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

PROYECCIÓN DE DATOS MULTIDIMENSIONALES UTILIZANDO TEORÍA DE LA INFORMACIÓN

TESIS PARA OPTAR AL GRADO DE MAGISTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

PABLO ANDRÉS VERA CADENAS

PROFESOR GUÍA:
PABLO ESTÉVEZ VALENCIA

MIEMBROS DE LA COMISIÓN:
CLAUDIO PÉREZ FLORES
JORGE SILVA SANCHEZ
PABLO ZEGERS FERNÁNDEZ

SANTIAGO DE CHILE
DICIEMBRE 2010

RESUMEN DE LA TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA INGENIERÍA,
MENCIÓN ELÉCTRICA Y AL TÍTULO DE INGENIERO
CIVIL ELECTRICISTA POR:
PABLO ANDRÉS VERA CADENAS
FECHA: 06/01/2011
PROF. GUÍA: DR. PABLO ESTÉVEZ V.

Proyección de Datos Multidimensionales Utilizando Teoría de la Información

En el presente trabajo se desarrolló un método no supervisado de proyección y visualización de datos multidimensionales a espacios de baja dimensión, en especial a 2D. El modelo de proyección propuesto consiste en una matriz de transformación lineal y ortonormal entre los espacios de entrada y salida. Para la optimización de los parámetros se utilizaron como criterios de proyección medidas basadas en la Teoría de la Información, en particular la Información Mutua. Debido a la complejidad del cálculo de la Información Mutua utilizando la forma clásica de Shannon, se trabajó con medidas basadas en la entropía de Renyi, las que combinadas con un estimador de funciones de densidad de probabilidad, llamado ventana de Parzen, permitieron el cálculo de la Información Mutua Cuadrática directamente a partir de los ejemplos. El método es no paramétrico ya que no requiere información a priori sobre la distribución de los datos. Adicionalmente, para mejorar el desempeño se añadió un pre-procesamiento para los datos llamado Blanqueo, el cual transforma los datos linealmente de forma que las características de los mismos no tengan correlación y que la varianza sea unitaria.

El método fue probado en cuatro bases de datos distintas con diversa complejidad y fue comparado con otros algoritmos como Análisis de Componentes Principales *(PCA), Stochastic Neighbor Embedding (SNE)* y Mapas de Sammon *(NLM),* utilizando como criterios de desempeño tanto medidas de preservación topológica como otras basadas en *clustering*. Los resultados mostraron que el método propuesto es capaz de proyectar datos de alta a baja dimensión manteniendo gran parte de la información de los mismos, en especial en términos de *clustering*. El algoritmo superó a PCA en todas las pruebas y obtuvo resultados comparables con SNE y NLM a pesar de que estos métodos son no-lineales. Se desarrolló además una caracterización del método para determinar aspectos como orden computacional y dependencia de parámetros. Por otro lado, se demostró la necesidad de desarrollar nuevas métricas para medir el desempeño de los algoritmos de proyección.

A mi familia y amigos,
gracias por el apoyo y la confianza...

Agradecimientos

Al llegar a este punto y mirar atras, son muchas las personas a las que uno quisiera agradecer. A todos aquellos que de una u otra forma hicieron posible que terminara este ciclo de mi vida les agradezco en el alma. Obviamente los pilares fundamentales fueron mis padres, Pedro y Magdalena, quienes siempre me apoyaron respondiendo mis inquietudes y me dieron las herramientas para desarrollarme académica y personalmente. Gracias por su confianza y por dejarme ser lo que quise ser. El agradecimiento se extiende también a mi hermano Ezequiel, y hermanas, Angela, Carolina, Francisca y Antonia, la Lola, mis abuelos que ya no están, Tata, Lela y Lelo, y que de seguro estarían muy felices, a mi abuela Maria y la tia Claudia, quien nos dejó de forma muy temprana. A todos ellos les debo parte de mi vida. Al resto de la familia, tios y primos. A Javiera, a quien quiero mucho y que me ha acompañado este último año, y que espero me acompañe por muchos más.

