en la fase de codificación. Sin embargo, lo aprendido es generalizado a los conjuntos de test. Esto se puede deber a dos razones.

En primer lugar, la información cruda de los electrodos entregada al codificador tiene demasiado ruido, en comparación a sólo usar las bandas alpha y beta de dichos electrodos, por lo cual, el modelo no es capaz de decodificar correctamente.

Por otro lado, el mal desempeño puede deberse a que la señal de valencia no es acumulativa, es decir, los valores en la ventana de información no son suficientes para asegurar el comportamiento futuro en las ventanas de predicción. Esto se evidencia en la ilustración 6.84, donde la señal de valencia tiende a presentar valores característicos en momentos puntuales. Es decir, es una señal que presenta poca información de manera acumulativa y tiende a ser más notable en momentos específicos.

6.5.3. Carga Cognitiva

Otro indicador de estado mental del usuario corresponde a la carga cognitiva. Esta se basa en la teoría de que la memoria de trabajo es limitada para tareas nuevas, pero ilimitada cuando se trabaja con información de la memoria de largo plazo.

El proceso de carga cognitiva se define como la carga de ejecución que cierta tarea impone durante su ejecución. La capacidad de procesar la carga cognitiva por un usuario es limitada y cuando es sobrepasada, el rendimiento del usuario decae. Por lo tanto, si el sistema recomendador es capaz de entregar la información acerca de este indicador, se puede tomar decisiones en cuanto a la información que se le entrega al usuario en cada momento a través del sitio web.

Una de las maneras más utilizadas para la medición de la carga cognitiva de un usuario se basa en la utilización de los datos de la dilatación pupilar. Este método es factible ya que los datos fueron adquiridos en un ambiente de iluminación controlada, por lo cual los cambios lumínicos no afectan el tamaño pupilar de los usuarios.

Utilizando los métodos de filtrado de señal expuestos en la sección 6.3.5 se genera una señal con los datos de pupila de mayor confiabilidad. En las ilustraciones 6.88 y 6.89 se presentan las señales de tamaño pupilar de los ojos izquierdo y derecho respectivamente para un usuario de ejemplo. Al aplicar los filtros a la señal y ajustar los valores por la línea base del usuario, se obtienen las señales graficadas en las ilustraciones 6.90 y 6.91.

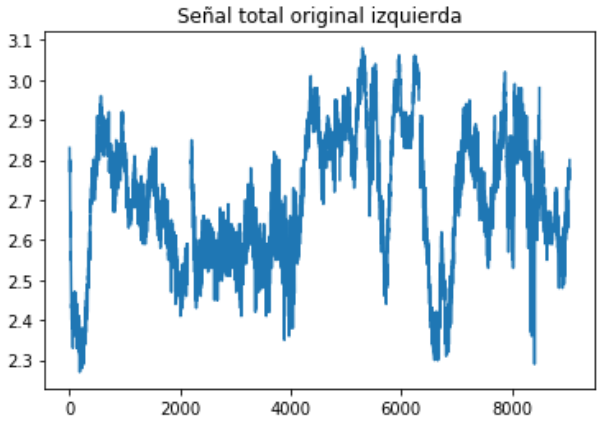


Figura 6.88: Señal tamaño pupilar ojo izquierdo para un usuario

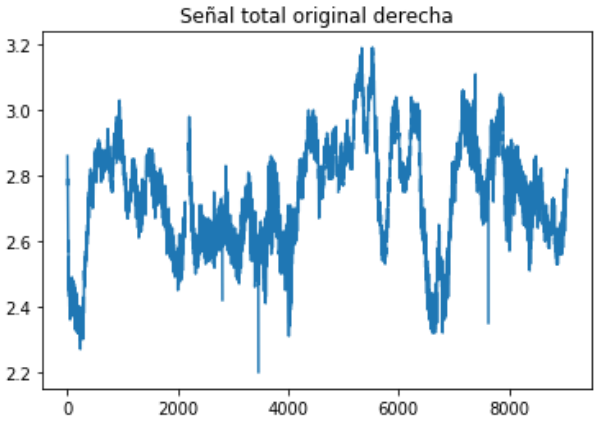


Figura 6.89: Señal tamaño pupilar ojo derecho para un usuario

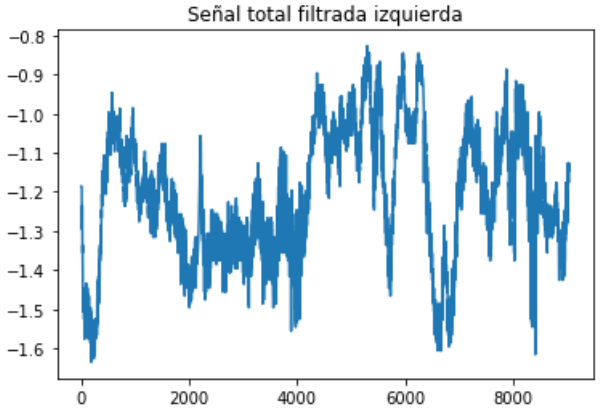


Figura 6.90: Señal tamaño pupilar ojo izquierdo, filtrada y ajustada por línea base, para un usuario

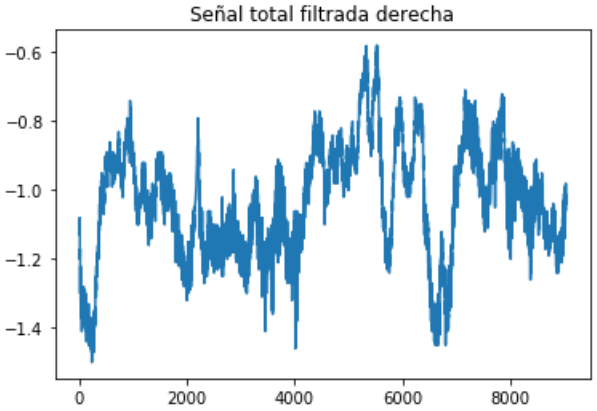


Figura 6.91: Señal tamaño pupilar ojo derecho, filtrada y ajustada por línea base, para un usuario

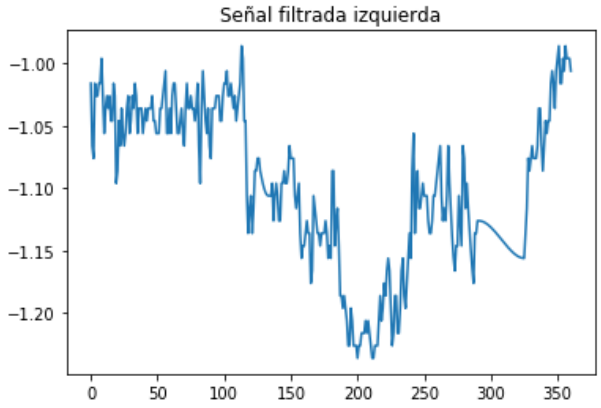


Figura 6.92: Señal tamaño pupilar ojo izquierdo, filtrada y ajustada por línea base, para una ventana de tiempo

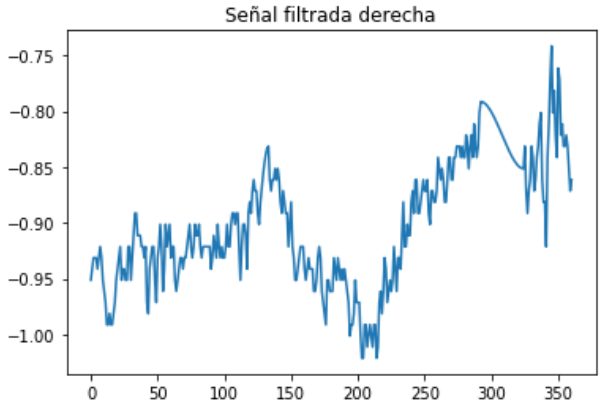


Figura 6.93: Señal tamaño pupilar ojo derecho, filtrada y ajustada por línea base, para una ventana de tiempo

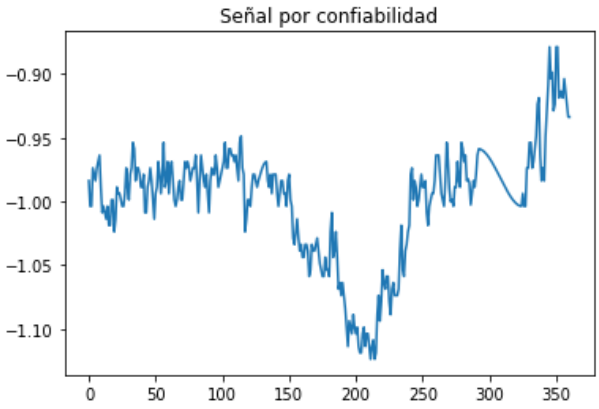


Figura 6.94: Señal tamaño pupilar promedio por confiabilidad, filtrada y ajustada por línea base, para una ventana de tiempo

En las ilustraciones 6.92 y 6.93 se presentan las señales de tamaño pupilar para una ventana de tiempo de predicción tomada como ejemplo. Se observa el efecto de la interpolación en los pestañeos detectados. Con estas dos señales se genera la señal mostrada en 6.94, correspondiente a la señal de tamaño pupilar ponderada por la confiabilidad de cada muestra.

Cada ventana de tiempo de predicción se caracterizan por la media y desviación estándar de la señal pupilar filtrada. Con estas variables, se crean clusters de las ventanas de predicción del conjunto de entrenamiento. Se prueba con la validación cruzada en los 10 folds con el número de cluster en el rango [2,5]. En las ilustraciones 6.95, 6.96, 6.97, 6.98 se muestran los resultados de la clusterización del primer fold. Dado el índice de silhouette se establece la utilización de dos clusters, ya que es más cercano a uno. Para dos clusters, el índice es de 0.42, para 3 es de 0.36, para 4 es de 0.31 y para 5 es de 0.32. Todos los clusters han sido generados luego de normalizar las señales de cada usuario al intervalo [-1,1] para hacer los resultados comparables entre ellos.

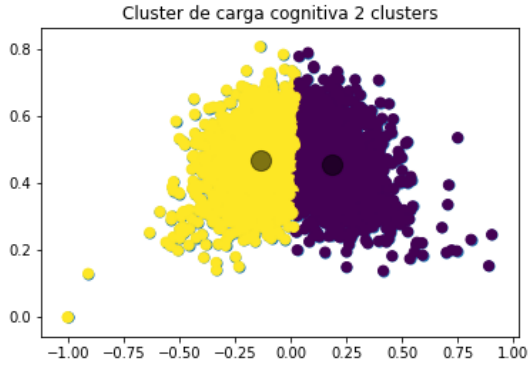


Figura 6.95: Utilización de 2 clusters para la categorización de carga cognitiva

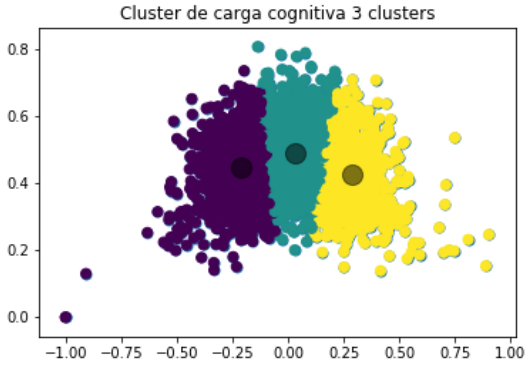


Figura 6.96: Utilización de 3 clusters para la categorización de carga cognitiva

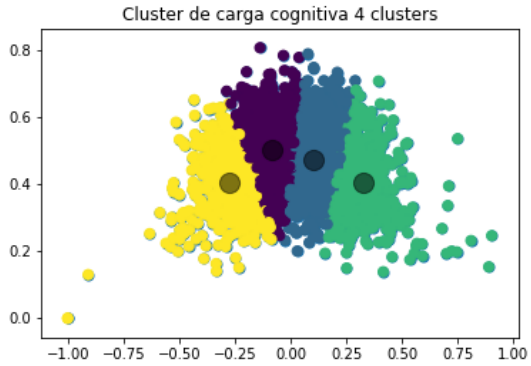


Figura 6.97: Utilización de 4 clusters para la categorización de carga cognitiva

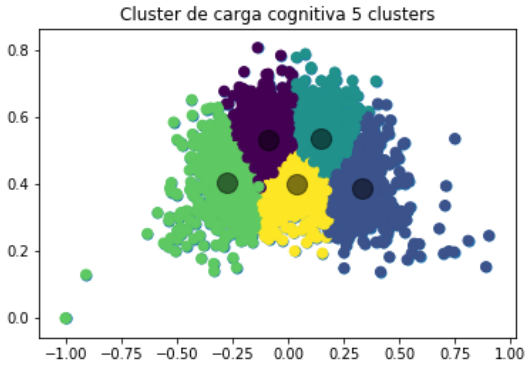


Figura 6.98: Utilización de 5 clusters para la categorización de carga cognitiva

Dado los clusters generados en los conjuntos de entrenamiento, se guardan los centros de los clusters para generar las etiquetas de los conjuntos de testeo. Cada ventana de predicción del conjunto de test es clasificado en una de las dos clases de carga cognitiva: carga alta y carga baja.

En la tabla 6.20 se resume el total de muestras categorizadas en cada clase para los conjuntos de entrenamiento y test, en cada uno de los folds utilizados en la evaluación cruzada. Se observa que el porcentaje de cada clase se mantiene balanceado a través de los folds (aproximadamente 50 % para cada clase).

Tabla 6.20: Ventanas por clase, índice de carga cognitiva, conjuntos de entrenamiento y test, por cada fold.

		Fold											
Conjunto	**Etiqueta**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Promedio**	**Total**
Train	**Alta**	1731	1609	1709	1709	1709	1731	1819	1709	1731	1731	1719	17188
	(%)	(52 %)	(47 %)	(52 %)	(52 %)	(52 %)	(52 %)	(53 %)	(52 %)	(52 %)	(52 %)	(52 %)	(52 %)
	Baja	1588	1782	1576	1576	1576	1588	1644	1576	1588	1588	1608	16082
	(%)	(48 %)	(53 %)	(48 %)	(48 %)	(48 %)	(48 %)	(47 %)	(48 %)	(48 %)	(48 %)	(48 %)	(48 %)
Test	**Alta**	167	134	189	189	189	167	99	189	167	167	166	1657
	(%)	(49 %)	(50 %)	(51 %)	(51 %)	(51 %)	(49 %)	(51 %)	(51 %)	(49 %)	(49 %)	(50 %)	(50 %)
	Baja	173	134	185	185	185	173	97	185	173	173	166	1663
	(%)	(51 %)	(50 %)	(49 %)	(49 %)	(49 %)	(51 %)	(49 %)	(49 %)	(51 %)	(51 %)	(50 %)	(50 %)

Dada la definición de las clases, se lleva a cabo una clasificación dada la información en las ventanas de información de cada muestra. Para ello se prueba un primer modelo, mostrado en la ilustración 6.99.

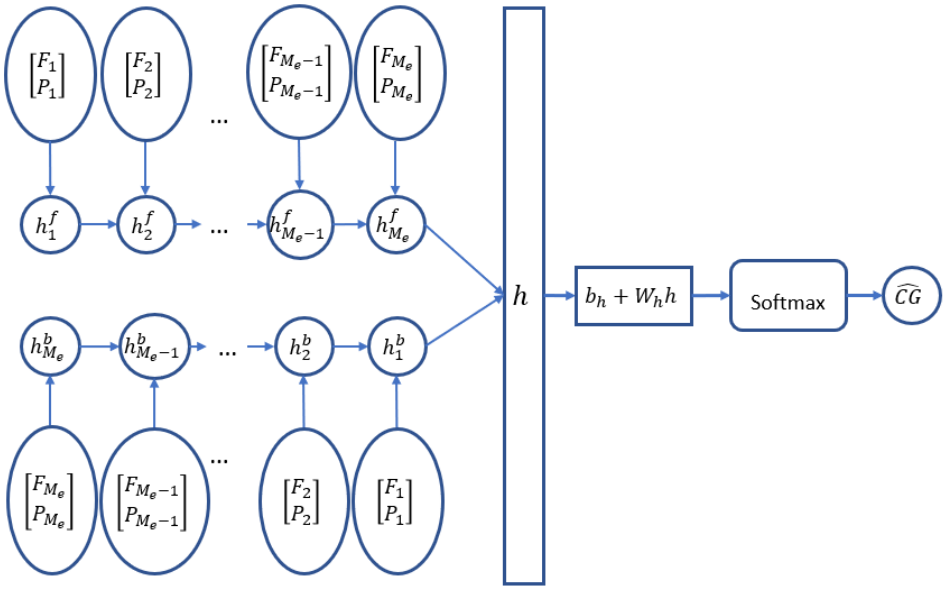


Figura 6.99: Arquitectura predicción nivel de carga cognitiva en ventana de predicción.

En la ilustración 6.99 se muestra un modelo de encoder decoder que funciona como clasificador de secuencias. En el encoder, se codifican los datos de la ventana de información. Con esto, se genera un estado h con el cual se genera una ponderación $b_h + W_h h$ con la cual es posible generar un resultado softmax, $\hat{CG}$, es decir las probabilidades de que la ventana de predicción pertenezca a una de las dos clases de carga cognitiva. La función de pérdida para el modelo queda representado por la ecuación 6.16.

$$L_{CargaCognitiva} = -\frac{1}{N_{train}} \sum_{\mathrm{i}=1}^{N_{train}} \sum_{j=1}^{2} CG_j \log\left(\hat{CG}_j\right) \tag{6.16}$$

En la codificación de ventanas de información, se busca usar las señales EEG, filtrada y ajustada por línea base, ya que presentan mayor información en ventanas de tiempo pequeñas.

El primer modelo, utiliza la información de los electrodos correspondientes a los lóbulos frontales y parietales: F3, F4, F7, F8, P7 y P8, específicamente, las bandas alfa y theta, las cuales han sido usadas en diversos estudios [58][6][90].

Dado que se busca un modelo con poco pre-procesamiento de los datos de entrada (rapidez de ejecución) cada input de la red del encoder se entregan los vectores $P_{\mathrm{i}} = [P7_{\mathrm{i}}, P8_{\mathrm{i}}]$ y $F_{\mathrm{i}} = [F3_{\mathrm{i}}, F4_{\mathrm{i}}, F7_{\mathrm{i}}, F8_{\mathrm{i}}]$, donde cada i representa una muestra de datos dentro de las M_{e} presentes en la ventana de tiempo de predicción.

Este modelo es aplicado corriendo los conjuntos de entrenamiento y test de los k-folds, con los parámetros de ventana de tiempo de $\tau = 5$ segundos, $\Delta = 3$ segundos y $\delta = 3$ segundos y los demás parámetros:

- tamaño de los batches : 100
- decay rate: 0.999
- tasa de aprendizaje: 0.001
- clip de gradiente: 1
- probabilidad dropout encoder : 0.7
- epochs : 1000
- número de neuronas encoder: 128

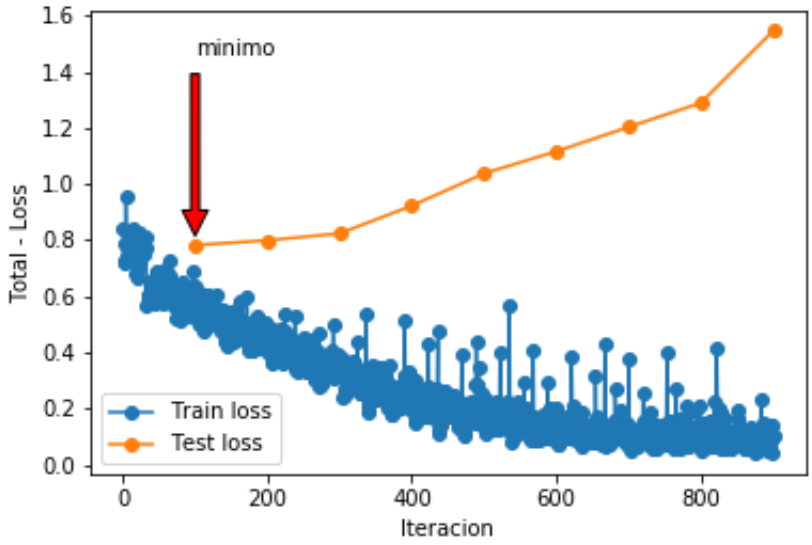


Figura 6.100: Función de pérdida a lo largo de las iteraciones, primer modelo carga cognitiva.

En la imagen 6.100 se observa el proceso de entrenamiento de la red para el primer fold de la validación cruzada, lo cual sigue un comportamiento similar en los otros folds. Se aprecia que la función de pérdida en el conjunto de entrenamiento disminuye a lo largo de las iteraciones. Sin embargo el aprendizaje adquirido por la red no es generalizable para los datos en el conjunto de test. Esto, a pesar de considerar una probabilidad de mantención de dropout de 0.7 en las celdas del encoder.

Una de las causas que puede provocar este sobre entrenamiento del modelo es la gran cantidad de parámetros que existen en la red de codificación, dada por la cantidad de inputs

entregados. Para evitar esto, se considera un siguiente modelo en el que para cada paso temporal del encoder se entrega la señal pupilar filtrada registrada en la ventana de información, en vez de los vectores $[F_i, P_i]$. La estructura de la red se mantiene al igual que en la ilustración 6.99.

Se corre el modelo utilizando los mismos parámetros que en le caso anterior. En la imagen 6.101 se observa el proceso de entrenamiento del modelo en el primer fold, comportamiento similar en todos los demás. En este caso, los datos de dilatación pupilar no logran generar un proceso de entrenamiento de la red, donde las funciones de pérdida de los conjuntos de entrenamiento y test se mantienen en torno al valor 0.7 sin presentar mejoras en las primeras 1000 iteraciones. Tampoco se aprecia una tendencia a la baja de estos valores.