A mis amigos de toda la vida, en especial a Iván, Sergio y Cristian, con quienes corriamos por el pasaje, o pateabamos una pelota mientras soñabamos con lo que haríamos o seríamos cuando grandes. Juntos crecimos con muchos sueños y de a poco se han ido haciendo realidad. A los amigos del Instituto Nacional, con los que compartimos seis valiosos años. A los amigos de la Universidad, a los que conocí desde primer año, a Manuel B., Samir, Marcelo, Flaco, Punky, Oso, Javier, Carolina, gracias por las experiencias vividas y la enseñanza de que la universidad no solo es un lugar para aprender sobre física y matemáticas, si no que un lugar para aprender de la vida. A los compañeros del Laboratorio, desde aquel lejano 2005, Rodrigo Flores, Neven, Meme, Rodrigo H., Leito Medina, Jorge, Daniel B., Daniel S., Rafa, Carlos, Leo y Javier Causa, Alonso, Alejandro, Lucho, Cament, Juan, Chubo, memoristas, tesistas y esclavos varios, por hacer del trabajo un momento grato, de todos aprendí y sigo aprendiendo muchas cosas en todo sentido (la mayor parte en el sentido nerd, debo decirlo). A Felipe y Christian, con los que compartimos muchas cosas durante este año.

Un agradecimiento especial al Prof. Pablo Estévez, a quien le debo la finalización de esta Tesis y quien me dió la oportunidad de vivir algunas de las experiencias más importantes de mi vida, como fueron los viajes a Japón y EE.UU. A los profesores de la comisión por su tiempo, al Dr. Saito, al Dr. Principe y a todos los que me ayudaron en mi formación profesional.

Esta tesis fue realizada gracias al financiamiento del proyecto FONDECYT 1080643.

Índice general

Capítulo 1

Introducción

En la actualidad, empresas, instituciones gubernamentales, universidades, e incluso personas particulares, manejan grandes cantidades de datos con diversa información de utilidad para ellos. El tamaño de estas bases de datos hace infactible que un humano, aunque sea experto, pueda hacer análisis profundos de éstos por simple observación o con cálculos manuales. Por esta razón se necesitan métodos computacionales que sean capaces de navegar en el inmenso conjunto de datos y extraer la información relevante. Estos métodos pueden ser estadísticos o heurísticos, entre otros. Su complejidad dependerá del tipo de información que se quiera extraer.

Uno de los tópicos importantes dentro de estas tareas es la reducción de dimensionalidad de los datos, es decir, extraer estructuras pertenecientes a subespacios de baja dimensión escondidas dentro de espacios de mayor dimensión. Los algoritmos de reducción de dimensionalidad, tienen generalmente dos objetivos: compresión y/o proyección de datos. Algunas aplicaciones son por ejemplo minería de datos, análisis de datos multivariables y reconocimiento de patrones. La compresión de datos es conveniente cuando no se tiene el suficiente espacio para almacenar cierta cantidad de datos, cuando la capacidad de un canal de comunicación no es suficiente para transmitir una señal completa o cuando se requiere entrenar algún clasificador pero la alta dimensión de los datos de entrada hacen que esto sea muy difícil. Los métodos de proyección y/o visualización en tanto, permiten extraer la estructura de los datos y proyectar la misma en un mapa en dos o tres dimensiones preservando la topología. Estos mapas permiten al usuario usar su sistema visual para encontrar información relevante en los datos.

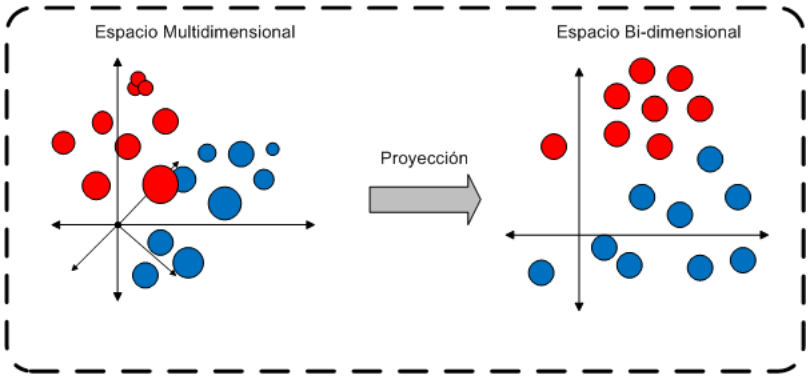


Figura 1.1: Ejemplo Proyección a 2 dimensiones

En la literatura actual, existen variados algoritmos de reducción de dimensionalidad que siguen distintos enfoques. Algunos algoritmos intentan preservar distancias, no necesariamente euclidianas, entre pares de datos [1–5], otros se enfocan en preservar la topología de los datos manteniendo relaciones de vecindad, en vez de distancias [6–8]. Además, los algoritmos se pueden dividir dependiendo de si la preservación (de distancias o topología) es global o local [6, 9]. Los algoritmos de preservación global intentan proyectar los datos de modo que muestras que sean lejanas en el espacio de entrada se proyecten lejos en el espacio de salida, a la vez que los datos que se encuentren cerca, se proyecten también cerca. Por otro lado, los algoritmos de preservación local tienen por proposito que los datos que sean cercanos en la entrada sean proyectados cerca en la salida sin ocuparse de los datos más lejanos. Un tercer criterio para clasificar los algoritmos de reducción de dimensionalidad es si estos son capaces de encontrar en el espacio de alta dimensión estructuras lineales o no-lineales. Los algoritmos de proyección lineal se pueden representar mediante una matriz de transformación lo que permite incluso hacer el mapeo inverso, sin embargo, estos algoritmos no son capaces de encontrar estructuras no-lineales en los datos. A su vez los algoritmos no-lineales son capaces de encontrar ciertas no linealidades pero a veces sus salidas son difíciles de interpretar y suelen ser mucho más complejos de analizar que los algoritmos lineales.

Los métodos de proyección lineal más conocidos son Análisis de Componentes Principales (*Principal Components Analysis,* PCA) [10] y Escalamiento Multidimensional (*Multidimensional Scaling*, MDS) [1]. PCA determina la proyección a un espacio de baja dimensión mediante una matriz de transformación formada por los vectores propios correspondientes a los valores propios más grandes de la matriz de covarianza de los datos originales. Esto implica que PCA proyecta los datos en las

direcciones de mayor varianza. Similarmente, MDS en su forma más clásica, es una técnica estadística que utiliza como entrada una matriz de disimilitud entre los datos, generalmente la distancia entre puntos. Mediante la minimización de cierta función de costos se busca generar las coordenadas de salida del sistema, buscando la configuración que mejor represente las distancias (disimilitudes) dadas.

Dentro de los métodos no lineales, uno de los más conocidos es el de Sammon ó *Non Linear Mapping* (NLM) [2], que consiste en un proceso iterativo que parte con una configuración al azar de los puntos en el espacio de baja dimensión y los va ajustando mediante la minimización de una medida de preservación global de distancias euclidianas entre todos los pares de puntos, conocida como el estrés de Sammon. Otro ejemplo de algoritmo no lineal es ISOMAP [5], el cual extiende MDS para utilizar distancias de grafo en vez de distancias euclidianas, siguiendo también un enfoque de preservación global. *Locally Linear Embedding* (LLE) [6,7], es un método local que intenta proyectar los datos que se encuentran cercanos en el espacio de entrada en puntos cercanos en el espacio de salida mediante proyecciones lineales locales optimizando algebraicamente una medida de error de reconstrucción.

Un enfoque distinto para la proyección de datos, lo otorgan las redes neuronales artificiales (RNA) auto-organizativas. A las RNA auto-organizativas, no se les entrega información externa sobre clases ni salida deseada, sólo se le entregan los datos de entrada. Esto hace que las neuronas aprendan competitivamente generando a la salida una representación de los datos de entrada. El algoritmo de este tipo más conocido y estudiado es el mapa auto organizativo de Kohonen, SOM (*Self Organizing Maps*), [8]. SOM cuantiza el espacio de alta dimensión, asociando a cada dato un vector prototipo o representante. Cada prototipo es asociado a una neurona en una grilla fija en el espacio de salida. La grilla de salida impone una relacion topológica entre los datos, pero al ser fija no da información sobre la distancia entre los mismos.