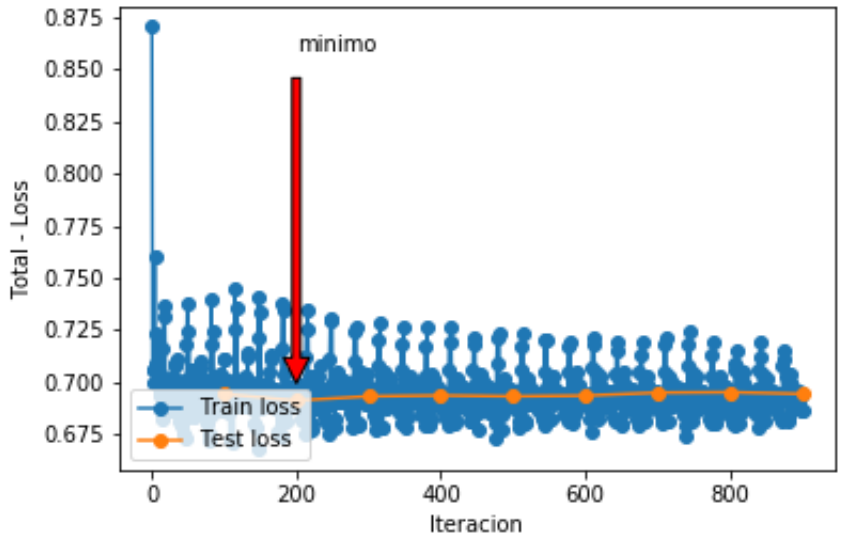


Figura 6.101: Función de pérdida a lo largo de las iteraciones, segundo modelo carga cognitiva.

Finalmente se prueba una red de clasificación de secuencias, donde en el encoder se entregan las secuencias de los datos de dilatación pupilar y de los datos de EEG de la ventana de información en redes separadas. Estas son codificadas por redes bidireccionales LSTM, las cuales se agregan formando el estado oculto h. Con este estado, se utiliza la ponderación por W_h, se le suma los sesgos b_h y finalmente se calcula la función softmax para las probabilidades de cada clase de carga cognitiva.

En la figura 6.102 se muestra el proceso de entrenamiento y los cambios en las funciones de pérdida de los conjuntos de entrenamiento y de test para el primer fold. Se observa que en este caso el modelo tiende aprender en el conjunto de entrenamiento, sin embargo nuevamente no es generalizable a los datos de los conjuntos de test.

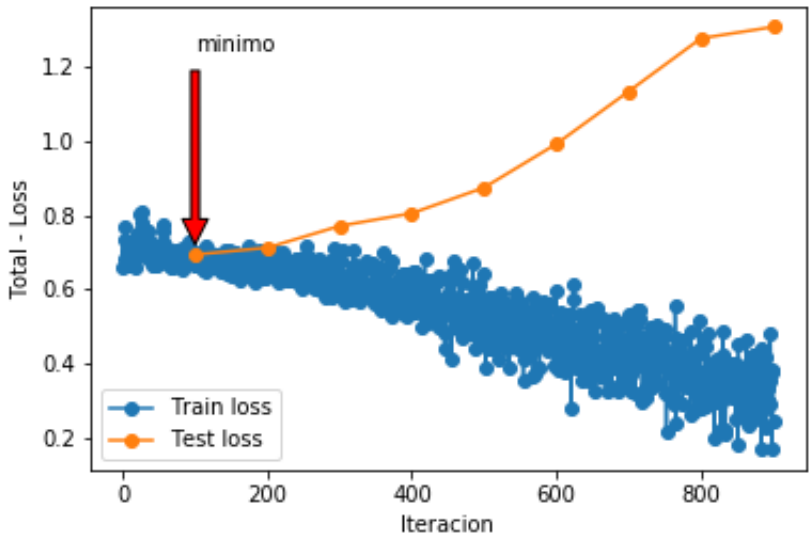


Figura 6.102: Función de pérdida a lo largo de las iteraciones, tercer modelo carga cognitiva.

En conclusión, para la predicción de carga cognitiva, se observa que la clasificación en dos estados mentales es posible. Sin embargo, el uso de los datos EGG del pasado no permiten una estimación de la clase de carga cognitiva en las ventanas de tiempo siguientes. Esto puede deberse a que la señal de tamaño pupilar tiende a ser muy variable en ventanas de tiempo de tamaño mayor a un segundo (tiempo entre la ventana de información y la ventana de predicción). Por otro lado, puede que la utilización de features extraídas de las señales EEG o pupilar permitan mejorar los resultados.

Capítulo 7

Conclusiones

El trabajo realizado en la presente tesis incluye la formulación e implementación completa de tres modelos de predicción de indicadores de usuarios que navegan un sitio web, basado en los registros de datos de mirada y otras señales fisio-psicológicas captadas durante el proceso. Así, se utilizan enfoques nuevos en cuanto a los trabajos revisados en el estado del arte, revelando datos útiles para futuras investigaciones.

La principal complejidad de este trabajo recae en la naturaleza dinámica del estímulo, constituyente a una página web que reacciona ante el accionar del usuario, desplegando barras de menú o desplazándose con las barras laterales, además de buscar resultados en tiempo real, es decir, predicciones futuras en función del comportamiento pasado, lo cual no ha sido desarrollado lo suficiente en la literatura.

El primer modelo, es decir, la predicción de fijaciones a nivel semántico de AOI's, se sustenta en un proceso de tres etapas.

En la primera etapa del modelo se propone un concepto de "intención de visita", el cual modela el ingreso o no del usuario a cada área de interés definido en la página web, durante una ventana de tiempo. Esta intención se modela como un vector con valores cero y uno y es posible generar una predicción utilizando métodos de clasificación multi-etiqueta. Esto se realiza con dos definiciones semánticas de componentes de la página, una con 6 AOI's y otra con 4 AOI's.

Se observa que la mejor configuración, para el caso de 6 AOI's, corresponde al método de 20 vecinos más cercanos con binary relevance, y una selección de 10 variables usando MLMIM, lo que permite una métrica de exactitud de 0.3 (vectores categorizados correctamente en todas sus componentes), e individualmente cada AOI es clasificado con un accuracy de 0.78 % en promedio y la métrica de accuracy es de 0.571 (correcta predicción de las etiquetas activas).

Se observa que la mejor configuración, para el caso de 4 AOI's, corresponde al método de 30 vecinos más cercanos con classification chain, y una selección de 10 variables usando MLMIM, lo que permite una métrica de exactitud de 0.313 (vectores categorizados correctamente en todas sus componentes), e individualmente cada AOI es clasificado con un accuracy de 0.733 %

en promedio y la métrica de accuracy es de 0.588 (correcta predicción de las etiquetas activas).

Por lo tanto, se observa que la clasificación de estos vectores es posible con la "ingeniería de features" realizada durante el proceso, donde todas las características fueron extraídas del historial de comportamiento de la mirada de cada usuario. Al mismo tiempo se observa que las mejoras al disminuir el número de AOI's no es lo suficiente como para preferir un modelo de mayor simplificación. Sin embargo, nos da información de que la forma de asignación de las áreas semánticas de interés influye en la calidad de los resultados de la clasificación de estos vectores.

Para la segunda etapa se ha demostrado que la categorización de ventanas temporales en estados mentales creados a partir de señales fisio-psicológicas, permite discriminar entre las diferentes formas en las distribuciones de sacada de los usuarios, con lo cual se pueden generar distribuciones de probabilidad que se diferencien en las formas que tomas las trayectorias oculares al trasladar la mirada entre dos cuadrantes del estímulo. Dado que estas probabilidades son calculadas a partir de sacadas, el cálculo para los valores de trayectorias entre diferentes AOI's pueden tomar diferentes formas (combinaciones entre cuadrantes).

De las diferentes configuraciones de clusterización usadas para la categorización de las ventanas en estados mentales, se prefiere el uso del método de Kmean, creados con 20 variables fisio-psicológicas, creando 3 clases de estados mentales.

De la tercera etapa, se observa que es posible el entrenamiento de redes neuronales recurrentes para la predicción de secuencia de áreas de interés. Sin embargo, no se logra un entrenamiento generalizable a los conjuntos de testeo. Es decir, los patrones en las ventanas de predicción de los conjuntos en entrenamiento no necesariamente son iguales a los del conjunto de test.

Lo anterior puede ser debido a diferentes factores, por ejemplo el uso de información en cada paso temporal de predicción o las variables EEG usadas para la creación del primer estado oculto. En particular, se observa un sobre-entrenamiento que puede ser tratado con técnicas de dropout o mejoras en los datos de entrada del modelo.

Por lo tanto, las principales contribuciones de este primer modelo consisten en:

- La creación del concepto de "intención de visita", y un método de predicción que permite generar resultados mayores al 70 % en cada AOI clasificada independientemente y 30 % para los vectores completos.
- Se ha mostrado que el uso de probabilidades entre las trayectorias de la mirada entre diferentes Áreas de interés, permiten generar una red RNN posible de entrenar, pero que no es generalizable en los conjuntos de test
- Se ha generado un primer enfoque de predicción que puede ser mejorado a partir del uso de otras estructuras más complejas tales como encoder-decoder.

El segundo modelo, es decir, la predicción de fijaciones a nivel píxel, muestra resultados favorables en la predicción de mapas de saliencia para cada usuario, calculados a partir de su comportamiento de la mirada en el pasado. Utilizando métricas de evaluación de mapas de saliencia, se demuestra que las distribuciones de probabilidad calculadas en ventanas muy

pequeñas, a partir de las distribuciones de diferencia de coordenadas Δx y Δy logran buenos resultados, sin conocer información de la ventana de predicción más allá de la coordenada en la cual comienza.

Es importante destacar el enfoque utilizado en esta modelación, donde se demuestra la buena calidad de los resultados obtenidos al utilizar mezclas de distribuciones gaussianas para generar los mapas de saliencia, y un enfoque de predicción de datos de mirada, más que la predicción de secuencias de fijaciones y sus coordenadas en los estímulos.

Dado los buenos resultados a nivel desplazamiento de la mirada, se desarrollan modelos de predicción de AOI's mediante dicha información. Sin embargo, los resultados tanto para la predicción de secuencias de AOI's como para la predicción del próximo AOI, no logran resultados esperados.

Para la predicción del próximo AOI, se observa que el modelo no es generalizable cuando se combinan los datos de mirada (coordenadas, diferenciales de desplazamiento y AOI alcanzados) con datos crudos de EEG. Sin embargo, al sólo utilizar los primeros, se logra la generalización. Por lo tanto se espera que este enfoque pueda ser mejorado a partir de cambios en las arquitecturas de redes encoder-decoder.

Un aprendizaje importante de este modelo, es que aunque la predicción de secuencias de áreas de interés resulta difícil debido a la poca generalización de patrones a los datos de test, la predicción de sólo el primer AOI de la secuencia parece factible, lo cual sería de gran utilidad para algún sistema recomendador que utilice estos modelos.

Por lo tanto, las principales contribuciones de este segundo modelo consisten en:

- La creación de un método de predicción de mapas de saliencia para ventanas de tiempo pequeñas a partir del historial de comportamiento de los datos de la mirada llevados a cabo por un usuario específico.
- Se demuestra la complejidad de la predicción a nivel AOI, complementando los resultados del primer modelo, usando información a nivel píxel en el sitio web.
- Se observa que una estructura encoder-decoder del modelo de predicción del próximo AOI es entrenada con resultados generalizables a los conjuntos de testeo, sin embargo, estos deben ser mejorados.

Para el modelo de predicción de indicadores mentales se han generado categorías para cada ventana de tiempo, para cada indicador, a partir de la clusterización de variables relacionadas en las ventanas de tiempo de los conjuntos de entrenamiento.

Para el caso de la excitación se han generado las clases alta y baja, utilizando las señales de los electrodos AF3, AF4, F3 y F4 del EEG, donde la mayoría de los casos recae en una de las clases. Para la valencia se han categorizado las ventanas en las clases positiva y negativa, utilizando las señales de los electrodos F3 y F4 del EEG, manteniendo el balance en la cantidad de ventanas de tiempo pertenecientes a cada categoría. Finalmente, utilizando la media de la dilatación pupilar en cada ventana, se han categorizado estas en dos clases de carga cognitiva que balanceadas en cantidad de ventanas presentes en cada una.

Para los tres índices mentales, se han generado modelos de clasificación de secuencias, basado en estructuras de encoder-decoder, donde dada las señales de los electrodos correspondientes o de la dilatación pupilar, dependiendo del caso, se busca la predicción de la clase a la cual pertenece cada ventana.

Del caso de la excitación se observa que el modelo de clasificación no logra los resultados deseados. Esto puede deberse a la arquitectura usada, a la información suministrada al modelo o a la forma de las señales. Al mirar los ratios de excitación (calculadas a partir de los electrodos AF3, AF4, F3 y F4), se observa que estos tienden a estar la mayor parte del tiempo en la ventana en torno a un valor constante con pequeñas variaciones. Sin embargo, parece ser que estos valores no se representan por los valores de las ventanas de información. Por lo tanto, queda la discusión si es posible la predicción de estados de excitación o si esta depende de conocer y predecir directamente los cambios en el estímulo (sitio web).

Para el caso de la valencia, se ha usado una celda LSTM para la decodificación de los resultados. De esta manera, se observa que lo aprendido en la fase de entrenamiento es generalizable a los datos de testeo. Sin embargo, al evaluar los resultados, no se logran resultados de clasificación superiores a la linea base de 50 % de accuracy. Esto puede deberse a las arquitecturas utilizadas. Sin embargo, parece ser que las señales de valencia no tienden a tener naturaleza acumulativa, por lo cual, el uso de información del pasado no permite la predicción de los valores de valencia en las ventanas siguientes.

Para el caso de la predicción de la carga cognitiva, se prueban dos modelos, uno en el que se codifican los valores de dos electrodos de EEG y otro en el que los inputs corresponden a valores de la dilatación pupilar en las ventanas de información. Se observa que ninguno de los modelos entrega resultados favorables en la predicción, sin embargo es interesante observar que la predicción con señales de dilatación pupilar en el encoder presenta peor entrenamiento de la red que el caso de usar las señales de los electrodos del EEG.

En conclusión, los modelos de predicción de indicadores mentales no logran resultados favorables pero entregan visiones acerca del comportamiento de estos en su predicción y abren el debate si es necesario conocer las características del estímulo más que las del propio usuario para su predicción.

7.1. Trabajo Futuro

Para el modelo de predicción de fijaciones a nivel semántico, la identificación de nuevas de áreas de interés permitiría construir un modelo de clasificación de los vectores de intención de visita mas robusto. Al mismo tiempo, se pueden mejorar los resultados, mediante la utilización de mejores características creadas a partir del historial de comportamiento pasado. Este primer modelo entrega una puerta de entrada al concepto de "intención de visita" el cual puede ser de utilidad en otros estímulos.

La creación de redes neuronales recurrentes puede ser mejorada mediante la integración de nuevas características en cada paso temporal, donde se puede tratar las probabilidades a priori generadas de manera distinta entre cada AOI, o integrar directamente la clase de estado mental en un modelo encoder decoder que permita una mejor generalización de los

patrones aprendidos en la fase de entrenamiento para los conjuntos de testeo.

Para el modelo de predicción de fijaciones a nivel píxel, se espera que la utilización de nuevas arquitecturas en las redes encoder-decoder permitan modelos generalizables para la predicción del próximo AOI en las ventanas de tiempo de predicción. En particular, llama la atención los posibles resultados que se pueden obtener al cambiar la estructura de decodificación de una red LSTM de un estado, a un método de clasificación a partir del estado oculto H_0 directamente, y diferentes combinaciones en los datos de entrada.

Además de lo anterior, es importante la integración de otros métodos de muestreo de resultados a partir de los modelos anteriores. En particular, este trabajo se basa en el muestreo de resultados a partir del método argmax (escoger los elementos de mayor probabilidad calculada), sin embargo, la generación de mapas de saliencia y secuencias de AOI's puede mejorarse usando métodos de muestreos de secuencia con ramificaciones, presentes en la literatura.

Para el modelo de predicción de indicadores mentales de los usuarios en las ventanas de tiempo de predicción, no se logran resultados favorables, sin embargo, se abre la puerta al estudio de la forma en que los indicadores de excitación, valencia, y carga cognitiva pueden ser predichos. En particular, abre la discusión si es posible generar modelos que predigan dichos indicadores en periodos cortos de tiempo, o simplemente, al ser mediciones fuertemente vinculadas al estímulo, no es posible hacerlo directamente de los datos de usuario, requiriendo conocer los cambios que el estímulo presenta.

Se espera que modelos de predicción usando variables como datos de entrada en vez de las señales crudas de cada electrodo, aplicando "ingeniería de features" permita un modelo con resultados más robustos en estas predicciones.

Bibliografía

[1] Mitul Kumar Ahirwal and Narendra D Londhe. Power spectrum analysis of eeg signals for estimating visual attention. *International Journal of computer applications*, 42(15), 2012.

[2] Ulf Ahlstrom and Ferne J Friedman-Berg. Using eye movement activity as a correlate of cognitive workload. *International Journal of Industrial Ergonomics*, 36(7):623–636, 2006.

[3] Rana Fayyaz Ahmad, Aamir Saeed Malik, Hafeez Ullah Amin, Nidal Kamel, and Faruque Reza. Classification of cognitive and resting states of the brain using eeg features. In *Medical Measurements and Applications (MeMeA), 2016 IEEE International Symposium on*, pages 1–5. IEEE, 2016.

[4] Alexandre Alahi, Vignesh Ramanathan, Kratarth Goel, Alexandre Robicquet, Amir A Sadeghian, Li Fei-Fei, and Silvio Savarese. Learning to predict human behavior in crowded scenes. In *Group and Crowd Behavior for Computer Vision*, pages 183–207. Elsevier, 2017.

[5] Hafeez Ullah Amin, Aamir Saeed Malik, Rana Fayyaz Ahmad, Nasreen Badruddin, Nidal Kamel, Muhammad Hussain, and Weng-Tink Chooi. Feature extraction and classification for eeg signals using wavelet transform and machine learning techniques. *Australasian Physical & Engineering Sciences in Medicine*, 38(1):139–149, 2015.

[6] Pavlo Antonenko, Fred Paas, Roland Grabner, and Tamara Van Gog. Using electroencephalography to measure cognitive load. *Educational Psychology Review*, 22(4):425–438, 2010.

[7] Eberhard App and Gunter Debus. Saccadic velocity and activation: development of a diagnostic tool for assessing energy regulation. *Ergonomics*, 41(5):689–697, 1998.

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[9] A Terry Bahill, Michael R Clark, and Lawrence Stark. The main sequence, a tool for studying human eye movements. *Mathematical Biosciences*, 24(3-4):191–204, 1975.

[10] Loris Bazzani, Hugo Larochelle, and Lorenzo Torresani. Recurrent mixture density

network for spatiotemporal visual attention. *arXiv preprint arXiv:1603.08199*, 2016.

[11] Jackson Beatty. Task-evoked pupillary responses, processing load, and the structure of processing resources. *Psychological bulletin*, 91(2):276, 1982.

[12] Mathias Benedek and Christian Kaernbach. A continuous measure of phasic electrodermal activity. *Journal of neuroscience methods*, 190(1):80–91, 2010.

[13] Mathias Benedek and Christian Kaernbach. Decomposition of skin conductance data by means of nonnegative deconvolution. *Psychophysiology*, 47(4):647–658, 2010.

[14] Alberto Betella, Riccardo Zucca, Ryszard Cetnarski, Alberto Greco, Antonio Lanatà, Daniele Mazzei, Alessandro Tognetti, Xerxes D Arsiwalla, Pedro Omedas, Danilo De Rossi, et al. Inference of human affective states from psychophysiological measurements extracted under ecologically valid conditions. *Frontiers in neuroscience*, 8:286, 2014.

[15] Christopher M Bishop. Mixture density networks. Technical report, Aston University, 1994.

[16] James W Bisley and Michael E Goldberg. Neural correlates of attention and distractibility in the lateral intraparietal area. *Journal of neurophysiology*, 95(3):1696–1717, 2006.

[17] Indu P Bodala, Yu Ke, Hasan Mir, Nitish V Thakor, and Hasan Al-Nashash. Cognitive workload estimation due to vague visual stimuli using saccadic eye movements. In *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE*, pages 2993–2996. IEEE, 2014.

[18] Ali Borji and Laurent Itti. State-of-the-art in visual attention modeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):185–207, 2013.

[19] Zdravko I Botev, Joseph F Grotowski, Dirk P Kroese, et al. Kernel density estimation via diffusion. *The Annals of Statistics*, 38(5):2916–2957, 2010.

[20] Gavin Brown, Adam Pocock, Ming-Jie Zhao, and Mikel Luján. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection. *Journal of machine learning research*, 13(Jan):27–66, 2012.

[21] Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark, 2015.

[22] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[23] John T Cacioppo and Curt A Sandman. Physiological differentiation of sensory and cognitive tasks as a function of warning, processing demands, and reported unpleasantness. *Biological Psychology*, 6(3):181–192, 1978.

[24] Genís Cardona and Noa Quevedo. Blinking and driving: the influence of saccades and cognitive workload. *Current eye research*, 39(3):239–244, 2014.

[25] Francisco Charte, Antonio J Rivera, María J del Jesus, and Francisco Herrera. Mlsmote: approaching imbalanced multilabel learning through synthetic instance generation. *Knowledge-Based Systems*, 89:385–397, 2015.

[26] Fang Chen, Jianlong Zhou, Yang Wang, Kun Yu, Syed Z Arshad, Ahmad Khawaji, and Dan Conway. *Robust Multimodal Cognitive Load Measurement*. Springer, 2016.

[27] Gang Chen. A gentle tutorial of recurrent neural network with error backpropagation. *arXiv preprint arXiv:1610.02583*, 2016.

[28] Siyuan Chen, Julien Epps, Natalie Ruiz, and Fang Chen. Eye activity as a measure of human mental effort in hci. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 315–318. ACM, 2011.

[29] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[30] Filipe Cristino, Sebastiaan Mathôt, Jan Theeuwes, and Iain D Gilchrist. Scanmatch: A novel method for comparing fixation sequences. *Behavior research methods*, 42(3):692–700, 2010.

[31] Tjerk de Greef, Harmen Lafeber, Herre van Oostendorp, and Jasper Lindenberg. Eye movement as indicators of mental workload to trigger adaptive automation. *Foundations of augmented cognition. Neuroergonomics and operational neuroscience*, pages 219–228, 2009.

[32] Arnaud Delorme and Scott Makeig. Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *Journal of neuroscience methods*, 134(1):9–21, 2004.

[33] Leandro L Di Stasi, Vanessa Álvarez-Valbuena, José J Cañas, Antonio Maldonado, Andrés Catena, Adoración Antolí, and Antonio Candido. Risk behaviour and mental workload: Multimodal assessment techniques applied to motorbike riding simulation. *Transportation research part F: traffic psychology and behaviour*, 12(5):361–370, 2009.