En el ámbito probabilístico, el algoritmo *Stochastic Neighbor Embedding* (SNE) [11], genera distribuciones en los espacios de entrada y salida, basadas en la probabilidad de que los datos sean vecinos. La diferencia entre distribuciones es minimizada mediante la divergencia de Kullback-Leibler (KL) [12]. El método *Parametric Embedding* (PE) [13] intenta generalizar SNE incorporando las etiquetas de clases en el espacio de baja dimensión.

En [14,15] Estévez, Figueroa y Saito utilizan una medida de entropía cruzada entre probabilida-

des de entrada y salida permitiendo proyectar tanto datos como vectores prototipos. *Neural Gas - Cross Entropy* (NGCE) obtiene los vectores prototipo cuantizando con *Neural Gas* [16]. La función de costos basada en la entropía cruzada se minimiza utilizando el método de Newton-Raphson.

En un ámbito más amplio, José Principe y otros [17] han hecho contribuciones al campo del aprendizaje utilizando Teoría de la Información (*Information Theoretic Learning*, ITL). El objetivo de ITL es extraer información directamente de los datos, lo que se conoce también como aprender a través de ejemplos, usando especialmente la función de densidad de probabilidad (fdp) de los mismos. Si un conjunto de datos contiene información sobre un evento del mundo real, el objetivo de ITL es capturar esa información en los parámetros de una máquina de aprendizaje (ej. red neuronal). Para entrenar estas máquinas de aprendizaje se utilizan diferentes criterios como la entropía e información mutua (IM) [18]. La IM ha sido usada ampliamente como criterio para extracción y/o selección de características, pero tiene el problema de acarrear una pesada carga computacional a menos que se asuma a priori cierta distribución de los datos. Cuando las variables son escalares, la IM puede ser estimada utilizando histogramas, pero esto no es posible cuando la dimensionalidad de los datos aumenta, debido a la baja densidad de los mismos. Otros estimadores basados en kernels pueden ser utilizados, con la ventaja de que estos estimadores pueden ser diferenciables, lo que permite encontrar reglas de optimización simples como las basadas en el gradiente. Con el objeto de facilitar el cálculo de la entropía, Principe propuso utilizar la entropía de Renyi [19] en vez de la entropía de Shannon [18], ya que al combinar la entropía de Renyi con el estimador de fdp llamado Ventana de Parzen [20], fue posible estimar tanto la entropía como la IM directamente desde los datos, sin la necesidad de hacer un cálculo explícito de las fdps.

Una de las ventajas de este método es que no es paramétrico, es decir, no es necesario tener información a priori sobre la distribución de los datos en el espacio de entrada, ya que la información se extrae de los mismos. Por otro lado este esquema puede ser utilizado tanto por sistemas de aprendizaje supervisado como no supervisado, lineales o no lineales, lo que permite utilizar el mismo criterio para resolver una gran gama de problemas donde el objetivo es entrenar una máquina de aprendizaje directamente desde los datos. Estas características son las que convierten a ITL en un marco teórico bastante completo para ser utilizado como plataforma base para resolver el problema de reducción de dimensionalidad.

1.1. Objetivo General

El objetivo de esta Tesis es desarrollar un nuevo método de proyección de datos multidimensionales utilizando Teoría de la Información. El método intenta preservar la mayor cantidad de información posible contenida en los datos de entrada al proyectar al espacio de salida, utilizando como criterio alguna medida basada en ITL.

1.2. Objetivos Específicos

Se plantean los siguientes objetivos específicos:

1. Proponer y caracterizar el método de proyección basado en Teoría de la Información, analizando ventajas y desventajas

2. Comparar resultados con métodos alternativos como PCA, NLM y SNE, tanto en bases de datos *benchmark* como en bases de aplicaciones reales, tomando en consideración medidas de desempeño existentes en la literatura.