[34] Leandro L Di Stasi, Mauro Marchitto, Adoracíon Antolí, Thierry Baccino, and José J Cañas. Approximation of on-line mental workload index in atc simulated multitasks. *Journal of Air Transport Management*, 16(6):330–333, 2010.

[35] Leandro Luigi Di Stasi, Adoración Antolí, and José Juan Cañas. Main sequence: an index for detecting mental workload variation in complex tasks. *Applied ergonomics*, 42(6):807–813, 2011.

[36] Johan Engström, Emma Johansson, and Joakim Östlund. Effects of visual and cognitive

load in real and simulated motorway driving. *Transportation Research Part F: Traffic Psychology and Behaviour*, 8(2):97–120, 2005.

[37] Thomas Fritz, Andrew Begel, Sebastian C Müller, Serap Yigit-Elliott, and Manuela Züger. Using psycho-physiological measures to assess task difficulty in software development. In *Proceedings of the 36th international conference on software engineering*, pages 402–413. ACM, 2014.

[38] Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*. SIAM, 2007.

[39] Ramazan Gokay, Engin Masazade, Cagla Aydin, and Duygun Erol-Barkana. Emotional state and cognitive load analysis using features from bvp and sc sensors. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2015 IEEE International Conference on*, pages 178–183. IEEE, 2015.

[40] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[41] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[42] Alberto Greco, Gaetano Valenza, Antonio Lanata, Enzo Pasquale Scilingo, and Luca Citi. cvxeda: A convex optimization approach to electrodermal activity processing. *IEEE Transactions on Biomedical Engineering*, 63(4):797–804, 2016.

[43] David Ha and Douglas Eck. A neural representation of sketch drawings. *arXiv preprint arXiv:1704.03477*, 2017.

[44] Eija Haapalainen, SeungJun Kim, Jodi F Forlizzi, and Anind K Dey. Psycho-physiological measures for assessing cognitive load. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 301–310. ACM, 2010.

[45] Francisco Herrera, Francisco Charte, Antonio J Rivera, and María J Del Jesus. *Multilabel Classification: Problem Analysis, Metrics and Techniques*. Springer, 2016.

[46] Uwe Herwig, Peyman Satrapi, and Carlos Schönfeldt-Lecuona. Using the international 10-20 eeg system for positioning of transcranial magnetic stimulation. *Brain topography*, 16(2):95–99, 2003.

[47] Curtis S Ikehara and Martha E Crosby. Assessing cognitive load with physiological sensors. In *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*, pages 295a–295a. IEEE, 2005.

[48] Imotions. *EEG Pocket Guide*. Imotions, 2016.

[49] Imotions. *GSR Pocket Guide*. Imotions, 2016.

[50] Jan Jarvis, Felix Putze, Dominic Heger, and Tanja Schultz. Multimodal person inde-

pendent recognition of workload related biosignal patterns. In *Proceedings of the 13th international conference on multimodal interfaces*, pages 205–208. ACM, 2011.

[51] Ling Jian, Jundong Li, Kai Shu, and Huan Liu. Multi-label informed feature selection. In *IJCAI*, pages 1627–1633, 2016.

[52] Ming Jiang, Xavier Boix, Gemma Roig, Juan Xu, Luc Van Gool, and Qi Zhao. Learning to predict sequences of human visual fixations. *IEEE transactions on neural networks and learning systems*, 27(6):1241–1252, 2016.

[53] Angel Jimenez-Molina, Cristian Retamal, and Hernan Lira. Using psychophysiological sensors to assess mental workload during web browsing. *Sensors*, 18(2):458, 2018.

[54] Putai Jin. Toward a reconceptualization of the law of initial value. *Psychological Bulletin*, 111(1):176, 1992.

[55] Solwin Johnson and TD Subha. A study on eye fixation prediction and salient object detection in supervised saliency. *Materials Today: Proceedings*, 4(2):4169–4181, 2017.

[56] Tilke Judd, Frédo Durand, and Antonio Torralba. A benchmark of computational models of saliency to predict human fixations. Technical report, MIT, 2012.

[57] Keigo Kimura, Lu Sun, and Mineichi Kudo. Mlc toolbox: A matlab/octave library for multi-label classification. *arXiv preprint arXiv:1704.02592*, 2017.

[58] Wolfgang Klimesch. Eeg alpha and theta oscillations reflect cognitive and memory performance: a review and analysis. *Brain research reviews*, 29(2-3):169–195, 1999.

[59] Wolfgang Klimesch, Bärbel Schack, and Paul Sauseng. The functional significance of theta and upper alpha oscillations. *Experimental psychology*, 52(2):99–108, 2005.

[60] Jeff Klingner, Rakshit Kumar, and Pat Hanrahan. Measuring the task-evoked pupillary response with a remote eye tracker. In *Proceedings of the 2008 symposium on Eye tracking research & applications*, pages 69–72. ACM, 2008.

[61] Alexander Koenig, Domen Novak, Ximena Omlin, Michael Pulfer, Eric Perreault, Lukas Zimmerli, Matjaz Mihelj, and Robert Riener. Real-time closed-loop control of cognitive load in neurological patients during robot-assisted gait training. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 19(4):453–464, 2011.

[62] Arthur F Kramer. Physiological metrics of mental workload: A review of recent progress. *Multiple-task performance*, pages 279–328, 1991.

[63] Srinivas SS Kruthiventi, Vennela Gudisa, Jaley H Dholakiya, and R Venkatesh Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5781–5790, 2016.

[64] Andrew L Kun, Zeljko Medenica, Oskar Palinko, and Peter A Heeman. Utilizing pupil

diameter to estimate cognitive load changes during human dialogue: A preliminary study. In *Workshop on Cognitive Load and In-Vehicle Human-Machine Interaction, Salzburg Austria*, 2011.

[65] Hindra Kurniawan, Alexandr V Maslov, and Mykola Pechenizkiy. Stress detection from speech and galvanic skin response signals. In *Computer-Based Medical Systems (CBMS), 2013 IEEE 26th International Symposium on*, pages 209–214. IEEE, 2013.

[66] Miron B Kursa, Witold R Rudnicki, et al. Feature selection with the boruta package. *J Stat Softw*, 36(11):1–13, 2010.

[67] Tao Lin and Atsumi Imamiya. Evaluating usability based on multimodal information: an empirical study. In *Proceedings of the 8th international conference on Multimodal interfaces*, pages 364–371. ACM, 2006.

[68] Huiying Liu, Dong Xu, Qingming Huang, Wen Li, Min Xu, and Stephen Lin. Semantically-based human scanpath estimation with hmms. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3232–3239, 2013.

[69] Marcus Liwicki and Horst Bunke. Feature selection for hmm and blstm based handwriting recognition of whiteboard notes. *International journal of pattern recognition and artificial intelligence*, 23(05):907–923, 2009.

[70] HERNÁN FELIPE LIRA LÓPEZ. Medición de cargas cognitivas durante actividades de interacción humano computador en ambiente móvil usando sensores psicofisiológicos. In *Medición de Cargas Cognitivas Durante Actividades de Interacción Humano Computador en Ambiente Móvil Usando Sensores Psico-fisiológicos*, 2015.

[71] Yijuan Lu, Ira Cohen, Xiang Sean Zhou, and Qi Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 301–304. ACM, 2007.

[72] Rohit Mallick, David Slayback, Jon Touryan, Anthony J Ries, and Brent J Lance. The use of eye metrics to index cognitive workload in video games. In *Eye Tracking and Visualization (ETVIS), IEEE Second Workshop on*, pages 60–64. IEEE, 2016.

[73] Olivier Le Meur, Antoine Coutrot, Zhi Liu, Adrien Le Roch, Andrea Helo, and Pia Rama. Computational model for predicting visual fixations from childhood to adulthood. *arXiv preprint arXiv:1702.04657*, 2017.

[74] MLSMOTE: a synthetic minority oversampling technique for imbalanced multilabel classification. `http://http://simidat.ujaen.es/~research/MLSMOTE/index.html`. Accessed: 2017-10-27.

[75] Carter Mundell, Juan Pablo Vielma, and Tauhid Zaman. Predicting performance under stressful conditions using galvanic skin response. *arXiv preprint arXiv:1606.01836*, 2016.

[76] Arturo Nakasone, Helmut Prendinger, and Mitsuru Ishizuka. Emotion recognition from

electromyography and skin conductance. In *Proc. of the 5th International Workshop on Biosignal Interpretation*, pages 219–222, 2005.

[77] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in neural information processing systems*, pages 1813–1821, 2010.

[78] R. Nieuwenhuys. *El sistema nervioso central humano*. Editorial Medica Panamericana Sa de, 2009. v. 2.

[79] Nargess Nourbakhsh, Yang Wang, and Fang Chen. Gsr and blink features for cognitive load classification. In *IFIP Conference on Human-Computer Interaction*, pages 159–166. Springer, 2013.

[80] Nargess Nourbakhsh, Yang Wang, Fang Chen, and Rafael A Calvo. Using galvanic skin response for cognitive load measurement in arithmetic and reading tasks. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*, pages 420–423. ACM, 2012.

[81] Task Force of the European Society of Cardiology et al. Heart rate variability, standards of measurement, physiological interpretation, and clinical use. *circulation*, 93:1043–1065, 1996.

[82] Calvin KL Or and Vincent G Duffy. Development of a facial skin temperature-based methodology for non-intrusive mental workload measurement. *Occupational Ergonomics*, 7(2):83–94, 2007.

[83] Xufang Pang, Ying Cao, Rynson WH Lau, and Antoni B Chan. Directing user attention via visual flow on web designs. *ACM Transactions on Graphics (TOG)*, 35(6):240, 2016.

[84] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, 27(8):1226–1238, 2005.

[85] Bastian Pfleging, Drea K Fekety, Albrecht Schmidt, and Andrew L Kun. A model relating pupil diameter to mental workload and lighting conditions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 5776–5788. ACM, 2016.

[86] Felix Putze, Jan-Philip Jarvis, and Tanja Schultz. Multimodal recognition of cognitive workload for multitasking in the car. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3748–3751. IEEE, 2010.

[87] Rafael Ramirez and Zacharias Vamvakousis. Detecting emotion from eeg signals using the emotive epoc device. In *International Conference on Brain Informatics*, pages 175–184. Springer, 2012.

[88] Agusto Alegría H Retamal Contreras C Jiménez Molina A, Ruiz del Solar J. *Estudio del comportamiento de la carga cognitiva de usuarios que navegan en un sitio Web.*

Thesis, Universidad de Chile, 2017.

[89] Antonio Quintero Rincón, Marcelo Risk, and Sergio Liberczuk. Eeg preprocessing with hampel filters. *IEEE Latin Am. Trans. Córdoba, Argentina*, pages 13–15, 2012.

[90] Raphaëlle N Roy, Stephane Bonnet, Sylvie Charbonnier, and Aurélie Campagne. Mental fatigue and working memory load estimation: interaction and implications for eeg-based passive bci. In *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, pages 6607–6610. IEEE, 2013.

[91] Konstantinos Sechidis, Nikolaos Nikolaou, and Gavin Brown. Information theoretic feature selection in multi-label data through composite likelihood. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 143–152. Springer, 2014.

[92] Cornelia Setz, Bert Arnrich, Johannes Schumm, Roberto La Marca, Gerhard Tröster, and Ulrike Ehlert. Discriminating stress from cognitive load using a wearable eda device. *IEEE Transactions on information technology in biomedicine*, 14(2):410–417, 2010.

[93] Mohammad Shenasa, Mark E Josephson, and NA Mark Estes III. *The ECG handbook of contemporary challenges*. Cardiotext Publishing, 2015.

[94] Yu Shi, Natalie Ruiz, Ronnie Taib, Eric Choi, and Fang Chen. Galvanic skin response (gsr) as an index of cognitive load. In *CHI'07 extended abstracts on Human factors in computing systems*, pages 2651–2656. ACM, 2007.

[95] Greg J Siegle, Stuart R Steinhauer, V Andrew Stenger, Roma Konecky, and Cameron S Carter. Use of concurrent pupil dilation assessment to inform interpretation and analysis of fmri data. *Neuroimage*, 20(1):114–124, 2003.

[96] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[97] He Tang, Chuanbo Chen, and Yanan Bie. Prediction of human eye fixation by a single filter. *Journal of Signal Processing Systems*, 87(2):197–202, 2017.

[98] Yi-Fang Tsai, Erik Viirre, Christopher Strychacz, Bradley Chase, and Tzyy-Ping Jung. Task performance and eye activity: predicting behavior relating to cognitive workload. *Aviation, space, and environmental medicine*, 78(5):B176–B185, 2007.

[99] Pascal WM Van Gerven, Fred Paas, Jeroen JG Van Merriënboer, and Henk G Schmidt. Memory load and the cognitive pupillary response in aging. *Psychophysiology*, 41(2):167–174, 2004.

[100] Karl F Van Orden, Wendy Limbert, Scott Makeig, and Tzyy-Ping Jung. Eye activity correlates of workload during a visuospatial memory task. *Human factors*, 43(1):111–121, 2001.

[101] JA Veltman and AWK Gaillard. Physiological workload reactions to increasing levels of task difficulty. *Ergonomics*, 41(5):656–669, 1998.

[102] Lee-ming Wang, Vincent G Duffy, and Yingzi Du. A composite measure for the evaluation of mental workload. In *International Conference on Digital Human Modeling*, pages 460–466. Springer, 2007.

[103] Wei Wang, Cheng Chen, Yizhou Wang, Tingting Jiang, Fang Fang, and Yuan Yao. Simulating human saccadic scanpaths on natural images. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 441–448. IEEE, 2011.

[104] Glenn F Wilson. An analysis of mental workload in pilots during flight using multiple psychophysiological measures. *The International Journal of Aviation Psychology*, 12(1):3–18, 2002.

[105] Jie Xu, Yang Wang, Fang Chen, Ho Choi, Guanzhong Li, Siyuan Chen, and Sazzad Hussain. Pupillary response based cognitive workload index under luminance and emotional changes. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 1627–1632. ACM, 2011.

[106] Kota Yamaguchi, Alexander C Berg, Luis E Ortiz, and Tamara L Berg. Who are you with and where are you going? In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1345–1352. IEEE, 2011.

[107] Pega Zarjam, Julien Epps, and Nigel H Lovell. Beyond subjective self-rating: Eeg signal classification of cognitive workload. *IEEE Transactions on Autonomous Mental Development*, 7(4):301–310, 2015.

[108] Yu Zhao, Rennong Yang, Guillaume Chevalier, Rajiv C Shah, and Rob Romijnders. Applying deep bidirectional lstm and mixture density network for basketball trajectory prediction. *Optik-International Journal for Light and Electron Optics*, 158:266–272, 2018.

[109] Li Zhiwei and Shen Minfen. Classification of mental task eeg signals using wavelet packet entropy and svm. In *Electronic Measurement and Instruments, 2007. ICEMI'07. 8th International Conference on*, pages 3–906. IEEE, 2007.

[110] Jianlong Zhou, Syed Z Arshad, Simon Luo, Kun Yu, Shlomo Berkovsky, and Fang Chen. Indexing cognitive load using blood volume pulse features. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 2269–2275. ACM, 2017.

Capítulo 8

Anexos

En este capítulo se entrega mayor detalle de las métricas obtenidas en diferentes modelos a lo largo del trabajo y formulaciones útiles para comprender algunos modelos. En particular se muestran los siguientes resultados:

- Resultados clasificación multi-etiqueta: muestra los resultados de las diferentes configuraciones utilizadas para la clasificación de la intención de visita en la etapa uno del modelo de predicción de fijaciones a nivel semántico de AOI.
- Resultados asignación estado mental: muestra los resultados de las diferentes configuraciones de clusterización usada en la etapa dos del modelo de predicción de fijaciones a nivel semántico para la categorización en estados mentales de las ventanas de tiempo de cada usuario.
- Definición backpropagation RNN: muestra la formulación del método de entrenamiento de las redes RNN. Se entrega sólo como referencia para mejor entendimiento del modelo.
- Definición Backpropagation RNN-MDN: muestra la formulación del método de entrenamiento de las redes RNN con mezclas de densidades gaussianas. Se entrega sólo como referencia para mejor entendimiento del modelo.

8.1. Resultados clasificación multi-etiqueta

Tabla 8.1: Resultados Binary Relevance Ridge Regresion

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P0.75 F40	**M P0.25 F40**	**M P2 F40**	**M P2 F40**	**M P0.5 F40**	**M P1.75 F30**
auc	0,844 (0,005)	0,828 (0,006)	0,846 (0,006)	0,846 (0,006)	0,843 (0,006)	0,844 (0,006)
exact	0,276 (0,009)	0,262 (0,009)	0,278 (0,013)	0,280 (0,013)	0,274 (0,012)	0,285 (0,01)
hamming	0,782 (0,005)	0,763 (0,003)	0,782 (0,006)	0,783 (0,006)	0,777 (0,005)	0,782 (0,004)
macroF1	0,606 (0,007)	0,561 (0,008)	0,605 (0,008)	0,604 (0,006)	0,610 (0,005)	0,598 (0,005)
microF1	0,647 (0,008)	0,604 (0,004)	0,642 (0,008)	0,644 (0,008)	0,643 (0,008)	0,640 (0,007)
fscore	0,664 (0,012)	0,615 (0,01)	0,662 (0,012)	0,664 (0,012)	0,659 (0,012)	0,658 (0,011)
acc	0,560 (0,011)	0,517 (0,009)	0,559 (0,012)	0,561 (0,012)	0,556 (0,012)	0,556 (0,01)
pre	0,878 (0,005)	0,858 (0,007)	0,881 (0,006)	0,880 (0,006)	0,875 (0,006)	0,877 (0,007)
rank	0,873 (0,005)	0,852 (0,007)	0,874 (0,005)	0,874 (0,005)	0,871 (0,005)	0,872 (0,007)
one	0,125 (0,014)	0,156 (0,012)	0,118 (0,015)	0,119 (0,015)	0,134 (0,015)	0,124 (0,016)
cov	0,309 (0,008)	0,322 (0,009)	0,307 (0,008)	0,308 (0,008)	0,310 (0,007)	0,307 (0,008)
trainT	0,580 (0,006)	4,186 (0,084)	0,013 (0,006)	0,427 (0,007)	1,561 (0,034)	1,380 (0,051)
testT	0,000 0	0,000 0	0,000 0	0,000 0	0,000 0	0,000 0
AoI1	0,782 (0,016)	0,750 (0,015)	0,782 (0,016)	0,781 (0,016)	0,778 (0,016)	0,775 (0,016)
AoI2	0,687 (0,012)	0,676 (0,007)	0,679 (0,017)	0,687 (0,017)	0,672 (0,014)	0,686 (0,012)
AoI3	0,752 (0,007)	0,708 (0,014)	0,757 (0,006)	0,760 (0,009)	0,753 (0,008)	0,747 (0,009)
AoI4	0,717 (0,009)	0,698 (0,01)	0,716 (0,011)	0,718 (0,01)	0,715 (0,01)	0,725 (0,012)
AoI5	0,779 (0,013)	0,773 (0,011)	0,779 (0,014)	0,779 (0,013)	0,772 (0,016)	0,785 (0,013)
AoI6	0,972 (0,007)	0,972 (0,008)	0,976 (0,004)	0,974 (0,005)	0,974 (0,005)	0,971 (0,006)

Tabla 8.2: Resultados Binary Relevance KNN

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P30 F20	**M P20 F40**	**M P30 F10**	**M P20 F10**	**M P30 F15**	**M P20 F30**
auc	0,840 (0,007)	0,824 (0,008)	0,846 (0,005)	0,843 (0,005)	0,841 (0,004)	0,837 (0,007)
exact	0,281 (0,008)	0,266 (0,011)	0,297 (0,007)	0,303 (0,008)	0,293 (0,007)	0,291 (0,009)
hamming	0,774 (0,005)	0,764 (0,006)	0,786 (0,004)	0,786 (0,005)	0,779 (0,004)	0,777 (0,005)
macroF1	0,577 (0,008)	0,564 (0,009)	0,604 (0,008)	0,606 (0,009)	0,583 (0,009)	0,586 (0,008)
microF1	0,620 (0,006)	0,601 (0,011)	0,645 (0,004)	0,647 (0,006)	0,630 (0,005)	0,628 (0,008)
fscore	0,637 (0,011)	0,612 (0,015)	0,668 (0,007)	0,670 (0,009)	0,652 (0,007)	0,650 (0,011)
acc	0,538 (0,01)	0,516 (0,013)	0,568 (0,006)	0,571 (0,008)	0,554 (0,006)	0,551 (0,011)
pre	0,866 (0,006)	0,849 (0,007)	0,883 (0,005)	0,879 (0,004)	0,876 (0,003)	0,868 (0,007)
rank	0,851 (0,006)	0,829 (0,008)	0,867 (0,005)	0,858 (0,006)	0,858 (0,004)	0,846 (0,007)
one	0,133 (0,015)	0,156 (0,015)	0,108 (0,013)	0,110 (0,013)	0,120 (0,011)	0,125 (0,013)
cov	0,317 (0,009)	0,327 (0,008)	0,305 (0,008)	0,312 (0,009)	0,312 (0,008)	0,320 (0,006)
trainT	0,658 (0,005)	5,547 (0,171)	0,000 0	0,548 (0,02)	0,920 (0,004)	0,922 (0,03)
testT	0,488 (0,006)	0,619 (0,018)	0,489 (0,008)	0,481 (0,012)	0,452 (0,008)	0,531 (0,01)
AoI1	0,769 (0,015)	0,766 (0,019)	0,771 (0,014)	0,769 (0,019)	0,773 (0,017)	0,764 (0,015)
AoI2	0,673 (0,009)	0,673 (0,008)	0,702 (0,009)	0,702 (0,01)	0,697 (0,006)	0,680 (0,012)
AoI3	0,740 (0,006)	0,705 (0,013)	0,754 (0,01)	0,760 (0,008)	0,746 (0,005)	0,739 (0,007)
AoI4	0,718 (0,011)	0,693 (0,014)	0,728 (0,011)	0,730 (0,01)	0,717 (0,01)	0,722 (0,015)
AoI5	0,771 (0,012)	0,774 (0,014)	0,785 (0,014)	0,782 (0,013)	0,769 (0,012)	0,781 (0,014)
AoI6	0,971 (0,008)	0,974 (0,006)	0,972 (0,004)	0,972 (0,004)	0,971 (0,003)	0,975 (0,003)