El texto a continuación se organiza de la siguiente forma:

En el capítulo 2 se describen los algoritmos de visualización de datos más conocidos en la literatura como PCA, MDS, NLM, SOM, ISOMAP, LLE y SNE, además, las medidas de desempeño que se utilizan para evaluar los algoritmos. Se explican los fundamentos de la Teoría de Información, máquinas de aprendizaje y los principios de optimización basados en ITL.

El capítulo 3 describe el algoritmo de visualización propuesto basado en ITL. En el capítulo 4 se presentan las simulaciones realizadas para probar el algoritmo, las bases de datos utilizadas y los resultados de aplicar el algoritmo sobre las mismas. En el capítulo 5 se discuten los resultados mostrados en el capítulo 4, analizando las ventajas y desventajas del algoritmo propuesto con respecto a los algoritmos existentes. En capítulo 6 se presentan las conclusiones sobre el trabajo realizado, destacando los principales aportes y proponiendo investigaciones futuras. Finalmente se presenta la bibliografía utilizada para desarrollar esta Tesis.

Capítulo 2

Antecedentes

2.1. Visualización de Datos Multidimensionales

Los algoritmos de reducción de dimensionalidad crean representaciones en baja dimensión de datos de alta dimensión. Formalmente, se tiene la transformación de un conjunto de N vectores $x_i \in \Re^D$, en otro conjunto de N vectores $y_i \in \Re^d$ con $d < D$.

El objetivo de reducir la dimensionalidad de los datos puede ser variado dependiendo de la aplicación que se le quiera dar. Con SOM [8] por ejemplo se puede tener una cuantización de los datos además de su proyección, lo que permitiría comprimir una gran base de datos en una cantidad inferior de vectores representantes. Algoritmos como *Linear Discriminant Analysis* (LDA) [21, 22] utilizan la información de la clase a la que pertenece cada dato para proyectar de forma de maximizar la separación entre las distintas clases, lo que puede ayudar a diseñar o entrenar un clasificador de una forma mucho más simple. Algunos algoritmos intentan buscar estructuras de baja dimensión escondidas en los datos de alta dimensión, las cuales pueden estar embebidas lineal o no linealmente [5–7]. Este aprendizaje es conocido como *manifold learning,* donde *manifold* es un objeto geométrico en cualquier dimensión, tal como un subespacio contenido en un espacio de mayor dimensión. Por otro lado, un método de visualización de datos impone la restricción que el espacio de salida debe ser de dos o tres dimensiones. Además tiene que cumplir con mantener cierta estructura de los datos, para que la proyección tenga sentido y sea de utilidad ante los ojos del observador. Sin embargo, a pesar de estas diferencias, muchas veces se puede utilizar un mismo

método para distintas tareas.

Los métodos de visualización se pueden separar generalmente en dos grandes grupos, métodos globales y métodos locales. Los primeros tienden a preservar la geometría de los datos a todas las escalas proyectando puntos cercanos en el espacio de entrada a coordenadas cercanas en el espacio de salida, así como los puntos distantes a coordenadas distantes. Algunos de los métodos globales más conocidos son PCA, MDS, NLM e ISOMAP. Por otro lado, los métodos locales buscan preservar las geometrías locales de los datos, por lo que se preocupan de que los puntos cercanos, o vecinos, sean proyectados a coordenadas o vecindades cercanas en el espacio de salida. Algunos métodos que favorecen las proyecciones locales son SOM, ISOTOP, *Laplacian Eigenmaps* [23] y LLE. Esto implica que existe un *trade off* entre generar una proyección de preservación local y una global.

En la siguiente sección se revisan las características de los principales métodos de visualización existentes en la literatura.