Tabla 8.3: Resultados Binary Relevance SVM

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P0.2 F40	**M P1.4 F40**	**M P1 F40**	**M P0.2 F40**	**M P0.2 F20**	**M P0.2 F30**
exact	0,259 (0,011)	0,267 (0,005)	0,280 (0,012)	0,281 (0,012)	0,206 (0,011)	0,278 (0,013)
hamming	0,770 (0,005)	0,766 (0,004)	0,779 (0,005)	0,780 (0,005)	0,757 (0,005)	0,772 (0,006)
macroF1	0,604 (0,009)	0,566 (0,007)	0,606 (0,01)	0,606 (0,009)	0,634 (0,005)	0,583 (0,01)
microF1	0,637 (0,009)	0,598 (0,006)	0,634 (0,009)	0,634 (0,009)	0,639 (0,008)	0,624 (0,007)
fscore	0,651 (0,014)	0,610 (0,01)	0,654 (0,013)	0,654 (0,013)	0,646 (0,012)	0,642 (0,011)
acc	0,545 (0,013)	0,515 (0,009)	0,552 (0,013)	0,552 (0,013)	0,532 (0,012)	0,541 (0,011)
trainT	1,975 (0,006)	5,811 (0,085)	1,350 (0,005)	1,773 (0,005)	1,905 (0,008)	1,870 (0,039)
testT	0,005 (0,002)	0,009 (0,003)	0,005 (0,002)	0,008 (0,003)	0,008 (0,003)	0,002 (0,002)
AoI1	0,778 (0,014)	0,767 (0,016)	0,779 (0,015)	0,777 (0,014)	0,784 (0,015)	0,777 (0,015)
AoI2	0,669 (0,01)	0,673 (0,008)	0,683 (0,011)	0,681 (0,01)	0,661 (0,012)	0,679 (0,008)
AoI3	0,753 (0,006)	0,713 (0,011)	0,753 (0,006)	0,757 (0,008)	0,748 (0,007)	0,747 (0,009)
AoI4	0,672 (0,016)	0,700 (0,015)	0,706 (0,012)	0,712 (0,01)	0,682 (0,016)	0,723 (0,013)
AoI5	0,777 (0,011)	0,772 (0,013)	0,782 (0,013)	0,782 (0,013)	0,692 (0,015)	0,782 (0,012)
AoI6	0,970 (0,007)	0,973 (0,008)	0,971 (0,007)	0,969 (0,008)	0,976 (0,006)	0,923 (0,026)

Tabla 8.4: Resultados Clasificacion Chain Ridge Regresion

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P0.75 F40	**M P1.5 F35**	**M P2 F40**	**M P1.5 F40**	**M P2 F25**	**M P1.75 F30**
auc	0,844 (0,005)	0,829 (0,005)	0,846 (0,006)	0,846 (0,006)	0,841 (0,005)	0,844 (0,006)
exact	0,276 (0,009)	0,272 (0,009)	0,278 (0,013)	0,280 (0,013)	0,271 (0,013)	0,285 (0,01)
hamming	0,782 (0,005)	0,765 (0,004)	0,782 (0,006)	0,783 (0,006)	0,779 (0,005)	0,782 (0,004)
macroF1	0,606 (0,007)	0,565 (0,011)	0,605 (0,008)	0,604 (0,006)	0,614 (0,005)	0,598 (0,005)
microF1	0,647 (0,008)	0,603 (0,008)	0,642 (0,008)	0,644 (0,008)	0,644 (0,008)	0,640 (0,007)
fscore	0,664 (0,012)	0,614 (0,015)	0,662 (0,012)	0,664 (0,012)	0,663 (0,012)	0,658 (0,011)
acc	0,560 (0,011)	0,519 (0,012)	0,559 (0,012)	0,561 (0,012)	0,559 (0,012)	0,556 (0,01)
pre	0,878 (0,005)	0,853 (0,008)	0,881 (0,006)	0,880 (0,006)	0,877 (0,006)	0,877 (0,007)
rank	0,873 (0,005)	0,846 (0,008)	0,874 (0,005)	0,873 (0,005)	0,873 (0,005)	0,872 (0,007)
one	0,125 (0,014)	0,162 (0,019)	0,118 (0,015)	0,119 (0,015)	0,129 (0,015)	0,124 (0,016)
cov	0,309 (0,008)	0,328 (0,008)	0,307 (0,008)	0,308 (0,008)	0,307 (0,008)	0,307 (0,008)
trainT	0,755 (0,011)	4,588 (0,046)	0,177 (0,013)	0,602 (0,016)	1,159 (0,017)	1,392 (0,051)
testT	0,006 (0,006)	0,000 0	0,006 (0,006)	0,000 0	0,000 0	0,000 0
AoI1	0,782 (0,016)	0,753 (0,017)	0,782 (0,016)	0,781 (0,016)	0,784 (0,016)	0,775 (0,016)
AoI2	0,687 (0,012)	0,685 (0,006)	0,679 (0,017)	0,687 (0,017)	0,669 (0,014)	0,686 (0,012)
AoI3	0,752 (0,007)	0,705 (0,008)	0,757 (0,006)	0,760 (0,009)	0,752 (0,006)	0,747 (0,009)
AoI4	0,717 (0,009)	0,698 (0,016)	0,716 (0,011)	0,718 (0,01)	0,718 (0,012)	0,725 (0,012)
AoI5	0,779 (0,013)	0,776 (0,012)	0,779 (0,014)	0,779 (0,013)	0,776 (0,013)	0,785 (0,013)
AoI6	0,972 (0,007)	0,975 (0,008)	0,976 (0,004)	0,974 (0,005)	0,975 (0,005)	0,971 (0,006)

Tabla 8.5: Resultados Clasificacion Chain KNN

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P30 F40	**M P15 F35**	**M P15 F10**	**M P15 F10**	**M P30 F15**	**M P30 F40**
auc	0,839 (0,007)	0,818 (0,007)	0,839 (0,005)	0,838 (0,004)	0,841 (0,004)	0,841 (0,006)
exact	0,281 (0,009)	0,266 (0,009)	0,302 (0,006)	0,295 (0,007)	0,297 (0,006)	0,294 (0,009)
hamming	0,775 (0,005)	0,758 (0,006)	0,786 (0,004)	0,784 (0,004)	0,780 (0,005)	0,779 (0,004)
macroF1	0,585 (0,008)	0,581 (0,01)	0,613 (0,01)	0,617 (0,01)	0,579 (0,01)	0,589 (0,007)
microF1	0,630 (0,007)	0,610 (0,009)	0,660 (0,004)	0,654 (0,004)	0,628 (0,005)	0,633 (0,005)
fscore	0,647 (0,011)	0,616 (0,012)	0,679 (0,007)	0,676 (0,007)	0,652 (0,008)	0,655 (0,01)
acc	0,546 (0,01)	0,520 (0,012)	0,579 (0,006)	0,574 (0,006)	0,555 (0,006)	0,555 (0,009)
pre	0,867 (0,006)	0,835 (0,009)	0,878 (0,003)	0,878 (0,004)	0,875 (0,003)	0,868 (0,007)
rank	0,851 (0,006)	0,809 (0,009)	0,848 (0,004)	0,850 (0,005)	0,858 (0,004)	0,852 (0,006)
one	0,139 (0,014)	0,171 (0,015)	0,103 (0,012)	0,105 (0,011)	0,121 (0,011)	0,136 (0,014)
cov	0,316 (0,008)	0,344 (0,008)	0,319 (0,009)	0,319 (0,009)	0,313 (0,008)	0,314 (0,007)
trainT	0,597 (0,002)	4,428 (0,056)	0,000 0	0,445 (0,003)	0,863 (0,003)	0,772 (0,022)
testT	0,511 (0,003)	0,497 (0,006)	0,436 (0,004)	0,423 (0,004)	0,441 (0,007)	0,519 (0,006)
AoI1	0,772 (0,013)	0,721 (0,014)	0,763 (0,015)	0,767 (0,015)	0,773 (0,017)	0,768 (0,015)
AoI2	0,672 (0,01)	0,669 (0,013)	0,710 (0,008)	0,694 (0,008)	0,692 (0,008)	0,688 (0,008)
AoI3	0,739 (0,006)	0,696 (0,013)	0,757 (0,008)	0,752 (0,009)	0,752 (0,005)	0,735 (0,007)
AoI4	0,717 (0,011)	0,709 (0,011)	0,729 (0,01)	0,731 (0,009)	0,718 (0,011)	0,724 (0,014)
AoI5	0,776 (0,013)	0,780 (0,013)	0,790 (0,013)	0,791 (0,012)	0,773 (0,013)	0,780 (0,011)
AoI6	0,972 (0,008)	0,971 (0,008)	0,967 (0,004)	0,970 (0,003)	0,969 (0,003)	0,978 (0,004)

Tabla 8.6: Resultados Clasificacion Chain SVM

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P1 F40	**M P0.4 F40**	**M P1.6 F40**	**M P0.2 F40**	**M P0.8 F40**	**M P0.4 F30**
exact	0,277 (0,011)	0,273 (0,008)	0,289 (0,012)	0,289 (0,011)	0,263 (0,015)	0,291 (0,009)
hamming	0,772 (0,004)	0,765 (0,004)	0,777 (0,006)	0,780 (0,005)	0,768 (0,007)	0,779 (0,004)
macroF1	0,612 (0,011)	0,570 (0,012)	0,611 (0,011)	0,618 (0,01)	0,638 (0,015)	0,613 (0,007)
microF1	0,649 (0,01)	0,609 (0,011)	0,642 (0,009)	0,647 (0,009)	0,653 (0,012)	0,646 (0,005)
fscore	0,663 (0,014)	0,621 (0,017)	0,661 (0,012)	0,665 (0,014)	0,665 (0,016)	0,663 (0,009)
acc	0,560 (0,013)	0,525 (0,015)	0,561 (0,013)	0,563 (0,014)	0,560 (0,016)	0,564 (0,009)
trainT	1,961 (0,01)	5,741 (0,046)	1,322 (0,007)	1,753 (0,008)	2,867 (0,022)	1,870 (0,031)
testT	0,003 (0,002)	0,006 (0,003)	0,005 (0,002)	0,006 (0,003)	0,002 (0,002)	0,005 (0,002)
AoI1	0,782 (0,015)	0,763 (0,013)	0,776 (0,015)	0,776 (0,014)	0,779 (0,017)	0,775 (0,015)
AoI2	0,681 (0,012)	0,682 (0,009)	0,685 (0,011)	0,693 (0,009)	0,677 (0,015)	0,682 (0,009)
AoI3	0,750 (0,006)	0,714 (0,012)	0,758 (0,006)	0,762 (0,005)	0,749 (0,009)	0,751 (0,007)
AoI4	0,687 (0,012)	0,691 (0,012)	0,708 (0,01)	0,705 (0,01)	0,688 (0,013)	0,721 (0,011)
AoI5	0,766 (0,011)	0,769 (0,013)	0,766 (0,013)	0,773 (0,012)	0,740 (0,019)	0,775 (0,014)
AoI6	0,966 (0,007)	0,972 (0,007)	0,972 (0,005)	0,970 (0,007)	0,973 (0,005)	0,970 (0,006)

Tabla 8.7: Resultados MLKNN

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P15 F20	**M P15 F40**	**M P15 F10**	**M P15 F10**	**M P15 F15**	**M P15 F20**
auc	0,838 (0,007)	0,826 (0,006)	0,846 (0,005)	0,844 (0,005)	0,841 (0,004)	0,840 (0,008)
exact	0,271 (0,008)	0,260 (0,007)	0,297 (0,009)	0,301 (0,009)	0,284 (0,006)	0,277 (0,012)
hamming	0,776 (0,004)	0,764 (0,006)	0,786 (0,004)	0,786 (0,005)	0,777 (0,004)	0,778 (0,006)
macroF1	0,599 (0,008)	0,566 (0,008)	0,609 (0,009)	0,610 (0,008)	0,591 (0,009)	0,597 (0,007)
microF1	0,636 (0,007)	0,612 (0,01)	0,655 (0,004)	0,653 (0,006)	0,637 (0,006)	0,639 (0,009)
fscore	0,651 (0,011)	0,617 (0,014)	0,676 (0,007)	0,675 (0,009)	0,656 (0,008)	0,656 (0,013)
acc	0,548 (0,01)	0,520 (0,013)	0,575 (0,006)	0,574 (0,009)	0,556 (0,007)	0,554 (0,012)
pre	0,867 (0,006)	0,850 (0,005)	0,883 (0,005)	0,882 (0,005)	0,877 (0,004)	0,874 (0,01)
rank	0,848 (0,007)	0,833 (0,006)	0,866 (0,005)	0,862 (0,006)	0,855 (0,004)	0,858 (0,008)
one	0,133 (0,016)	0,162 (0,012)	0,109 (0,012)	0,109 (0,011)	0,115 (0,011)	0,119 (0,019)
cov	0,319 (0,009)	0,327 (0,008)	0,307 (0,009)	0,308 (0,009)	0,313 (0,008)	0,314 (0,007)
trainT	0,706 (0,011)	5,033 (0,116)	0,000 0	0,538 (0,008)	1,092 (0,026)	0,911 (0,042)
testT	0,091 (0,003)	0,092 (0,002)	0,078 (0,002)	0,081 (0,002)	0,094 (0,004)	0,080 (0,003)
AoI1	0,771 (0,014)	0,749 (0,015)	0,767 (0,015)	0,767 (0,017)	0,773 (0,017)	0,780 (0,017)
AoI2	0,677 (0,012)	0,665 (0,012)	0,711 (0,009)	0,708 (0,01)	0,693 (0,008)	0,678 (0,017)
AoI3	0,742 (0,009)	0,721 (0,009)	0,757 (0,007)	0,756 (0,007)	0,743 (0,007)	0,749 (0,011)
AoI4	0,718 (0,011)	0,707 (0,017)	0,728 (0,012)	0,732 (0,011)	0,711 (0,01)	0,713 (0,013)
AoI5	0,775 (0,012)	0,773 (0,012)	0,782 (0,013)	0,781 (0,013)	0,770 (0,016)	0,774 (0,012)
AoI6	0,973 (0,007)	0,968 (0,008)	0,970 (0,004)	0,971 (0,004)	0,970 (0,004)	0,975 (0,004)

Tabla 8.8: Resultados Binary Relevance RR, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P0.25 F35	**M P1.5 F40**	**M P0.25 F40**	**M P1.25 F40**	**M P0.25 F40**	**M P0.5 F40**
auc	0,805 (0,006)	0,791 (0,011)	0,802 (0,008)	0,801 (0,007)	0,800 (0,009)	0,805 (0,007)
exact	0,307 (0,015)	0,282 (0,013)	0,294 (0,012)	0,296 (0,009)	0,292 (0,02)	0,311 (0,011)
hamming	0,727 (0,007)	0,708 (0,011)	0,724 (0,006)	0,724 (0,005)	0,718 (0,009)	0,729 (0,007)
macroF1	0,684 (0,009)	0,657 (0,015)	0,678 (0,008)	0,678 (0,007)	0,674 (0,01)	0,686 (0,01)
microF1	0,693 (0,009)	0,672 (0,013)	0,688 (0,008)	0,688 (0,007)	0,683 (0,011)	0,698 (0,009)
fscore	0,691 (0,016)	0,666 (0,018)	0,685 (0,015)	0,687 (0,013)	0,680 (0,017)	0,695 (0,015)
acc	0,593 (0,016)	0,567 (0,017)	0,585 (0,014)	0,586 (0,013)	0,581 (0,018)	0,597 (0,014)
pre	0,902 (0,005)	0,885 (0,009)	0,903 (0,005)	0,903 (0,005)	0,900 (0,007)	0,900 (0,006)
rank	0,836 (0,008)	0,812 (0,012)	0,834 (0,008)	0,835 (0,008)	0,832 (0,01)	0,832 (0,009)
one	0,136 (0,019)	0,160 (0,02)	0,128 (0,019)	0,128 (0,02)	0,140 (0,022)	0,139 (0,021)
cov	0,332 (0,011)	0,349 (0,013)	0,335 (0,011)	0,334 (0,012)	0,335 (0,01)	0,333 (0,011)
trainT	0,484 0	3,153 (0,037)	0,013 (0,006)	0,364 (0,003)	1,370 (0,003)	0,719 (0,015)
testT	0,003 (0,002)	0,000 0	0,000 0	0,000 0	0,000 0	0,000 0
AoI1	0,778 (0,02)	0,747 (0,015)	0,774 (0,019)	0,779 (0,016)	0,775 (0,017)	0,767 (0,015)
AoI2	0,683 (0,013)	0,670 (0,011)	0,689 (0,012)	0,687 (0,011)	0,666 (0,021)	0,684 (0,015)
AoI3	0,752 (0,009)	0,731 (0,02)	0,741 (0,008)	0,742 (0,007)	0,744 (0,01)	0,752 (0,01)
AoI4	0,693 (0,009)	0,683 (0,016)	0,689 (0,01)	0,689 (0,012)	0,686 (0,01)	0,714 (0,013)

Tabla 8.9: Resultados Binary Relevance KNN, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P25 F35	**M P25 F40**	**M P15 F10**	**M P25 F10**	**M P15 F10**	**M P30 F40**
auc	0,805 (0,008)	0,783 (0,007)	0,800 (0,006)	0,805 (0,006)	0,799 (0,006)	0,802 (0,007)
exact	0,293 (0,013)	0,269 (0,01)	0,310 (0,009)	0,304 (0,01)	0,309 (0,011)	0,303 (0,012)
hamming	0,718 (0,008)	0,698 (0,005)	0,728 (0,005)	0,731 (0,004)	0,724 (0,005)	0,720 (0,007)
macroF1	0,680 (0,011)	0,654 (0,008)	0,682 (0,009)	0,680 (0,006)	0,679 (0,008)	0,668 (0,007)
microF1	0,692 (0,01)	0,666 (0,008)	0,693 (0,008)	0,691 (0,005)	0,691 (0,008)	0,681 (0,009)
fscore	0,688 (0,015)	0,662 (0,012)	0,692 (0,014)	0,691 (0,012)	0,687 (0,014)	0,680 (0,014)
acc	0,588 (0,014)	0,561 (0,01)	0,595 (0,012)	0,592 (0,011)	0,592 (0,012)	0,583 (0,013)
pre	0,900 (0,006)	0,881 (0,005)	0,903 (0,005)	0,908 (0,004)	0,902 (0,005)	0,893 (0,006)
rank	0,815 (0,008)	0,791 (0,006)	0,810 (0,007)	0,829 (0,007)	0,812 (0,007)	0,809 (0,008)
one	0,125 (0,019)	0,160 (0,016)	0,118 (0,02)	0,110 (0,018)	0,123 (0,019)	0,133 (0,02)
cov	0,337 (0,01)	0,351 (0,011)	0,337 (0,009)	0,331 (0,012)	0,337 (0,01)	0,343 (0,01)
trainT	0,461 (0,003)	2,992 (0,04)	0,000 0	0,355 (0,006)	0,605 (0,002)	0,700 (0,021)
testT	0,352 (0,003)	0,358 (0,006)	0,295 (0,004)	0,306 (0,004)	0,297 (0,005)	0,361 (0,004)
AoI1	0,756 (0,019)	0,749 (0,014)	0,775 (0,017)	0,782 (0,017)	0,773 (0,016)	0,762 (0,015)
AoI2	0,677 (0,013)	0,662 (0,015)	0,689 (0,008)	0,693 (0,008)	0,694 (0,011)	0,677 (0,016)
AoI3	0,740 (0,011)	0,717 (0,014)	0,755 (0,007)	0,752 (0,008)	0,747 (0,008)	0,748 (0,009)
AoI4	0,699 (0,01)	0,664 (0,012)	0,692 (0,01)	0,697 (0,01)	0,684 (0,01)	0,694 (0,009)

Tabla 8.10: Resultados Binary Relevance SVM, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P0.2 F40	**M P1.4 F40**	**M P0.2 F35**	**M P0.2 F30**	**M P0.2 F25**	**M P0.2 F40**
exact	0,305 (0,014)	0,287 (0,008)	0,303 (0,009)	0,304 (0,01)	0,297 (0,009)	0,307 (0,01)
hamming	0,724 (0,007)	0,706 (0,005)	0,723 (0,006)	0,723 (0,006)	0,719 (0,005)	0,723 (0,006)
macroF1	0,674 (0,012)	0,644 (0,009)	0,667 (0,008)	0,664 (0,009)	0,658 (0,009)	0,673 (0,01)
microF1	0,681 (0,012)	0,653 (0,01)	0,673 (0,008)	0,669 (0,008)	0,663 (0,008)	0,682 (0,01)
fscore	0,681 (0,019)	0,648 (0,014)	0,674 (0,013)	0,671 (0,014)	0,666 (0,013)	0,680 (0,016)
acc	0,584 (0,018)	0,552 (0,013)	0,576 (0,013)	0,574 (0,013)	0,568 (0,012)	0,583 (0,015)
trainT	1,481 (0,015)	4,159 (0,109)	0,733 (0,014)	0,983 (0,019)	1,445 (0,009)	1,697 (0,028)
testT	0,002 (0,002)	0,005 (0,002)	0,005 (0,002)	0,008 (0,003)	0,003 (0,002)	0,002 (0,002)
AoI1	0,770 (0,017)	0,761 (0,016)	0,775 (0,017)	0,776 (0,015)	0,780 (0,015)	0,761 (0,016)
AoI2	0,684 (0,011)	0,668 (0,007)	0,684 (0,01)	0,678 (0,01)	0,671 (0,012)	0,670 (0,013)
AoI3	0,749 (0,01)	0,721 (0,01)	0,741 (0,007)	0,743 (0,008)	0,739 (0,007)	0,749 (0,01)
AoI4	0,694 (0,01)	0,675 (0,01)	0,692 (0,01)	0,694 (0,011)	0,687 (0,011)	0,712 (0,012)