2.1.1. Métodos de visualización

Uno de los algoritmos más estudiados y utilizados para la reducción de dimensionalidad es PCA [10]. Para definir PCA existen dos opciones que llevan al mismo resultado [24]. Pearson define PCA como la proyección lineal que minimiza el error cuadrático medio o distancia cuadrática media entre los datos y sus proyecciones. A este error se le conoce también como el costo de proyección promedio. Por otro lado PCA puede ser definido como la proyección ortogonal de los datos en un espacio de menor dimensión llamado subespacio principal, tal que la varianza de los datos proyectados es maximizada [25].

La definición basada en la máxima varianza permite encontrar la proyección con el cálculo de los vectores propios de la matriz de covarianza de los datos de entrada, generando con estos vectores una proyección lineal y ortogonal de los mismos, utilizando los d vectores propios con valores propios más grandes. Esto es equivalente a decir que PCA hace un cambio de base algebraica, donde la nueva base encontrada es una combinación lineal de las bases originales. Esto permite expresar a los datos de una mejor manera, ya que estas componentes son las que maximizan la relación señal a ruido, o sea entregan la información más relevante existente en los datos. La transformación puede ser descrita como, $\mathbf{Y} = \mathbf{PX}$, donde $\mathbf{P}$ es una matriz formada por los vectores propios de la matriz de covarianzas de $\mathbf{X}$. La ventaja de PCA sobre otros métodos es que no es necesario fijar ningún

parámetro, ni tampoco realizar un proceso de optimización iterativo que puede llevar a mínimos locales. Sin embargo, al utilizar solo la información de la varianza, las coordenadas encontradas son linealmente independientes solamente cuando los datos pertenecen a una distribución gaussiana. Para distribuciones no-gaussianas o multimodales, PCA simplemente descorrelaciona los ejes. La formulación basada en la minimización del error de proyección lleva a los mismos resultados. La figura 2.1 muestra un ejemplo de ejes encontrados por PCA en una base de datos tridimensional. En este ejemplo los ejes encontrados son los que definen la proyección en dos dimensiones.

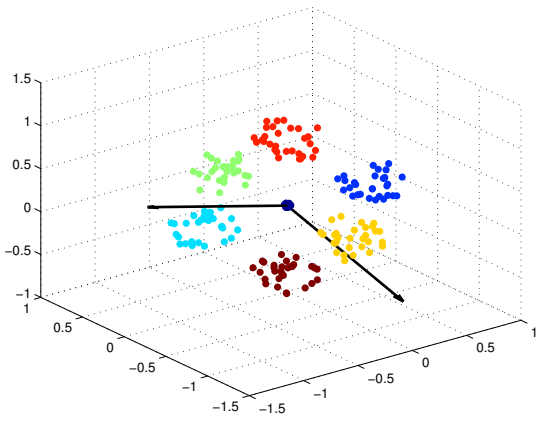


Figura 2.1: Ejemplo ejes encontrados por PCA

MDS clásico [1], es otro tipo de proyección lineal, que en vez de utilizar una matriz de correlaciones utiliza una matriz de disimilitud de los datos, D. Usualmente, pero no necesariamente, esta matriz está compuesta por las distancias euclidianas entre los datos en el espacio de entrada, aunque existen variaciones que utilizan por ejemplo distancia de grafo o geodésica [5]. La proyección generada por MDS minimiza la diferencia $\|B - XX^T\|$, donde B es la matriz D^2 centrada en sus filas y columnas. Para la proyección, se utiliza una *Singular Value Decomposition* (SVD) de B, por lo que al igual que en PCA, se necesita resolver un problema de cálculo de vectores propios. MDS y PCA poseen los mismos óptimos cuando la matriz D está basada en la distancia euclidiana, siendo la principal ventaja de MDS que puede ser utilizado cuando no se tiene la información vectorial de los datos, ya que basta tener una medida de disimilitud entre los mismos.