Tabla 8.11: Resultados Clasification Chain RR, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P0.5 F35	**M P1 F40**	**M P1.75 F40**	**M P2 F30**	**M P0.25 F30**	**M P0.5 F40**
auc	0,805 (0,006)	0,779 (0,01)	0,802 (0,008)	0,802 (0,007)	0,800 (0,007)	0,805 (0,007)
exact	0,307 (0,015)	0,280 (0,012)	0,294 (0,012)	0,296 (0,012)	0,295 (0,019)	0,311 (0,011)
hamming	0,727 (0,007)	0,702 (0,01)	0,724 (0,006)	0,723 (0,006)	0,720 (0,009)	0,729 (0,007)
macroF1	0,684 (0,009)	0,650 (0,014)	0,677 (0,008)	0,677 (0,008)	0,672 (0,011)	0,686 (0,01)
microF1	0,693 (0,009)	0,666 (0,013)	0,688 (0,008)	0,685 (0,008)	0,681 (0,011)	0,698 (0,009)
fscore	0,691 (0,016)	0,661 (0,018)	0,684 (0,015)	0,685 (0,014)	0,680 (0,018)	0,695 (0,015)
acc	0,593 (0,016)	0,562 (0,017)	0,584 (0,015)	0,584 (0,014)	0,581 (0,019)	0,597 (0,014)
pre	0,902 (0,005)	0,879 (0,01)	0,903 (0,005)	0,902 (0,005)	0,901 (0,005)	0,900 (0,006)
rank	0,836 (0,007)	0,804 (0,013)	0,834 (0,008)	0,835 (0,008)	0,834 (0,007)	0,832 (0,009)
one	0,136 (0,019)	0,176 (0,025)	0,128 (0,019)	0,133 (0,019)	0,136 (0,019)	0,139 (0,021)
cov	0,332 (0,011)	0,350 (0,01)	0,335 (0,011)	0,334 (0,011)	0,334 (0,01)	0,333 (0,011)
trainT	0,508 (0,012)	3,122 (0,018)	0,048 (0,007)	0,400 (0,011)	1,211 (0,017)	0,788 (0,025)
testT	0,000 0	0,000 0	0,000 0	0,000 0	0,000 0	0,005 (0,005)
AoI1	0,778 (0,02)	0,744 (0,018)	0,774 (0,019)	0,780 (0,017)	0,780 (0,016)	0,767 (0,015)
AoI2	0,683 (0,013)	0,674 (0,017)	0,690 (0,012)	0,672 (0,015)	0,662 (0,021)	0,684 (0,015)
AoI3	0,752 (0,009)	0,717 (0,016)	0,741 (0,008)	0,748 (0,008)	0,748 (0,008)	0,752 (0,01)
AoI4	0,693 (0,009)	0,674 (0,008)	0,688 (0,01)	0,691 (0,01)	0,689 (0,01)	0,714 (0,013)

Tabla 8.12: Resultados Clasification Chain KNN, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P30 F35	**M P25 F40**	**M P30 F10**	**M P30 F10**	**M P15 F10**	**M P25 F40**
auc	0,808 (0,008)	0,784 (0,011)	0,806 (0,008)	0,806 (0,007)	0,797 (0,007)	0,800 (0,007)
exact	0,306 (0,015)	0,272 (0,011)	0,310 (0,011)	0,313 (0,009)	0,307 (0,009)	0,296 (0,013)
hamming	0,720 (0,008)	0,696 (0,01)	0,730 (0,005)	0,733 (0,004)	0,721 (0,005)	0,715 (0,006)
macroF1	0,673 (0,01)	0,651 (0,013)	0,668 (0,009)	0,670 (0,007)	0,675 (0,008)	0,672 (0,008)
microF1	0,685 (0,01)	0,662 (0,014)	0,680 (0,008)	0,682 (0,006)	0,686 (0,008)	0,684 (0,009)
fscore	0,683 (0,015)	0,662 (0,019)	0,682 (0,014)	0,685 (0,013)	0,681 (0,014)	0,681 (0,014)
acc	0,585 (0,015)	0,562 (0,017)	0,585 (0,012)	0,588 (0,011)	0,586 (0,012)	0,582 (0,013)
pre	0,901 (0,006)	0,876 (0,009)	0,904 (0,005)	0,909 (0,004)	0,898 (0,006)	0,892 (0,007)
rank	0,819 (0,008)	0,775 (0,013)	0,825 (0,008)	0,833 (0,007)	0,807 (0,007)	0,805 (0,009)
one	0,124 (0,018)	0,160 (0,024)	0,117 (0,018)	0,110 (0,018)	0,128 (0,021)	0,136 (0,02)
cov	0,336 (0,011)	0,357 (0,012)	0,336 (0,011)	0,331 (0,011)	0,338 (0,01)	0,344 (0,009)
trainT	0,463 (0,003)	3,005 (0,057)	0,000 0	0,344 0	0,611 (0,002)	0,708 (0,015)
testT	0,352 (0,006)	0,356 (0,006)	0,308 (0,005)	0,298 (0,007)	0,306 (0,005)	0,363 (0,006)
AoI1	0,761 (0,019)	0,736 (0,022)	0,779 (0,017)	0,786 (0,015)	0,772 (0,017)	0,756 (0,014)
AoI2	0,677 (0,017)	0,661 (0,018)	0,690 (0,011)	0,691 (0,012)	0,691 (0,011)	0,676 (0,017)
AoI3	0,741 (0,01)	0,708 (0,011)	0,748 (0,008)	0,752 (0,01)	0,741 (0,009)	0,740 (0,008)
AoI4	0,699 (0,009)	0,680 (0,012)	0,704 (0,008)	0,703 (0,009)	0,681 (0,011)	0,689 (0,008)

Tabla 8.13: Resultados Clasification Chain SVM, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P1.4 F35	**M P1.4 F40**	**M P0.8 F35**	**M P1.8 F30**	**M P0.8 F25**	**M P2 F40**
exact	0,306 (0,012)	0,284 (0,013)	0,311 (0,009)	0,312 (0,011)	0,296 (0,011)	0,311 (0,011)
hamming	0,724 (0,007)	0,700 (0,008)	0,725 (0,005)	0,724 (0,006)	0,720 (0,005)	0,725 (0,007)
macroF1	0,684 (0,01)	0,655 (0,01)	0,675 (0,008)	0,673 (0,009)	0,671 (0,009)	0,684 (0,012)
microF1	0,691 (0,01)	0,665 (0,01)	0,682 (0,008)	0,681 (0,008)	0,677 (0,008)	0,692 (0,012)
fscore	0,691 (0,016)	0,659 (0,014)	0,682 (0,013)	0,680 (0,014)	0,677 (0,013)	0,688 (0,017)
acc	0,592 (0,015)	0,561 (0,014)	0,585 (0,012)	0,584 (0,013)	0,578 (0,013)	0,592 (0,017)
trainT	1,383 (0,018)	3,992 (0,032)	0,713 (0,024)	0,942 (0,029)	1,428 (0,01)	1,658 (0,03)
testT	0,008 (0,003)	0,003 (0,002)	0,003 (0,002)	0,002 (0,002)	0,000 0	0,000 0
AoI1	0,770 (0,018)	0,744 (0,015)	0,778 (0,016)	0,770 (0,013)	0,780 (0,016)	0,756 (0,019)
AoI2	0,679 (0,013)	0,675 (0,01)	0,686 (0,01)	0,680 (0,013)	0,665 (0,012)	0,684 (0,017)
AoI3	0,747 (0,01)	0,705 (0,017)	0,742 (0,009)	0,748 (0,007)	0,743 (0,007)	0,750 (0,009)
AoI4	0,702 (0,01)	0,675 (0,008)	0,696 (0,008)	0,699 (0,01)	0,694 (0,01)	0,710 (0,011)

Tabla 8.14: Resultados MLKNN, Reestructuración AOI

	F-Score	MIFS	MLJMI	MLMIM	MLMRMR	RFS
	M P15 F35	**M P15 F40**	**M P15 F10**	**M P15 F10**	**M P15 F10**	**M P15 F40**
auc	0,805 (0,008)	0,780 (0,01)	0,805 (0,007)	0,805 (0,006)	0,801 (0,007)	0,801 (0,007)
exact	0,293 (0,013)	0,286 (0,012)	0,306 (0,008)	0,304 (0,01)	0,302 (0,011)	0,292 (0,013)
hamming	0,718 (0,008)	0,699 (0,009)	0,730 (0,005)	0,731 (0,004)	0,725 (0,005)	0,716 (0,006)
macroF1	0,680 (0,011)	0,659 (0,01)	0,679 (0,008)	0,680 (0,006)	0,675 (0,009)	0,672 (0,007)
microF1	0,692 (0,01)	0,667 (0,01)	0,691 (0,008)	0,691 (0,005)	0,686 (0,008)	0,684 (0,008)
fscore	0,688 (0,015)	0,663 (0,014)	0,691 (0,014)	0,691 (0,012)	0,684 (0,014)	0,680 (0,013)
acc	0,588 (0,014)	0,566 (0,013)	0,593 (0,012)	0,592 (0,011)	0,586 (0,013)	0,581 (0,012)
pre	0,900 (0,006)	0,876 (0,009)	0,902 (0,005)	0,908 (0,004)	0,899 (0,005)	0,893 (0,007)
rank	0,815 (0,008)	0,780 (0,013)	0,822 (0,007)	0,829 (0,007)	0,820 (0,008)	0,805 (0,009)
one	0,125 (0,019)	0,172 (0,023)	0,116 (0,02)	0,110 (0,018)	0,129 (0,018)	0,134 (0,02)
cov	0,337 (0,01)	0,353 (0,013)	0,336 (0,011)	0,331 (0,012)	0,337 (0,01)	0,344 (0,01)
trainT	0,466 (0,002)	2,928 (0,043)	0,000 0	0,347 (0,002)	0,613 (0,002)	0,747 (0,036)
testT	0,088 (0,003)	0,095 (0,004)	0,077 (0,002)	0,078 0	0,080 (0,002)	0,094 (0,002)
AoI1	0,756 (0,019)	0,738 (0,015)	0,777 (0,018)	0,782 (0,017)	0,777 (0,017)	0,757 (0,013)
AoI2	0,677 (0,013)	0,651 (0,012)	0,690 (0,011)	0,693 (0,008)	0,692 (0,012)	0,678 (0,017)
AoI3	0,740 (0,011)	0,721 (0,014)	0,754 (0,007)	0,752 (0,008)	0,748 (0,009)	0,741 (0,008)
AoI4	0,699 (0,01)	0,687 (0,013)	0,698 (0,008)	0,697 (0,01)	0,682 (0,009)	0,687 (0,008)

8.2. Resultados asignación estado mental

A continuación se presentan las tablas de los resultados de clusterización usado para la categorización de los denominados "estados mentales" de los usuarios, usados en la etapa dos del modelo de predicción de fijaciones a nivel semántico de AOI's.

En las filas se muestran el número de clusters generados y en cada columna se muestra la cantidad de features usadas para generarlos. Para cada uno se muestran los valores de las métricas SC, S, XB, DI y ADI en promedio entre los diferentes folds de la validación cruzada, y entre paréntesis su desviación estándar.

Tabla 8.15: Resultados clusters Kmean, Variables 1 a 17

		Nvariables															
N clusters		2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2	SC	0,611 (0,012)	0,441 (0,009)	0,458 (0,009)	0,477 (0,012)	0,557 (0,013)	0,558 (0,013)	0,559 (0,013)	0,560 (0,013)	0,561 (0,013)	0,562 (0,013)	0,563 (0,013)	0,564 (0,013)	0,565 (0,013)	0,566 (0,014)	0,568 (0,013)	0,569 (0,013)
	S	3,E-04 (6E-06)	2,E-04 (4E-06)	2,E-04 (4E-06)	2,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)	3,E-04 (6E-06)
	XB	4,721 (0,354)	3,598 (0,115)	3,829 (0,216)	3,813 (0,091)	2,180 (0,019)	2,180 (0,018)	2,184 (0,018)	2,187 (0,018)	2,190 (0,017)	2,194 (0,017)	2,199 (0,017)	2,203 (0,017)	2,208 (0,016)	2,212 (0,017)	2,219 (0,017)	2,223 (0,017)
	DI	6,E-03 (2E-03)	1,E-02 (4E-03)	2,E-02 (2E-03)	2,E-02 (2E-03)	2,E-02 (2E-03)	3,E-03 (1E-04)	2,E-03 (1E-04)	2,E-03 (2E-04)	3,E-03 (4E-04)	3,E-03 (4E-04)	3,E-03 (5E-04)	2,E-03 (4E-04)	3,E-03 (4E-04)	2,E-03 (4E-04)	2,E-03 (4E-04)	2,E-03 (3E-04)
	ADI	6,E-02 (2E-02)	4,E-02 (1E-02)	4,E-02 (1E-02)	7,E-02 (2E-02)	6,E-02 (2E-02)	7,E-03 (2E-03)	6,E-03 (2E-03)	5,E-03 (2E-03)	5,E-03 (2E-03)	5,E-03 (2E-03)	5,E-03 (2E-03)	4,E-03 (2E-03)	4,E-03 (2E-03)	4,E-03 (2E-03)	4,E-03 (2E-03)	4,E-03 (1E-03)
3	SC	0,555 (0,007)	0,806 (0,034)	0,915 (0,047)	1,057 (0,049)	1,146 (0,03)	1,263 (0,057)	1,301 (0,096)	0,854 (0,061)	0,793 (0,008)	0,737 (0,006)	0,623 (0,004)	0,628 (0,005)	0,644 (0,005)	0,660 (0,006)	0,669 (0,005)	0,693 (0,006)
	S	4,E-04 (3E-05)	6,E-04 (8E-05)	6,E-04 (1E-04)	8,E-04 (1E-04)	7,E-04 (1E-04)	9,E-04 (2E-04)	8,E-04 (2E-04)	6,E-04 (9E-05)	6,E-04 (6E-06)	6,E-04 (5E-06)	5,E-04 (3E-06)	5,E-04 (3E-06)	5,E-04 (3E-06)	5,E-04 (4E-06)	5,E-04 (3E-06)	5,E-04 (4E-06)
	XB	18,402 (5,016)	8,429 (1,377)	4,611 (0,441)	4,114 (0,636)	3,001 (0,329)	3,376 (0,46)	2,605 (0,132)	2,387 (0,16)	2,375 (0,061)	2,325 (0,039)	2,347 (0,027)	2,090 (0,027)	2,064 (0,051)	2,012 (0,021)	2,033 (0,038)	1,981 (0,027)
	DI	4,E-03 (2E-03)	1,E-02 (3E-03)	2,E-02 (7E-03)	3,E-02 (1E-02)	4,E-02 (7E-03)	6,E-02 (1E-02)	7,E-02 (2E-03)	7,E-02 (7E-03)	8,E-02 (9E-03)	8,E-02 (1E-02)	8,E-02 (1E-02)	1,E-01 (2E-02)	1,E-01 (5E-03)	1,E-01 (3E-03)	1,E-01 (1E-02)	1,E-01 (1E-02)
	ADI	5,E-04 (6E-04)	3,E-04 (3E-04)	9,E-04 (8E-04)	1,E-03 (6E-04)	4,E-04 (4E-04)	9,E-05 (1E-04)	3,E-04 (3E-04)	4,E-04 (3E-04)	8,E-04 (2E-04)	7,E-04 (2E-04)	6,E-04 (1E-04)	6,E-04 (1E-04)	7,E-05 (5E-05)	5,E-04 (1E-04)	1,E-03 (2E-04)	1,E-03 (2E-04)
4	SC	0,430 (0,01)	0,652 (0,08)	0,798 (0,017)	0,969 (0,1)	1,113 (0,166)	0,876 (0,055)	0,688 (0,023)	0,702 (0,021)	0,615 (0,029)	0,641 (0,028)	0,633 (0,034)	0,642 (0,035)	0,650 (0,031)	0,672 (0,032)	0,575 (0,043)	0,591 (0,029)
	S	3,E-04 (9E-06)	5,E-04 (5E-05)	6,E-04 (2E-05)	7,E-04 (6E-05)	9,E-04 (9E-05)	6,E-04 (9E-05)	5,E-04 (2E-05)	5,E-04 (2E-05)	4,E-04 (1E-05)	4,E-04 (1E-05)	5,E-04 (1E-05)	4,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (2E-05)	4,E-04 (4E-05)	4,E-04 (2E-05)
	XB	24,621 (17,905)	9,127 (3,972)	5,079 (0,229)	4,458 (0,644)	3,447 (0,189)	2,892 (0,178)	2,713 (0,134)	2,511 (0,062)	2,482 (0,039)	2,343 (0,079)	2,406 (0,105)	2,348 (0,077)	2,394 (0,109)	2,362 (0,067)	2,418 (0,048)	2,255 (0,1)
	DI	1,E-03 (1E-03)	6,E-03 (3E-03)	2,E-02 (5E-03)	3,E-02 (2E-03)	4,E-02 (8E-03)	5,E-02 (1E-02)	5,E-02 (2E-02)	5,E-02 (1E-02)	7,E-02 (2E-02)	7,E-02 (1E-02)	3,E-02 (9E-03)	3,E-02 (4E-03)	3,E-02 (8E-03)	3,E-02 (9E-03)	4,E-02 (1E-02)	4,E-02 (1E-02)
	ADI	4,E-04 (2E-04)	4,E-04 (2E-04)	4,E-04 (2E-04)	3,E-04 (2E-04)	1,E-04 (2E-04)	8,E-05 (9E-05)	8,E-05 (2E-04)	1,E-04 (2E-04)	1,E-04 (2E-04)	2,E-04 (1E-04)	1,E-04 (3E-05)	7,E-05 (4E-05)	8,E-05 (3E-05)	2,E-05 (2E-05)	2,E-05 (1E-05)	3,E-05 (2E-05)
5	SC	1,843 (0,05)	1,873 (0,098)	2,414 (0,08)	2,616 (0,078)	3,011 (0,112)	1,086 (0,124)	1,026 (0,206)	0,725 (0,048)	0,674 (0,022)	0,566 (0,031)	0,563 (0,025)	0,595 (0,046)	0,555 (0,025)	0,579 (0,016)	0,558 (0,039)	0,555 (0,033)
	S	1,E-03 (2E-04)	1,E-03 (4E-05)	2,E-03 (8E-05)	2,E-03 (1E-04)	2,E-03 (9E-05)	7,E-04 (9E-05)	7,E-04 (2E-04)	5,E-04 (6E-05)	5,E-04 (1E-05)	4,E-04 (2E-05)	4,E-04 (2E-05)	4,E-04 (3E-05)	4,E-04 (1E-05)	4,E-04 (3E-05)	4,E-04 (3E-05)	4,E-04 (9E-06)
	XB	33,585 (42,775)	8,818 (2,577)	5,114 (0,612)	4,150 (1,304)	3,356 (0,169)	3,008 (0,162)	2,776 (0,2)	2,467 (0,049)	2,569 (0,193)	2,532 (0,131)	2,553 (0,186)	2,595 (0,147)	2,458 (0,052)	2,609 (0,195)	2,545 (0,146)	2,617 (0,171)
	DI	3,E-03 (1E-03)	7,E-03 (2E-03)	1,E-02 (4E-03)	3,E-02 (7E-03)	4,E-02 (6E-03)	4,E-02 (9E-03)	6,E-02 (8E-03)	7,E-02 (4E-03)	8,E-02 (1E-02)	8,E-02 (2E-02)	4,E-02 (1E-02)	3,E-02 (3E-03)	2,E-02 (2E-03)	2,E-02 (4E-03)	2,E-02 (3E-03)	2,E-02 (4E-03)
	ADI	6,E-04 (6E-04)	6,E-04 (4E-04)	2,E-04 (2E-04)	2,E-04 (2E-04)	2,E-04 (1E-04)	1,E-04 (1E-04)	2,E-04 (2E-04)	2,E-04 (3E-04)	1,E-04 (1E-04)	3,E-04 (1E-04)	1,E-04 (8E-05)	1,E-04 (8E-05)	9,E-05 (5E-05)	8,E-05 (5E-05)	5,E-05 (5E-05)	6,E-05 (3E-05)

Tabla 8.16: Resultados clusters Kmean, Variables 18 a 33

		Nvariables															
N clusters		18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
2	SC	0,571 (0,013)	0,577 (0,014)	0,584 (0,015)	0,591 (0,015)	0,603 (0,015)	0,613 (0,014)	0,624 (0,016)	0,638 (0,016)	0,654 (0,017)	0,668 (0,017)	0,682 (0,017)	0,697 (0,017)	0,710 (0,018)	0,724 (0,018)	0,745 (0,018)	0,759 (0,019)
	S	3,E-04 (6E-06)	3,E-04 (7E-06)	3,E-04 (7E-06)	3,E-04 (7E-06)	3,E-04 (7E-06)	3,E-04 (7E-06)	3,E-04 (8E-06)	3,E-04 (8E-06)	3,E-04 (8E-06)	3,E-04 (8E-06)	3,E-04 (8E-06)	3,E-04 (8E-06)	4,E-04 (8E-06)	4,E-04 (8E-06)	4,E-04 (9E-06)	4,E-04 (9E-06)
	XB	2,228 (0,017)	2,253 (0,021)	2,242 (0,026)	2,234 (0,028)	2,249 (0,014)	2,258 (0,021)	2,144 (0,014)	2,112 (0,039)	2,012 (0,016)	2,036 (0,016)	2,058 (0,034)	2,080 (0,049)	2,106 (0,071)	2,043 (0,044)	1,932 (0,058)	1,859 (0,029)
	DI	2,E-03 (4E-04)	4,E-03 (1E-03)	7,E-03 (1E-03)	7,E-03 (1E-03)	8,E-03 (2E-03)	9,E-03 (2E-03)	1,E-02 (3E-03)	1,E-02 (2E-03)	2,E-02 (2E-03)	2,E-02 (2E-03)	2,E-02 (3E-03)	2,E-02 (3E-03)	3,E-02 (3E-03)	3,E-02 (3E-03)	3,E-02 (3E-03)	3,E-02 (4E-03)
	ADI	4,E-03 (1E-03)	4,E-03 (1E-03)	4,E-03 (1E-03)	4,E-03 (1E-03)	4,E-03 (1E-03)	2,E-03 (7E-04)	2,E-03 (8E-04)	2,E-03 (8E-04)	3,E-03 (9E-04)	3,E-03 (9E-04)	2,E-03 (9E-04)	3,E-03 (9E-04)	2,E-03 (8E-04)	2,E-03 (8E-04)	2,E-03 (8E-04)	2,E-03 (8E-04)
3	SC	0,711 (0,005)	0,726 (0,005)	0,760 (0,006)	0,791 (0,005)	0,809 (0,005)	0,827 (0,005)	0,846 (0,005)	0,852 (0,007)	0,849 (0,007)	0,851 (0,007)	0,851 (0,008)	0,852 (0,007)	0,853 (0,007)	0,853 (0,007)	0,854 (0,007)	0,855 (0,006)
	S	5,E-04 (3E-06)	5,E-04 (3E-06)	6,E-04 (4E-06)	6,E-04 (3E-06)	6,E-04 (3E-06)	6,E-04 (3E-06)	6,E-04 (4E-06)	6,E-04 (4E-06)	6,E-04 (4E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	6,E-04 (4E-06)	6,E-04 (5E-06)	6,E-04 (4E-06)	6,E-04 (4E-06)
	XB	1,914 (0,038)	1,892 (0,027)	1,884 (0,018)	1,804 (0,018)	1,810 (0,014)	1,778 (0,022)	1,710 (0,01)	1,697 (0,01)	1,694 (0,009)	1,696 (0,009)	1,698 (0,009)	1,699 (0,009)	1,700 (0,008)	1,703 (0,009)	1,705 (0,009)	1,706 (0,009)
	DI	1,E-01 (1E-02)	2,E-01 (1E-02)	2,E-01 (1E-02)	2,E-01 (1E-02)	2,E-01 (2E-02)	2,E-01 (1E-02)	2,E-01 (1E-02)	2,E-01 (1E-02)	9,E-02 (2E-02)	9,E-02 (3E-03)	8,E-02 (4E-03)	6,E-02 (4E-03)	6,E-02 (3E-03)	6,E-02 (4E-03)	6,E-02 (3E-03)	5,E-02 (3E-03)
	ADI	1,E-03 (2E-04)	1,E-03 (2E-04)	2,E-03 (4E-04)	2,E-03 (3E-04)	2,E-03 (3E-04)	1,E-03 (6E-04)	1,E-03 (8E-04)	1,E-03 (9E-04)	6,E-04 (4E-04)	5,E-04 (4E-04)	5,E-04 (4E-04)	4,E-04 (4E-04)	4,E-04 (3E-04)	4,E-04 (3E-04)	4,E-04 (3E-04)	3,E-04 (3E-04)
4	SC	0,625 (0,051)	0,650 (0,051)	0,658 (0,067)	0,667 (0,045)	0,647 (0,067)	0,678 (0,049)	0,705 (0,068)	0,719 (0,084)	0,741 (0,025)	0,756 (0,055)	0,794 (0,052)	0,765 (0,046)	0,742 (0,085)	0,791 (0,027)	0,819 (0,104)	0,799 (0,043)
	S	4,E-04 (2E-05)	5,E-04 (2E-05)	5,E-04 (5E-05)	5,E-04 (3E-05)	4,E-04 (6E-05)	5,E-04 (4E-05)	5,E-04 (6E-05)	5,E-04 (7E-05)	5,E-04 (2E-05)	5,E-04 (3E-05)	6,E-04 (2E-05)	5,E-04 (5E-05)	5,E-04 (6E-05)	6,E-04 (2E-05)	6,E-04 (7E-05)	6,E-04 (4E-05)
	XB	2,159 (0,088)	2,012 (0,045)	1,935 (0,077)	1,854 (0,012)	1,867 (0,027)	1,862 (0,024)	1,816 (0,019)	1,778 (0,04)	1,734 (0,009)	1,738 (0,019)	1,741 (0,014)	1,755 (0,034)	1,755 (0,03)	1,765 (0,035)	1,778 (0,043)	1,784 (0,056)
	DI	4,E-02 (9E-03)	5,E-02 (8E-03)	5,E-02 (1E-02)	5,E-02 (2E-02)	5,E-02 (2E-02)	4,E-02 (1E-02)	5,E-02 (7E-03)	5,E-02 (9E-03)	5,E-02 (9E-03)	4,E-02 (9E-03)	5,E-02 (6E-03)	5,E-02 (5E-03)	4,E-02 (8E-03)	4,E-02 (5E-03)	4,E-02 (7E-03)	4,E-02 (6E-03)
	ADI	2,E-04 (7E-05)	2,E-04 (1E-04)	4,E-04 (3E-04)	7,E-04 (3E-04)	5,E-04 (2E-04)	5,E-04 (2E-04)	4,E-04 (2E-04)	4,E-04 (1E-04)	3,E-04 (1E-04)	2,E-04 (1E-04)	4,E-04 (5E-05)	3,E-04 (6E-05)	3,E-04 (9E-05)	1,E-04 (1E-04)	1,E-04 (2E-04)	1,E-04 (2E-04)
5	SC	0,496 (0,043)	0,520 (0,042)	0,538 (0,045)	0,571 (0,064)	0,578 (0,033)	0,592 (0,041)	0,579 (0,043)	0,597 (0,054)	0,615 (0,078)	0,560 (0,049)	0,568 (0,044)	0,554 (0,078)	0,570 (0,047)	0,580 (0,042)	0,592 (0,031)	0,599 (0,043)
	S	3,E-04 (3E-05)	4,E-04 (3E-05)	4,E-04 (3E-05)	4,E-04 (3E-05)	4,E-04 (2E-05)	4,E-04 (4E-05)	4,E-04 (5E-05)	4,E-04 (3E-05)	4,E-04 (5E-05)	4,E-04 (4E-05)	4,E-04 (3E-05)	4,E-04 (6E-05)	4,E-04 (3E-05)	4,E-04 (4E-05)	4,E-04 (2E-05)	4,E-04 (4E-05)
	XB	2,508 (0,124)	2,546 (0,25)	2,173 (0,095)	2,179 (0,139)	2,195 (0,09)	2,212 (0,171)	2,188 (0,095)	1,945 (0,079)	1,951 (0,07)	2,015 (0,081)	2,017 (0,081)	1,920 (0,083)	1,940 (0,08)	1,917 (0,072)	1,936 (0,07)	1,920 (0,076)
	DI	2,E-02 (5E-03)	2,E-02 (7E-03)	2,E-02 (6E-03)	3,E-02 (7E-03)	2,E-02 (7E-03)	2,E-02 (5E-03)	3,E-02 (5E-03)	3,E-02 (7E-03)	3,E-02 (6E-03)	3,E-02 (5E-03)	3,E-02 (4E-03)	3,E-02 (6E-03)	3,E-02 (6E-03)	3,E-02 (5E-03)	2,E-02 (4E-03)	3,E-02 (4E-03)
	ADI	5,E-05 (5E-05)	5,E-05 (4E-05)	5,E-05 (7E-05)	2,E-04 (7E-05)	1,E-04 (8E-05)	9,E-05 (5E-05)	9,E-05 (4E-05)	5,E-05 (3E-05)	6,E-05 (4E-05)	5,E-05 (7E-05)	6,E-05 (8E-05)	7,E-05 (9E-05)	6,E-05 (8E-05)	5,E-05 (5E-05)	4,E-05 (5E-05)	4,E-05 (6E-05)

Tabla 8.17: Resultados clusters Kmean, Variables 34 a 49

		Nvariables															
N clusters		34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
2	SC	0,760 (0,019)	0,765 (0,017)	0,782 (0,017)	0,782 (0,018)	0,815 (0,019)	0,819 (0,02)	0,839 (0,02)	0,853 (0,02)	0,872 (0,02)	0,887 (0,021)	0,896 (0,021)	0,909 (0,021)	0,938 (0,022)	0,952 (0,022)	0,966 (0,023)	0,985 (0,021)
	S	4,E-04 (9E-06)	4,E-04 (8E-06)	4,E-04 (8E-06)	4,E-04 (8E-06)	4,E-04 (9E-06)	4,E-04 (9E-06)	4,E-04 (1E-05)	4,E-04 (1E-05)	4,E-04 (1E-05)	4,E-04 (1E-05)	4,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)
	XB	1,863 (0,029)	1,868 (0,029)	1,841 (0,01)	1,841 (0,011)	1,802 (0,016)	1,802 (0,014)	1,779 (0,018)	1,776 (0,015)	1,734 (0,007)	1,715 (0,008)	1,721 (0,018)	1,721 (0,016)	1,728 (0,018)	1,720 (0,016)	1,710 (0,019)	1,689 (0,011)
	DI	3,E-02 (3E-03)	3,E-02 (3E-03)	3,E-02 (3E-03)	3,E-02 (4E-03)	3,E-02 (4E-03)	3,E-02 (4E-03)	3,E-02 (5E-03)	3,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (6E-03)	4,E-02 (5E-03)
	ADI	2,E-03 (8E-04)	2,E-03 (8E-04)	2,E-03 (8E-04)	2,E-03 (7E-04)	2,E-03 (7E-04)	2,E-03 (7E-04)	2,E-03 (7E-04)	2,E-03 (7E-04)	2,E-03 (7E-04)	1,E-03 (3E-04)	1,E-03 (3E-04)	1,E-03 (3E-04)	2,E-03 (5E-04)	2,E-03 (5E-04)	2,E-03 (5E-04)	2,E-03 (5E-04)
3	SC	0,856 (0,007)	0,857 (0,007)	0,858 (0,006)	0,860 (0,007)	0,860 (0,007)	0,864 (0,008)	0,865 (0,007)	0,866 (0,007)	0,869 (0,007)	0,881 (0,007)	0,947 (0,008)	0,951 (0,011)	0,950 (0,011)	0,914 (0,134)	0,931 (0,009)	0,947 (0,008)
	S	6,E-04 (4E-06)	6,E-04 (5E-06)	6,E-04 (4E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	6,E-04 (5E-06)	7,E-04 (5E-06)	7,E-04 (4E-05)	7,E-04 (4E-05)	7,E-04 (5E-05)	6,E-04 (1E-04)	7,E-04 (6E-06)	7,E-04 (6E-06)
	XB	1,709 (0,009)	1,711 (0,009)	1,713 (0,009)	1,715 (0,009)	1,717 (0,009)	1,719 (0,01)	1,721 (0,009)	1,722 (0,009)	1,729 (0,01)	1,732 (0,007)	1,724 (0,011)	1,712 (0,017)	1,714 (0,02)	1,721 (0,027)	1,735 (0,013)	1,728 (0,013)
	DI	5,E-02 (3E-03)	5,E-02 (3E-03)	5,E-02 (2E-03)	4,E-02 (3E-03)	4,E-02 (3E-03)	4,E-02 (3E-03)	4,E-02 (2E-03)	4,E-02 (3E-03)	4,E-02 (3E-03)	4,E-02 (3E-03)	4,E-02 (5E-03)	4,E-02 (4E-03)	4,E-02 (5E-03)	4,E-02 (7E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)
	ADI	3,E-04 (3E-04)	3,E-04 (3E-04)	3,E-04 (3E-04)	3,E-04 (3E-04)	3,E-04 (2E-04)	3,E-04 (2E-04)	3,E-04 (2E-04)	3,E-04 (2E-04)	3,E-04 (2E-04)	4,E-04 (2E-04)	8,E-05 (1E-04)	1,E-04 (1E-04)	8,E-05 (1E-04)	1,E-04 (1E-04)	7,E-05 (1E-04)	8,E-05 (8E-05)
4	SC	0,814 (0,075)	0,834 (0,07)	0,856 (0,068)	0,827 (0,074)	0,879 (0,027)	0,857 (0,03)	0,868 (0,081)	0,838 (0,082)	0,827 (0,08)	0,841 (0,106)	0,867 (0,067)	0,842 (0,079)	0,878 (0,026)	0,855 (0,087)	0,859 (0,094)	0,840 (0,158)
	S	6,E-04 (5E-05)	6,E-04 (7E-05)	6,E-04 (6E-05)	6,E-04 (8E-05)	6,E-04 (3E-05)	6,E-04 (3E-05)	6,E-04 (6E-05)	6,E-04 (8E-05)	6,E-04 (7E-05)	6,E-04 (8E-05)	6,E-04 (6E-05)	6,E-04 (7E-05)	6,E-04 (3E-05)	6,E-04 (8E-05)	6,E-04 (8E-05)	6,E-04 (1E-04)
	XB	1,774 (0,055)	1,721 (0,025)	1,715 (0,059)	1,698 (0,044)	1,685 (0,052)	1,732 (0,056)	1,694 (0,043)	1,689 (0,033)	1,699 (0,053)	1,710 (0,058)	1,686 (0,039)	1,743 (0,057)	1,715 (0,062)	1,721 (0,055)	1,703 (0,05)	1,731 (0,044)
	DI	4,E-02 (1E-02)	5,E-02 (6E-03)	5,E-02 (8E-03)	5,E-02 (8E-03)	5,E-02 (5E-03)	5,E-02 (5E-03)	5,E-02 (6E-03)	4,E-02 (4E-03)	4,E-02 (7E-03)	4,E-02 (7E-03)	4,E-02 (6E-03)	4,E-02 (8E-03)	4,E-02 (7E-03)	4,E-02 (7E-03)	4,E-02 (6E-03)	4,E-02 (7E-03)
	ADI	1,E-04 (2E-04)	2,E-04 (1E-04)	2,E-04 (1E-04)	2,E-04 (2E-04)	2,E-04 (1E-04)	2,E-04 (1E-04)	2,E-04 (4E-05)	2,E-04 (1E-04)	8,E-05 (7E-05)	2,E-04 (2E-04)	2,E-04 (2E-04)	1,E-04 (7E-05)	1,E-04 (3E-05)	2,E-04 (2E-04)	2,E-04 (2E-04)	1,E-04 (5E-05)
5	SC	0,597 (0,054)	0,589 (0,075)	0,620 (0,027)	0,633 (0,065)	0,678 (0,043)	0,642 (0,114)	0,662 (0,058)	0,725 (0,068)	0,706 (0,059)	0,672 (0,092)	0,687 (0,066)	0,715 (0,03)	0,740 (0,079)	0,707 (0,051)	0,700 (0,116)	0,711 (0,066)
	S	4,E-04 (5E-05)	4,E-04 (4E-05)	4,E-04 (2E-05)	5,E-04 (6E-05)	5,E-04 (4E-05)	5,E-04 (7E-05)	5,E-04 (4E-05)	5,E-04 (7E-05)	5,E-04 (4E-05)	5,E-04 (6E-05)	5,E-04 (4E-05)	5,E-04 (3E-05)	5,E-04 (6E-05)	5,E-04 (4E-05)	5,E-04 (9E-05)	5,E-04 (5E-05)
	XB	1,942 (0,07)	1,883 (0,088)	1,898 (0,07)	1,914 (0,086)	1,792 (0,059)	1,807 (0,037)	1,719 (0,05)	1,723 (0,066)	1,705 (0,069)	1,658 (0,065)	1,650 (0,063)	1,666 (0,063)	1,667 (0,061)	1,638 (0,058)	1,664 (0,049)	1,661 (0,059)
	DI	3,E-02 (4E-03)	3,E-02 (6E-03)	3,E-02 (5E-03)	3,E-02 (5E-03)	3,E-02 (4E-03)	3,E-02 (6E-03)	3,E-02 (3E-03)	3,E-02 (3E-03)	4,E-02 (3E-03)	4,E-02 (6E-03)	4,E-02 (4E-03)	4,E-02 (3E-03)	4,E-02 (6E-03)	4,E-02 (7E-03)	5,E-02 (9E-03)	4,E-02 (3E-03)
	ADI	6,E-05 (6E-05)	4,E-05 (6E-05)	3,E-05 (3E-05)	1,E-04 (7E-05)	1,E-04 (7E-05)	1,E-04 (8E-05)	9,E-05 (7E-05)	6,E-05 (4E-05)	4,E-05 (5E-05)	1,E-04 (9E-05)	2,E-04 (9E-05)	2,E-04 (9E-05)	2,E-04 (9E-05)	2,E-04 (9E-05)	1,E-04 (9E-05)	1,E-04 (1E-04)

Tabla 8.18: Resultados clusters Kmean, Variables 50 a 62

		Nvariables												
N clusters		50	51	52	53	54	55	56	57	58	59	60	61	62
2	SC	1,000 (0,023)	1,000 (0,023)	1,009 (0,021)	1,012 (0,023)	1,030 (0,024)	1,045 (0,024)	1,048 (0,022)	1,066 (0,024)	1,085 (0,028)	1,098 (0,028)	1,154 (0,024)	1,164 (0,028)	1,184 (0,029)
	S	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	5,E-04 (1E-05)	6,E-04 (1E-05)	6,E-04 (1E-05)	6,E-04 (1E-05)	6,E-04 (1E-05)
	XB	1,686 (0,012)	1,686 (0,011)	1,694 (0,014)	1,696 (0,016)	1,683 (0,008)	1,702 (0,008)	1,706 (0,008)	1,708 (0,008)	1,718 (0,007)	1,729 (0,014)	1,741 (0,007)	1,753 (0,008)	1,774 (0,007)
	DI	4,E-02 (6E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (6E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (3E-03)	4,E-02 (3E-03)	5,E-02 (5E-03)	5,E-02 (5E-03)	5,E-02 (5E-03)	5,E-02 (4E-03)
	ADI	2,E-03 (6E-04)	2,E-03 (6E-04)	2,E-03 (6E-04)	2,E-03 (6E-04)	2,E-03 (6E-04)	2,E-03 (6E-04)	2,E-03 (6E-04)	2,E-03 (7E-04)	2,E-03 (7E-04)	1,E-03 (3E-04)	9,E-04 (3E-04)	9,E-04 (3E-04)	9,E-04 (3E-04)
3	SC	0,963 (0,009)	0,973 (0,01)	0,982 (0,011)	0,992 (0,01)	0,999 (0,007)	1,002 (0,008)	1,010 (0,01)	1,023 (0,009)	1,038 (0,01)	1,058 (0,009)	1,069 (0,011)	1,079 (0,011)	1,104 (0,011)
	S	7,E-04 (6E-06)	7,E-04 (7E-06)	7,E-04 (7E-06)	7,E-04 (7E-06)	7,E-04 (5E-06)	7,E-04 (6E-06)	7,E-04 (6E-06)	7,E-04 (6E-06)	8,E-04 (7E-06)	8,E-04 (6E-06)	8,E-04 (8E-06)	8,E-04 (7E-06)	8,E-04 (7E-06)
	XB	1,724 (0,018)	1,720 (0,007)	1,732 (0,008)	1,737 (0,009)	1,728 (0,014)	1,726 (0,012)	1,722 (0,011)	1,736 (0,006)	1,742 (0,011)	1,762 (0,018)	1,769 (0,022)	1,764 (0,021)	1,784 (0,023)
	DI	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (5E-03)	4,E-02 (6E-03)	4,E-02 (6E-03)	4,E-02 (6E-03)	4,E-02 (6E-03)	5,E-02 (6E-03)	5,E-02 (7E-03)	5,E-02 (7E-03)	5,E-02 (7E-03)	5,E-02 (7E-03)	5,E-02 (7E-03)
	ADI	1,E-04 (8E-05)	2,E-04 (6E-05)	2,E-04 (6E-05)	1,E-04 (1E-04)	1,E-04 (1E-04)	2,E-04 (1E-04)	2,E-04 (8E-05)	3,E-04 (1E-04)	3,E-04 (1E-04)	3,E-04 (1E-04)	3,E-04 (1E-04)	4,E-04 (1E-04)	9,E-05 (6E-05)
4	SC	0,864 (0,094)	0,901 (0,045)	0,827 (0,094)	0,911 (0,085)	0,915 (0,113)	0,906 (0,086)	0,905 (0,069)	0,935 (0,02)	0,933 (0,098)	0,912 (0,071)	0,897 (0,07)	0,943 (0,049)	0,943 (0,097)
	S	6,E-04 (9E-05)	6,E-04 (6E-05)	6,E-04 (8E-05)	7,E-04 (8E-05)	6,E-04 (9E-05)	6,E-04 (7E-05)	6,E-04 (6E-05)	7,E-04 (2E-05)	7,E-04 (9E-05)	7,E-04 (4E-05)	7,E-04 (5E-05)	7,E-04 (3E-05)	7,E-04 (7E-05)
	XB	1,703 (0,043)	1,702 (0,046)	1,731 (0,031)	1,652 (0,018)	1,687 (0,04)	1,714 (0,047)	1,729 (0,04)	1,713 (0,046)	1,712 (0,026)	1,705 (0,026)	1,703 (0,041)	1,684 (0,044)	1,722 (0,046)
	DI	4,E-02 (6E-03)	4,E-02 (7E-03)	4,E-02 (6E-03)	4,E-02 (7E-03)	4,E-02 (7E-03)	4,E-02 (1E-02)	4,E-02 (5E-03)	4,E-02 (5E-03)	5,E-02 (7E-03)	4,E-02 (6E-03)	5,E-02 (7E-03)	4,E-02 (8E-03)	5,E-02 (8E-03)
	ADI	2,E-04 (2E-04)	1,E-04 (2E-04)	1,E-04 (2E-04)	1,E-04 (2E-04)	1,E-04 (2E-04)	1,E-04 (2E-04)	2,E-04 (1E-04)	2,E-04 (1E-04)	1,E-04 (6E-05)	4,E-05 (2E-05)	3,E-05 (2E-05)	4,E-05 (4E-05)	6,E-05 (3E-05)
5	SC	0,716 (0,088)	0,738 (0,075)	0,755 (0,061)	0,789 (0,009)	0,787 (0,035)	0,751 (0,085)	0,774 (0,114)	0,804 (0,044)	0,742 (0,104)	0,826 (0,038)	0,781 (0,075)	0,835 (0,067)	0,816 (0,096)
	S	5,E-04 (6E-05)	5,E-04 (4E-05)	5,E-04 (5E-05)	6,E-04 (2E-05)	6,E-04 (3E-05)	5,E-04 (7E-05)	5,E-04 (9E-05)	6,E-04 (3E-05)	5,E-04 (7E-05)	6,E-04 (4E-05)	5,E-04 (5E-05)	6,E-04 (6E-05)	6,E-04 (6E-05)
	XB	1,668 (0,031)	1,696 (0,038)	1,691 (0,029)	1,657 (0,041)	1,678 (0,033)	1,660 (0,039)	1,658 (0,036)	1,680 (0,048)	1,648 (0,028)	1,646 (0,032)	1,660 (0,033)	1,686 (0,037)	1,681 (0,036)
	DI	5,E-02 (9E-03)	5,E-02 (6E-03)	5,E-02 (5E-03)	5,E-02 (6E-03)	5,E-02 (4E-03)	5,E-02 (7E-03)	5,E-02 (9E-03)	5,E-02 (5E-03)	5,E-02 (5E-03)	5,E-02 (8E-03)	5,E-02 (8E-03)	5,E-02 (7E-03)	5,E-02 (7E-03)
	ADI	2,E-04 (9E-05)	1,E-04 (3E-05)	2,E-04 (3E-05)	2,E-04 (1E-04)	1,E-04 (9E-05)	8,E-05 (5E-05)	7,E-05 (6E-05)	9,E-05 (7E-05)	6,E-05 (4E-05)	1,E-04 (1E-04)	9,E-05 (9E-05)	8,E-05 (6E-05)	3,E-05 (2E-05)

Tabla 8.19: Resultados clusters Kmediod, Variables 1 a 17

		Nvariables															
N clusters		2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2	SC	0,214	0,239	0,536	0,623	0,551	0,521	0,632	0,715	0,573	0,616	0,657	0,613	0,547	0,56	0,558	0,668
		(0,015)	(0,041)	(0,39)	(0,429)	(0,257)	(0,061)	(0,259)	(0,335)	(0,145)	(0,093)	(0,096)	(0,119)	(0,147)	(0,061)	(0,141)	(0,331)
	S	1E-04	1E-04	3E-04	3E-04	3E-04	3E-04	3E-04	4E-04	3E-04	0,0003	3E-04	3E-04	0,00027	0,0003	3E-04	3E-04
		(8E-06)	(2E-05)	(2E-04)	(2E-04)	(1E-04)	(3E-05)	(1E-04)	(2E-04)	(7E-05)	(5E-05)	(5E-05)	(6E-05)	(7E-05)	(3E-05)	(7E-05)	(2E-04)
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	0,009	0,004	0,007	0,016	0,027	0,039	0,035	0,044	0,063	0,069	0,076	0,084	0,09	0,095	0,096	0,07
		(0,002)	(0,003)	(0,007)	(0,011)	(0,013)	(0,011)	(0,011)	(0,016)	(0,014)	(0,014)	(0,014)	(0,013)	(0,016)	(0,013)	(0,016)	(0,017)
	ADI	0,282	0,177	0,109	0,098	0,061	0,059	0,038	0,037	0,015	0,023	0,028	0,033	0,051	0,075	0,063	0,05
		(0,104)	(0,067)	(0,055)	(0,043)	(0,034)	(0,026)	(0,034)	(0,033)	(0,017)	(0,02)	(0,028)	(0,019)	(0,025)	(0,017)	(0,016)	(0,018)
3	SC	0,08	0,116	0,138	0,141	0,163	0,158	0,186	0,274	0,383	0,329	0,447	0,411	0,431	0,545	0,604	0,57
		(0,012)	(0,007)	(0,048)	(0,054)	(0,031)	(0,022)	(0,061)	(0,08)	(0,193)	(0,177)	(0,144)	(0,164)	(0,166)	(0,23)	(0,206)	(0,163)
	S	6E-05	8E-05	9E-05	9E-05	1E-04	1E-04	1E-04	2E-04	2E-04	0,0002	3E-04	2E-04	0,00026	0,0003	4E-04	3E-04
		(1E-05)	(4E-06)	(3E-05)	(4E-05)	(2E-05)	(2E-05)	(4E-05)	(5E-05)	(1E-04)	(1E-04)	(7E-05)	(9E-05)	(9E-05)	(1E-04)	(1E-04)	(9E-05)
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	0,001	0,005	0,006	0,002	0,003	0,003	0,006	0,006	0,009	0,016	0,017	0,022	0,029	0,031	0,035	0,04
		(0,001)	(0,002)	(0,003)	(0,001)	(0,001)	(0,001)	(0,002)	(0,002)	(0,003)	(0,004)	(0,004)	(0,005)	(0,008)	(0,004)	(0,004)	(0,005)
	ADI	0,211	0,075	0,026	0,009	0,007	0,007	0,005	0,016	0,02	0,016	0,005	0,008	0,006	0,006	0,008	0,007
		(0,067)	(0,027)	(0,017)	(0,006)	(0,004)	(0,004)	(0,002)	(0,008)	(0,007)	(0,01)	(0,003)	(0,004)	(0,006)	(0,005)	(0,005)	(0,005)
4	SC	0,029	0,048	0,125	0,245	0,333	0,589	0,612	0,543	0,582	0,637	0,524	0,487	0,352	0,356	0,426	0,483
		(0,037)	(0,036)	(0,086)	(0,158)	(0,131)	(0,118)	(0,159)	(0,192)	(0,225)	(0,289)	(0,236)	(0,172)	(0,187)	(0,158)	(0,139)	(0,1)
	S	2E-05	5E-05	1E-04	2E-04	2E-04	4E-04	4E-04	3E-04	4E-04	0,0004	3E-04	3E-04	0,00022	0,0002	3E-04	3E-04
		(3E-05)	(3E-05)	(7E-05)	(9E-05)	(8E-05)	(7E-05)	(9E-05)	(1E-04)	(1E-04)	(2E-04)	(1E-04)	(9E-05)	(1E-04)	(9E-05)	(9E-05)	(6E-05)
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	2E-04	4E-04	7E-04	0,002	0,003	0,006	0,01	0,013	0,013	0,0179	0,018	0,015	0,01813	0,0216	0,02	0,026
		(2E-04)	(5E-04)	(2E-04)	(4E-04)	(9E-04)	(9E-04)	(2E-03)	(2E-03)	(2E-03)	(2E-03)	(2E-03)	(6E-03)	(1E-03)	(3E-03)	(3E-03)	(3E-03)
	ADI	0,002	8E-04	0,004	0,003	0,003	0,004	0,004	0,003	0,004	0,0023	0,004	0,002	0,00235	0,0031	0,003	0,002
		(6E-04)	(2E-04)	(1E-02)	(6E-03)	(7E-03)	(1E-02)	(1E-02)	(6E-03)	(1E-02)	(6E-03)	(8E-03)	(7E-03)	(6E-03)	(7E-03)	(6E-03)	(4E-03)
5	SC	0,029	0,035	0,041	0,06	0,228	0,228	0,422	0,469	0,439	0,5063	0,516	0,429	0,4662	0,4856	0,448	0,365
		(0,041)	0,044	0,018	0,034	0,186	0,114	0,153	0,133	0,207	0,2012	0,13	0,185	0,13874	0,1287	0,211	0,176
	S	2E-05	3E-05	5E-05	6E-05	2E-04	1E-04	3E-04	3E-04	3E-04	0,0003	3E-04	3E-04	0,00029	0,0003	3E-04	2E-04
		(2E-05)	2E-05	2E-05	2E-05	1E-04	6E-05	9E-05	9E-05	1E-04	0,0001	8E-05	1E-04	8,1E-05	8E-05	1E-04	1E-04
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	4E-04	3E-04	3E-04	9E-04	0,002	0,003	0,004	0,007	0,01	0,0126	0,016	0,018	0,02125	0,026	0,031	0,031
		(2E-04)	1E-04	1E-04	2E-04	4E-04	5E-04	0,001	0,002	0,001	0,0015	0,002	0,003	0,00173	0,0021	0,004	0,005
	ADI	0,001	2E-04	2E-04	5E-04	0,001	6E-04	8E-04	0,001	0,002	0,0009	0,001	0,003	0,00138	0,0007	0,003	0,003
		(0,002)	1E-04	1E-04	3E-04	0,003	5E-04	9E-04	0,002	0,003	0,0008	0,001	0,007	0,00154	0,0012	0,006	0,006

Tabla 8.20: Resultados clusters Kmediod, Variables 18 a 33

		Nvariables															
N clusters		18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
2	SC	0,49 (0,173)	0,596 (0,206)	0,723 (0,399)	0,608 (0,168)	0,616 (0,25)	0,638 (0,454)	0,546 (0,244)	0,491 (0,134)	0,563 (0,125)	0,537 (0,175)	0,604 (0,201)	0,704 (0,261)	0,406 (0,168)	0,606 (0,152)	0,529 (0,105)	0,475 (0,279)
	S	2E-04 (9E-05)	3E-04 (1E-04)	4E-04 (2E-04)	3E-04 (8E-05)	3E-04 (1E-04)	3E-04 (2E-04)	3E-04 (1E-04)	2E-04 (7E-05)	3E-04 (6E-05)	3E-04 (9E-05)	3E-04 (1E-04)	4E-04 (1E-04)	2E-04 (8E-05)	3E-04 (8E-05)	3E-04 (5E-05)	2E-04 (1E-04)
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,063 (0,041)	0,041 (0,008)	0,027 (0,003)	0,034 (0,005)	0,033 (0,002)	0,042 (0,03)	0,04 (0,018)	0,039 (0,017)	0,036 (0,004)	0,032 (0,012)	0,028 (0,004)	0,028 (0,002)	0,037 (0,015)	0,027 (0,005)	0,031 (0,004)	0,049 (0,04)
	ADI	0,029 (0,008)	0,025 (0,007)	0,018 (0,005)	0,017 (0,004)	0,016 (0,006)	0,019 (0,005)	0,019 (0,006)	0,018 (0,009)	0,017 (0,009)	0,012 (0,008)	0,014 (0,005)	0,016 (0,004)	0,013 (0,005)	0,014 (0,003)	0,006 (0,005)	0,005 (0,006)
3	SC	0,602 (0,289)	0,705 (0,22)	0,478 (0,258)	0,709 (0,286)	0,542 (0,197)	0,416 (0,253)	0,603 (0,199)	0,453 (0,144)	0,413 (0,15)	0,333 (0,114)	0,37 (0,178)	0,475 (0,171)	0,41 (0,118)	0,441 (0,18)	0,437 (0,218)	0,356 (0,083)
	S	3E-04 (2E-04)	4E-04 (1E-04)	3E-04 (1E-04)	4E-04 (2E-04)	3E-04 (1E-04)	3E-04 (2E-04)	4E-04 (1E-04)	3E-04 (8E-05)	2E-04 (8E-05)	2E-04 (6E-05)	2E-04 (1E-04)	3E-04 (9E-05)	2E-04 (6E-05)	2E-04 (1E-04)	2E-04 (1E-04)	2E-04 (5E-05)
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,046 (0,006)	0,052 (0,005)	0,056 (0,008)	0,043 (0,008)	0,041 (0,008)	0,032 (0,004)	0,034 (0,004)	0,036 (0,004)	0,041 (0,006)	0,034 (0,006)	0,038 (0,006)	0,04 (0,005)	0,047 (0,006)	0,04 (0,005)	0,041 (0,006)	0,041 (0,004)
	ADI	0,007 (0,004)	0,006 (0,004)	0,01 (0,005)	0,008 (0,003)	0,007 (0,004)	0,004 (0,003)	0,004 (0,003)	0,003 (0,001)	0,003 (0,001)	0,003 (0,002)	0,005 (0,003)	0,006 (0,005)	0,008 (0,005)	0,005 (0,004)	0,007 (0,005)	0,003 (0,003)
4	SC	0,366 (0,132)	0,324 (0,163)	0,345 (0,138)	0,338 (0,079)	0,363 (0,135)	0,33 (0,087)	0,312 (0,067)	0,406 (0,077)	0,354 (0,2)	0,296 (0,141)	0,29 (0,102)	0,279 (0,104)	0,25 (0,087)	0,269 (0,04)	0,246 (0,135)	0,307 (0,072)
	S	2E-04 (8E-05)	2E-04 (1E-04)	2E-04 (7E-05)	2E-04 (5E-05)	2E-04 (8E-05)	2E-04 (5E-05)	2E-04 (4E-05)	2E-04 (6E-05)	2E-04 (1E-04)	2E-04 (9E-05)	2E-04 (6E-05)	2E-04 (7E-05)	2E-04 (5E-05)	2E-04 (2E-05)	1E-04 (7E-05)	2E-04 (4E-05)
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,023 (4E-03)	0,027 (5E-03)	0,03 (5E-03)	0,029 (2E-03)	0,03 (4E-03)	0,033 (5E-03)	0,035 (3E-03)	0,037 (4E-03)	0,042 (6E-03)	0,043 (6E-03)	0,049 (6E-03)	0,05 (4E-03)	0,054 (3E-03)	0,057 (4E-03)	0,057 (3E-03)	0,043 (4E-03)
	ADI	0,003 (6E-03)	0,002 (4E-03)	0,002 (3E-03)	0,002 (4E-03)	0,002 (3E-03)	0,003 (5E-03)	0,002 (4E-03)	0,002 (2E-03)	0,002 (2E-03)	0,003 (2E-03)	0,003 (3E-03)	0,003 (4E-03)	0,002 (2E-03)	0,003 (3E-03)	0,002 (2E-03)	0,002 (2E-03)
5	SC	0,381 0,127	0,305 0,163	0,346 0,134	0,316 0,075	0,318 0,149	0,332 0,132	0,285 0,121	0,309 0,133	0,298 0,094	0,267 0,083	0,308 0,093	0,26 0,063	0,271 0,034	0,283 0,112	0,211 0,088	0,192 0,077
	S	2E-04 8E-05	2E-04 1E-04	2E-04 7E-05	2E-04 4E-05	2E-04 8E-05	2E-04 8E-05	2E-04 7E-05	2E-04 7E-05	2E-04 5E-05	2E-04 5E-05	2E-04 5E-05	2E-04 3E-05	2E-04 2E-05	2E-04 6E-05	1E-04 5E-05	1E-04 4E-05
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,03 0,003	0,036 0,006	0,037 0,003	0,036 0,002	0,039 0,003	0,04 0,003	0,039 0,004	0,037 0,005	0,04 0,004	0,042 0,005	0,045 0,004	0,045 0,003	0,045 0,005	0,039 0,004	0,042 0,006	0,041 0,005
	ADI	0,003 0,007	0,002 0,005	9E-04 0,001	0,001 0,002	7E-04 6E-04	0,001 0,003	0,001 0,002	8E-04 0,001	8E-04 0,001	0,001 0,002	0,002 0,002	0,001 0,002	9E-04 0,001	6E-04 9E-04	6E-04 4E-04	0,001 0,002

Tabla 8.21: Resultados clusters Kmediod, Variables 34 a 49

		Nvariables															
N clusters		34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
2	SC	0,455	0,603	0,477	0,495	0,526	0,514	0,483	0,483	0,428	0,413	0,438	0,444	0,442	0,393	0,409	0,449
		(0,117)	(0,221)	(0,087)	(0,163)	(0,172)	(0,193)	(0,116)	(0,227)	(0,098)	(0,051)	(0,154)	(0,059)	(0,059)	(0,139)	(0,124)	(0,096)
	S	2E-04	3E-04	2E-04	2E-04	3E-04	3E-04	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04
		(6E-05)	(1E-04)	(4E-05)	(8E-05)	(9E-05)	(1E-04)	(6E-05)	(1E-04)	(5E-05)	(3E-05)	(8E-05)	(3E-05)	(3E-05)	(7E-05)	(6E-05)	(5E-05)
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	0,034	0,032	0,031	0,033	0,035	0,035	0,033	0,04	0,037	0,037	0,041	0,037	0,038	0,046	0,042	0,039
		(0,005)	(0,006)	(0,004)	(0,004)	(0,003)	(0,002)	(0,005)	(0,014)	(0,005)	(0,005)	(0,021)	(0,003)	(0,007)	(0,021)	(0,012)	(0,008)
	ADI	0,005	0,003	0,003	0,003	0,006	0,006	0,005	0,007	0,004	0,005	0,004	0,003	0,003	0,002	0,002	0,002
		(0,006)	(0,006)	(0,005)	(0,005)	(0,005)	(0,005)	(0,005)	(0,007)	(0,005)	(0,006)	(0,003)	(0,004)	(0,005)	(0,003)	(0,004)	(0,004)
3	SC	0,368	0,349	0,298	0,276	0,307	0,351	0,233	0,34	0,305	0,272	0,252	0,274	0,288	0,31	0,189	0,264
		(0,166)	(0,201)	(0,105)	(0,148)	(0,129)	(0,143)	(0,131)	(0,122)	(0,095)	(0,079)	(0,137)	(0,113)	(0,135)	(0,058)	(0,091)	(0,119)
	S	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04	1E-04	2E-04	2E-04	2E-04	1E-04	2E-04	2E-04	2E-04	1E-04	2E-04
		(8E-05)	(1E-04)	(6E-05)	(8E-05)	(7E-05)	(8E-05)	(7E-05)	(7E-05)	(5E-05)	(4E-05)	(8E-05)	(7E-05)	(7E-05)	(3E-05)	(5E-05)	(7E-05)
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	0,041	0,041	0,047	0,051	0,047	0,045	0,055	0,05	0,051	0,052	0,053	0,049	0,049	0,048	0,054	0,049
		(0,007)	(0,003)	(0,005)	(0,007)	(0,007)	(0,005)	(0,008)	(0,004)	(0,005)	(0,005)	(0,007)	(0,006)	(0,008)	(0,005)	(0,006)	(0,004)
	ADI	0,003	0,003	0,003	0,004	0,004	0,003	0,003	0,004	0,004	0,004	0,002	0,002	0,002	0,003	0,004	0,003
		(0,002)	(0,002)	(0,002)	(0,003)	(0,004)	(0,002)	(0,002)	(0,004)	(0,004)	(0,003)	(0,003)	(0,002)	(0,002)	(0,002)	(0,004)	(0,002)
4	SC	0,227	0,287	0,272	0,259	0,262	0,292	0,292	0,21	0,271	0,246	0,217	0,298	0,273	0,232	0,254	0,242
		(0,122)	(0,103)	(0,087)	(0,088)	(0,094)	(0,087)	(0,06)	(0,118)	(0,111)	(0,136)	(0,118)	(0,088)	(0,099)	(0,106)	(0,104)	(0,124)
	S	1E-04	2E-04	2E-04	2E-04	2E-04	2E-04	2E-04	1E-04	2E-04	1E-04	1E-04	2E-04	2E-04	1E-04	1E-04	1E-04
		(6E-05)	(6E-05)	(4E-05)	(5E-05)	(5E-05)	(5E-05)	(4E-05)	(6E-05)	(6E-05)	(7E-05)	(6E-05)	(4E-05)	(5E-05)	(6E-05)	(6E-05)	(7E-05)
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	0,051	0,045	0,048	0,046	0,049	0,045	0,047	0,052	0,048	0,05	0,051	0,051	0,053	0,052	0,048	0,047
		(6E-03)	(6E-03)	(7E-03)	(4E-03)	(7E-03)	(5E-03)	(5E-03)	(6E-03)	(4E-03)	(5E-03)	(6E-03)	(5E-03)	(4E-03)	(6E-03)	(6E-03)	(8E-03)
	ADI	0,002	0,003	0,002	0,002	0,002	0,002	0,002	0,001	0,002	0,001	0,001	5E-04	8E-04	7E-04	6E-04	7E-04
		(2E-03)	(2E-03)	(2E-03)	(2E-03)	(2E-03)	(2E-03)	(2E-03)	(1E-03)	(2E-03)	(4E-04)	(1E-03)	(5E-04)	(8E-04)	(7E-04)	(8E-04)	(7E-04)
5	SC	0,243	0,253	0,235	0,235	0,255	0,179	0,238	0,188	0,242	0,149	0,225	0,17	0,227	0,199	0,215	0,207
		0,074	0,094	0,092	0,095	0,051	0,065	0,089	0,072	0,036	0,094	0,068	0,087	0,066	0,074	0,108	0,096
	S	1E-04	2E-04	1E-04	1E-04	2E-04	1E-04	1E-04	1E-04	1E-04	9E-05	1E-04	1E-04	1E-04	1E-04	1E-04	1E-04
		4E-05	5E-05	5E-05	5E-05	3E-05	4E-05	5E-05	4E-05	2E-05	5E-05	4E-05	5E-05	3E-05	4E-05	6E-05	5E-05
	XB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
		-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DI	0,042	0,046	0,043	0,041	0,043	0,045	0,042	0,046	0,046	0,054	0,045	0,049	0,047	0,047	0,047	0,045
		0,004	0,004	0,005	0,004	0,003	0,005	0,005	0,005	0,003	0,009	0,005	0,007	0,005	0,006	0,006	0,005
	ADI	0,001	5E-04	8E-04	4E-04	0,001	0,001	0,001	4E-04	0,001	0,001	7E-04	3E-04	1E-03	5E-04	0,001	6E-04
		0,002	4E-04	5E-04	4E-04	0,002	0,002	0,002	4E-04	0,003	0,001	7E-04	4E-04	0,002	4E-04	0,002	6E-04

Tabla 8.22: Resultados clusters Kmediod, Variables 50 a 62

		Nvariables												
N clusters		50	51	52	53	54	55	56	57	58	59	60	61	62
2	SC	0,458 (0,168)	0,444 (0,045)	0,425 (0,083)	0,413 (0,134)	0,511 (0,351)	0,358 (0,151)	0,478 (0,16)	0,425 (0,129)	0,493 (0,072)	0,418 (0,07)	0,458 (0,068)	0,36 (0,139)	0,404 (0,141)
	S	2E-04 (8E-05)	2E-04 (2E-05)	2E-04 (4E-05)	2E-04 (7E-05)	3E-04 (2E-04)	2E-04 (8E-05)	2E-04 (8E-05)	2E-04 (6E-05)	2E-04 (4E-05)	2E-04 (4E-05)	2E-04 (3E-05)	2E-04 (7E-05)	2E-04 (7E-05)
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,047 (0,017)	0,042 (0,004)	0,042 (0,005)	0,047 (0,018)	0,059 (0,049)	0,055 (0,03)	0,044 (0,007)	0,053 (0,02)	0,046 (0,004)	0,049 (0,004)	0,045 (0,005)	0,058 (0,023)	0,052 (0,024)
	ADI	0,004 (0,005)	0,002 (0,004)	0,002 (0,005)	0,002 (0,005)	0,003 (0,006)	0,005 (0,001)	0,007 (0,005)	0,007 (0,004)	0,005 (0,004)	0,006 (0,005)	0,005 (0,005)	0,005 (0,004)	0,005 (0,005)
3	SC	0,294 (0,052)	0,283 (0,121)	0,226 (0,114)	0,314 (0,054)	0,249 (0,135)	0,281 (0,118)	0,31 (0,057)	0,245 (0,121)	0,32 (0,06)	0,277 (0,089)	0,282 (0,052)	0,297 (0,137)	0,233 (0,097)
	S	2E-04 (2E-05)	2E-04 (6E-05)	1E-04 (6E-05)	2E-04 (3E-05)	1E-04 (7E-05)	2E-04 (6E-05)	2E-04 (3E-05)	1E-04 (6E-05)	2E-04 (3E-05)	2E-04 (5E-05)	2E-04 (3E-05)	2E-04 (7E-05)	1E-04 (5E-05)
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,048 (0,004)	0,054 (0,008)	0,054 (0,014)	0,047 (0,006)	0,049 (0,007)	0,048 (0,007)	0,048 (0,004)	0,048 (0,005)	0,046 (0,005)	0,055 (0,018)	0,045 (0,004)	0,047 (0,004)	0,047 (0,005)
	ADI	0,003 (0,001)	0,003 (0,002)	0,002 (0,003)	0,002 (0,002)	0,002 (0,002)	0,002 (0,002)	0,002 (0,002)	0,002 (0,002)	0,001 (0,001)	0,001 (0,001)	0,002 (0,001)	0,002 (0,001)	0,002 (0,001)
4	SC	0,182 (0,125)	0,292 (0,101)	0,243 (0,088)	0,218 (0,086)	0,182 (0,096)	0,219 (0,101)	0,236 (0,098)	0,185 (0,068)	0,222 (0,071)	0,186 (0,095)	0,24 (0,085)	0,234 (0,062)	0,167 (0,108)
	S	1E-04 (7E-05)	2E-04 (6E-05)	1E-04 (5E-05)	1E-04 (5E-05)	1E-04 (6E-05)	1E-04 (6E-05)	1E-04 (5E-05)	1E-04 (4E-05)	1E-04 (4E-05)	1E-04 (6E-05)	1E-04 (5E-05)	1E-04 (3E-05)	1E-04 (6E-05)
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,051 (7E-03)	0,05 (5E-03)	0,053 (7E-03)	0,048 (4E-03)	0,052 (7E-03)	0,05 (7E-03)	0,048 (5E-03)	0,049 (5E-03)	0,048 (6E-03)	0,05 (5E-03)	0,045 (3E-03)	0,046 (4E-03)	0,047 (5E-03)
	ADI	6E-04 (6E-04)	4E-04 (3E-04)	4E-04 (3E-04)	3E-04 (3E-04)	4E-04 (2E-04)	5E-04 (7E-04)	3E-04 (3E-04)	9E-04 (6E-04)	4E-04 (4E-04)	6E-04 (7E-04)	8E-04 (1E-03)	5E-04 (5E-04)	6E-04 (4E-04)
5	SC	0,214 0,078	0,197 0,097	0,176 0,09	0,164 0,061	0,193 0,102	0,182 0,088	0,175 0,107	0,184 0,083	0,198 0,07	0,182 0,062	0,164 0,07	0,159 0,056	0,169 0,076
	S	1E-04 4E-05	1E-04 5E-05	1E-04 5E-05	1E-04 3E-05	1E-04 6E-05	1E-04 5E-05	1E-04 6E-05	1E-04 5E-05	1E-04 4E-05	1E-04 3E-05	1E-04 4E-05	1E-04 3E-05	1E-04 5E-05
	XB	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -	- -
	DI	0,047 0,006	0,049 0,004	0,049 0,007	0,048 0,003	0,051 0,007	0,052 0,009	0,054 0,008	0,05 0,006	0,052 0,009	0,05 0,007	0,049 0,007	0,047 0,004	0,047 0,007
	ADI	8E-04 4E-04	9E-04 0,001	6E-04 4E-04	6E-04 7E-04	8E-04 9E-04	4E-04 2E-04	3E-04 2E-04	4E-04 5E-04	6E-04 7E-04	7E-04 7E-04	6E-04 3E-04	6E-04 6E-04	5E-04 5E-04

8.3. Definición Backpropagation RNN

En esta sección se presenta el modelo matemático detrás del método BackPropagation Through Time aplicado a las redes neuronales recurrentes. Dada la definición de la función de pérdida en la ecuación 6.5, se calcula la derivada parcial respecto a la pérdida de cada paso temporal:

$$\frac{\partial L}{\partial L^{(t)}} = 1 \tag{8.1}$$

Utilizando la regla de la cadena se tiene que la derivada de la función de pérdida respecto a los outputs en cualquier tiempo t y para cualquier coordenada i es:

$$(\nabla_{o^{(t)}} L)_{\mathrm{i}} = \frac{\partial L}{\partial o_{\mathrm{i}}^{(t)}} = \frac{\partial L}{\partial L^{(t)}} \frac{\partial L^{(t)}}{\partial o_{\mathrm{i}}^{(t)}} = \hat{y}_{\mathrm{i}}^{(t)} - \mathbb{1}_{\mathrm{i},y^{(t)}} \tag{8.2}$$

En el último nodo de la red neuronal recurrente se tiene que $h^{(\tau)}$ solo afecta en el resultado de L a través de $o^{(\tau)}$, por lo tanto el gradiente resulta ser:

$$\nabla_{h^{(\tau)}} L = W_{hy}^{\top} \nabla_{o^{(\tau)}} L \tag{8.3}$$

La ecuación 8.3 sirve como condición de borde para la recurrencia de los estados ocultos. De esta manera, se puede iterar desde $t = \tau$ a $t = 0$. Se observa que para cada estado oculto $h^{(t)}$ con $t < \tau$, su valor influye en los cálculos de $o^{(t)}$ y $h^{(t+1)}$, con lo cual el gradiente se puede calcular como:

$$\begin{aligned} \nabla_{h^{(t)}} L &= \left(\frac{\partial h^{(t+1)}}{h^{(t)}}\right)^{\top} (\nabla_{h^{(t+1)}} L) + \left(\frac{\partial o^{(t)}}{h^{(t)}}\right)^{\top} (\nabla_{o^{(t)}} L) \\ &= W_{hh}^{\top} (\nabla_{h^{(t+1)}} L) \mathrm{diag}(1 - (h^{(t+1)})^2) + W_{hy}^{\top} (\nabla_{o^{(t)}} L) \end{aligned} \tag{8.4}$$

Donde $\mathrm{diag}(1-(h^{(t+1)})^2)$ es una matriz diagonal que contiene los valores $1-(h_{\mathrm{i}}^{(t+1)})^2$ asociada a la derivada de la tangente hiperbólica utilizada como función de activación.

Calculados los gradientes de los nodos internos, Se calculan los gradientes de los parámetros relacionados con la red neuronal recurrentes los que se muestran en las ecuaciones 8.5 a 8.9.

$$\nabla_c L = \sum_t \left(\frac{\partial o^{(t)}}{\partial c}\right)^{\top} \nabla_{o^{(t)}} L = \sum_t \nabla_{o^{(t)}} L \tag{8.5}$$

$$\nabla_b L = \sum_t \left(\frac{\partial h^{(t)}}{\partial b^{(t)}}\right)^{\top} \nabla_{h^{(t)}} L = \sum_t \mathrm{diag}(1 - (h^{(t)})^2) \nabla_{h^{(t)}} L \tag{8.6}$$

$$\nabla_{W_{hy}} L = \sum_t \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial o_{\mathrm{i}}^{(t)}}\right) \nabla_{W_{hy}} o_{\mathrm{i}}^{(t)} = \sum_t (\nabla_{o^{(t)}} L) h^{(t)\top} \tag{8.7}$$

$$\nabla_{W_{hh}} L = \sum_t \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(t)}}\right) \nabla_{W_{hh}} h_{\mathrm{i}}^{(t)} = \sum_t \mathrm{diag}(1 - (h^{(t)})^2)(\nabla_{h^{(t)}} L) h^{(t-1)\top} \tag{8.8}$$

$$\nabla_{W_{xh}} L = \sum_t \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(t)}}\right) \nabla_{W_{xh}} h_{\mathrm{i}}^{(t)} = \sum_t \mathrm{diag}(1 - (h^{(t)})^2)(\nabla_{h^{(t)}} L) x^{(t)\top} \tag{8.9}$$

Para el estado oculto incial $h^{(0)}$, se tienen las ecuaciones 8.10 a 8.17.

$$\nabla_{W_{\mathrm{i}h}} L = \sum_t \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(t)}} \right) \nabla_{W_{\mathrm{i}h}} h_{\mathrm{i}}^{(t)} = \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(0)}} \right) \nabla_{W_{\mathrm{i}h}} h_{\mathrm{i}}^{(0)} \tag{8.10}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) I^\top$$

$$\nabla_{W_{sh}} L = \sum_t \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(t)}} \right) \nabla_{W_{sh}} h_{\mathrm{i}}^{(t)} = \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(0)}} \right) \nabla_{W_{sh}} h_{\mathrm{i}}^{(0)} \tag{8.11}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) S A^\top$$

$$\nabla_{W_{\mathrm{e}h}} L = \sum_t \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(t)}} \right) \nabla_{W_{\mathrm{e}h}} h_{\mathrm{i}}^{(t)} = \sum_{\mathrm{i}} \left(\frac{\partial L}{\partial h_{\mathrm{i}}^{(0)}} \right) \nabla_{W_{\mathrm{e}h}} h_{\mathrm{i}}^{(0)} \tag{8.12}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) E E G^\top$$

$$\nabla_{\mathrm{d}} L = \sum_t \left(\frac{\partial h^{(t)}}{\partial \mathrm{d}^{(t)}} \right)^\top \nabla_{h^{(t)}} L = \mathrm{di}ag(1 - (h^{(0)})^2) \nabla_{h^{(0)}} L \tag{8.13}$$

$$\nabla_{\mathrm{e}} L = \sum_t \left(\frac{\partial h^{(t)}}{\partial \mathrm{e}} \right)^\top \nabla_{h^{(t)}} L \tag{8.14}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) W_{sh}^\top \mathrm{di}ag(1 - (SA)^2)$$

$$\nabla_f L = \sum_t \left(\frac{\partial h^{(t)}}{\partial f} \right)^\top \nabla_{h^{(t)}} L \tag{8.15}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) W_{\mathrm{e}h}^\top \mathrm{di}ag(1 - (EEG)^2)$$

$$\nabla_S L = \sum_t \left(\frac{\partial h^{(t)}}{\partial S} \right)^\top \nabla_{h^{(t)}} L \tag{8.16}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) W_{sh}^\top \mathrm{di}ag(1 - (SA)^2) s$$

$$\nabla_{W\mathrm{e}\mathrm{e}g} L = \sum_t \left(\frac{\partial h^{(t)}}{\partial W \mathrm{e}\mathrm{e}g} \right)^\top \nabla_{h^{(t)}} L \tag{8.17}$$

$$= \mathrm{di}ag(1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L) W_{\mathrm{e}h}^\top \mathrm{di}ag(1 - (EEG)^2) s$$

8.4. Definición Backpropagation LSTM

Para el caso de LSTM se observa que las conexiones desde $h^{(t)}$ a cada $y^{(t)}$ no cambian en relación al modelo RNN, por lo cual, los gradientes se mantienen.

$$\frac{\partial L}{\partial L^{(t)}} = 1 \qquad (\nabla_{z^{(t)}} L)_{\mathrm{i}} = \hat{y}_{\mathrm{i}}^{(t)} - \mathbb{1}_{\mathrm{i},y^{(t)}} \qquad \nabla_{h^{(\tau)}} L = W_{hy}^{\top} \nabla_{z^{(\tau)}} L \tag{8.18}$$

$$\nabla_{W_{hy}} L = \nabla_{z^{(t)}} L (h^{(t)})^{\top} \qquad \nabla_b L = \nabla_{z^{(t)}} L \tag{8.19}$$

Donde sólo se ha calculado el gradiente de la función de pérdida en relación al último estado oculto, en el último paso temporal τ. Para los gradientes en relación a las celdas LSTM, se obtienen las ecuaciones 8.20.

$$\nabla_{o^{(t)}} L = \tanh\left(c^{(t)}\right) \cdot \left(\nabla_{h^{(t)}} L\right) \tag{8.20a}$$

$$\nabla_{c^{(t)}} L = \left(1 - \tanh\left(c^{(t)}\right)^2\right) \cdot o^{(t)} \cdot \left(\nabla_{h^{(t)}} L\right) \tag{8.20b}$$

$$\nabla_{f^{(t)}} L = c^{(t-1)} \cdot \left(\nabla_{c^{(t)}} L\right) \tag{8.20c}$$

$$\nabla_{c^{(t-1)}} L \mathrel{+}= f^{(t)} \cdot \left(\nabla_{c^{(t)}} L\right) \tag{8.20d}$$

$$\nabla_{\mathrm{i}^{(t)}} L = \widetilde{c}^{\,(t)} \cdot \left(\nabla_{c(t)} L\right) \tag{8.20e}$$

$$\nabla_{\widetilde{c}^{\,(t)}} L = \mathrm{i}^{(t)} \cdot \left(\nabla_{c(t)} L\right) \tag{8.20f}$$

Con estos gradientes se calculan las derivadas parciales de la función de pérdida respecto a los parámetros del modelo, obteniendo las ecuaciones 8.21 para los parámetros entre los datos de entrada y las celdas, las ecuaciones 8.22 para las conexiones entre los estados ocultos y las celdas y las ecuaciones 8.23 para la estimación de los sesgos presentes en las ecuaciones de cada compuerta.

$$\nabla_{W_{xo}} L = \sum_t o^{(t)}(1 - o^{(t)}) x^{(t)} \nabla_{o^{(t)}} L \tag{8.21a}$$

$$\nabla_{W_{x\mathrm{i}}} L = \sum_t \mathrm{i}^{(t)}(1 - \mathrm{i}^{(t)}) x^{(t)} \nabla_{\mathrm{i}^{(t)}} L \tag{8.21b}$$

$$\nabla_{W_{xf}} L = \sum_t f^{(t)}(1 - f^{(t)}) x^{(t)} \nabla_{f^{(t)}} L \tag{8.21c}$$

$$\nabla_{W_{xc}} L = \sum_t (1 - (\widetilde{c}^{\,(t)})^2) x^{(t)} \nabla_{\widetilde{c}^{\,(t)}} L \tag{8.21d}$$

$$\nabla_{W_{ho}} L = \sum_t o^{(t)}(1 - o^{(t)}) h^{(t-1)} \nabla_{o^{(t)}} L \tag{8.22a}$$

$$\nabla_{W_{h\mathrm{i}}} L = \sum_t \mathrm{i}^{(t)}(1 - \mathrm{i}^{(t)}) h^{(t-1)} \nabla_{\mathrm{i}^{(t)}} L \tag{8.22b}$$

$$\nabla_{W_{hf}} L = \sum_t f^{(t)}(1 - f^{(t)}) h^{(t-1)} \nabla_{f^{(t)}} L \tag{8.22c}$$

$$\nabla_{W_{hc}} L = \sum_t (1 - (\widetilde{c}^{\,(t)})^2) h^{(t-1)} \nabla_{\widetilde{c}^{\,(t)}} L \tag{8.22d}$$

$$\nabla_{b_o} L = \sum_t o^{(t)}(1 - o^{(t)})\nabla_{o^{(t)}} L \tag{8.23a}$$

$$\nabla_{b_{\mathrm{i}}} L = \sum_t \mathrm{i}^{(t)}(1 - \mathrm{i}^{(t)})\nabla_{\mathrm{i}^{(t)}} L \tag{8.23b}$$

$$\nabla_{b_f} L = \sum_t f^{(t)}(1 - f^{(t)})\nabla_{f^{(t)}} L \tag{8.23c}$$

$$\nabla_{b_c} L = \sum_t (1 - (\widetilde{c}^{\,(t)})^2)\nabla_{\widetilde{c}^{\,(t)}} L \tag{8.23d}$$

Finalmente, para los estados ocultos se obtiene la ecuación 8.24 la cual como se observa es diferente al modelo anterior.

$$\begin{aligned}\nabla_{h^{(t-1)}} L = {} & o^{(t)}(1 - o^{(t)})W_{ho}\nabla_{o^{(t)}} L + \mathrm{i}^{(t)}(1 - \mathrm{i}^{(t)})W_{h\mathrm{i}}\nabla_{\mathrm{i}^{(t)}} L \\ & + f^{(t)}(1 - f^{(t)})W_{hf}\nabla_{f^{(t)}} L + (1 - (\widetilde{c}^{\,(t)})^2)W_{hc}\nabla_{\widetilde{c}^{\,(t)}} L\end{aligned} \tag{8.24a}$$

$$\nabla_{h^{(t-1)}} L = \nabla_{h^{(k-1)}} L + W_{hy}\nabla_{z^{(t-1)}} L \tag{8.24b}$$

Para la generación del estado oculto inicial h_o, los gradientes son los representados en las ecuaciones 8.25 que se derivan de igual forma que en modelo RNN.

$$\begin{aligned}\nabla_{W_{\mathrm{i}h}} L &= (1 - (h^{(0)})^2)I(\nabla_{h^{(0)}} L) \\ \nabla_{W_{sh}} L &= (1 - (h^{(0)})^2)SA(\nabla_{h^{(0)}} L) \\ \nabla_{W_{\mathrm{e}h}} L &= (1 - (h^{(0)})^2)EEG(\nabla_{h^{(0)}} L) \\ \nabla_{\mathrm{d}} L &= (1 - (h^{(0)})^2)\nabla_{h^{(0)}} L \\ \nabla_{\mathrm{e}} L &= (1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L)W_{sh}^{\top}(1 - (SA)^2) \\ \nabla_f L &= (1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L)W_{\mathrm{e}h}^{\top}(1 - (EEG)^2) \\ \nabla_S L &= (1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L)W_{sh}^{\top}(1 - (SA)^2)s \\ \nabla_{W\mathrm{eeg}} L &= (1 - (h^{(0)})^2)(\nabla_{h^{(0)}} L)W_{\mathrm{e}h}^{\top}(1 - (EEG)^2)\mathrm{eeg}\end{aligned} \tag{8.25a}$$

(8.25b)

8.5. Definición Backpropagation RNN-MDN

En esta sección se muestran los cálculos de las derivadas de la función de pérdida en función de los nuevos outputs, es decir, $\hat{y}_t = (\hat{\mathrm{e}}_t, \{\hat{\pi}_t, \hat{\mu}_t, \hat{\sigma}_t, \hat{\rho}_t\}_{j=1}^{M})$, según lo expuesto en [41]. El resto de los cálculos es omitido, ya que esta implementación no se lleva a cabo.

La función de pérdida esta dada por la ecuación 8.26a y al reemplazar los valores de $P(x_{t+1}|y_t)$ se obtiene la ecuación 8.26b.

$$\mathcal{L}(x) = -\sum_{t=1}^{T} \log Pr(x_{t+1}|y_t) \tag{8.26a}$$

$$\mathcal{L}(x) = \sum_{t=1}^{T} -\log\left(\sum_{j=1}^{M} \pi_t^j \mathcal{N}(x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j)\right) - \begin{cases} \log \mathrm{e}_t & \text{si } (x_{t+1})_3 = 1 \\ \log 1 - \mathrm{e}_t & \text{si no} \end{cases} \tag{8.26b}$$

Se calcula la derivada de la función de pérdida respecto al indicador del final de fijaciones.

$$\frac{\partial \mathcal{L}(x)}{\partial \hat{\mathrm{e}}_t} = (x_{t+1})_3 - \mathrm{e}_t \tag{8.27}$$

Se definen las componentes responsabilidades γ_t^j

$$\hat{\gamma}_t^j = \pi_t^j \mathcal{N}(x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j) \tag{8.28a}$$

$$\gamma_t^j = \frac{\hat{\gamma}_t^j}{\sum_{j'=1}^{M} \hat{\gamma}_t^{j'}} \tag{8.28b}$$

Con lo que se obtiene:

$$\frac{\partial \mathcal{L}(x)}{\partial \hat{\pi_t^j}} = \pi_t^j - \gamma_t^j \tag{8.29}$$

$$\frac{\partial \mathcal{L}(x)}{\partial (\hat{\mu}_t^j, \hat{\sigma}_t^j, \hat{\rho}_t^j)} = -\gamma_t^j \frac{\partial \log \mathcal{N}(x_{t+1}|\mu_t^j, \sigma_t^j, \rho_t^j)}{\partial (\hat{\mu}_t^j, \hat{\sigma}_t^j, \hat{\rho}_t^j)} \tag{8.30}$$

Donde:

$$\frac{\partial \log \mathcal{N}(x|\mu, \sigma, \rho)}{\partial \hat{\mu}_1} = \frac{C}{\sigma_1}\left(\frac{x_1 - \mu_1}{\sigma_1} - \frac{\rho(x_2 - \mu_2)}{\sigma_2}\right) \tag{8.31a}$$

$$\frac{\partial \log \mathcal{N}(x|\mu, \sigma, \rho)}{\partial \hat{\mu}_2} = \frac{C}{\sigma_2}\left(\frac{x_2 - \mu_2}{\sigma_2} - \frac{\rho(x_1 - \mu_1)}{\sigma_1}\right) \tag{8.31b}$$

$$\frac{\partial \log \mathcal{N}(x|\mu, \sigma, \rho)}{\partial \hat{\sigma}_1} = \frac{C(x_1 - \mu_1)}{\sigma_1}\left(\frac{x_1 - \mu_1}{\sigma_1} - \frac{\rho(x_2 - \mu_2)}{\sigma_2}\right) - 1 \tag{8.31c}$$

$$\frac{\partial \log \mathcal{N}(x|\mu, \sigma, \rho)}{\partial \hat{\sigma}_2} = \frac{C(x_2 - \mu_2)}{\sigma_2}\left(\frac{x_2 - \mu_2}{\sigma_2} - \frac{\rho(x_1 - \mu_1)}{\sigma_1}\right) - 1 \tag{8.31d}$$

$$\frac{\partial \log \mathcal{N}(x|\mu, \sigma, \rho)}{\partial \hat{\rho}} = \frac{(x_1 - \mu_1)(x_2 - \mu_2)}{\sigma_1 \sigma_2} + \rho(1 - CZ) \tag{8.31e}$$

Con Z dado por la ecuación 6.8b y C dado por:

$$C = \frac{1}{1-\rho^2} \tag{8.32}$$

Con todas las derivadas entregadas, es posible calcular por regla de la cadena la derivada de la función de pérdida respecto a los parámetros de la red, pudiendo entrenarla a través del método back-propagation.