ISOMAP [5] extiende MDS al incluir como medida de disimilitud la distancia geodésica o de grafo de los datos en el espacio de alta dimensión. La proyección se obtiene de la misma forma que

MDS. El problema de ISOMAP es el costo computacional de calcular estas distancias y el posterior cálculo de los vectores propios del algoritmo MDS. Generalmente para el cálculo de las distancias geodésicas se utiliza el algoritmo de Dijkstra [26], el cual determina el camino más corto entre un nodo o vértice y el resto de los nodos en un grafo, incluso en el caso de que existan pesos en cada arista, por lo que un paso previo necesario es la construcción de dicho grafo. La ventaja de utilizar la distancia de grafo es que ésta permite a ISOMAP descubrir *manifolds* que no se pueden encontrar utilizando medidas de distancia euclidiana en forma global, como por ejemplo, espacios curvos.

LLE [6, 7], intenta encontrar encajes no lineales buscando estructuras lineales en forma local, basándose en la idea de que cada dato y sus vecinos pertenecerán a una aproximación lineal del *manifold* en forma local. Tal como los métodos anteriores, requiere resolver un problema vectorial y no necesita una optimización iterativa, por lo que no sufre de problemas de mínimos locales. La geometría local se caracteriza por pesos asignados a los vecinos de cada dato los cuales reconstruyen el dato en forma lineal minimizando el error de reconstrucción definido como

$$E(W) = \sum_i \left\| x_i - \sum_j W_{ij} x_j \right\|^2 . \tag{2.1}$$

Para encontrar los pesos óptimos se añade la restricción de que $W_{ij} = 0$ si i y j no son vecinos y además $\sum_j W_{ij} = 1$. Generalmente se utiliza como único parámetro el número de vecinos a partir del cual se reconstruye cada dato, sin embargo también es posible asignar un radio para escoger los vecinos de acuerdo a los que caigan dentro de la híper esfera definida por dicho radio. Una vez optimizados los pesos se debe calcular la proyección, para lo que se minimiza la siguiente medida de error,

$$E(Y) = \sum_i \left\| y_i - \sum_j W_{ij} y_j \right\|^2 . \tag{2.2}$$

La figura 2.2 muestra un ejemplo de proyección utilizando LLE, donde un *manifold* bidimensional se encuentra embebido en un espacio tridimensional. LLE es capaz de encontrar el manifold curvo y desenrollarlo. A pesar de no tener problemas con mínimos locales, puede tener problemas de tiempo de cómputo y memoria con grandes cantidades de datos.

Non-Linear Mapping (NLM) [2], publicado por J. Sammon el año 1969, busca crear un mapeo no-lineal mediante la preservación de las distancias entre los datos en el espacio de entrada, en

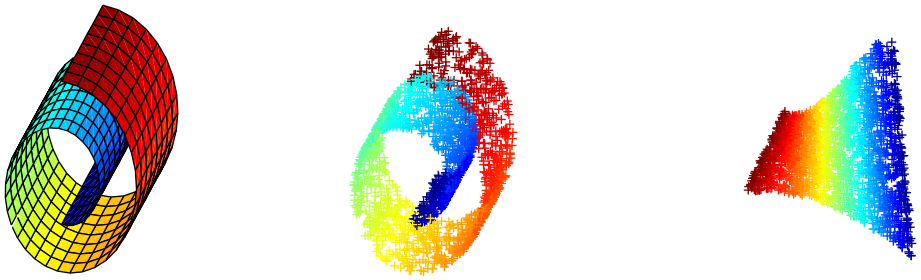


Figura 2.2: Ejemplo de proyección con LLE del conjunto *Swiss Roll,* que es un *manifold* de dos dimensiones embebido en un espacio tridimensional. El método es capaz de encontrar este *manifold* y desenrrollar la estructura

la proyección en el espacio de salida. Para conseguir esto Sammon propuso minimizar el siguiente funcional, o medida de error, conocido como el *estrés de Sammon*,

$$E = \frac{1}{\sum\sum_{i<j} d^*_{ij}} \sum\sum_{i<j} \frac{(d^*_{ij} - d_{ij})^2}{d^*_{ij}} \tag{2.3}$$

donde d^*_{ij} son las distancias entre los puntos i-ésimo y j-ésimo en el espacio de entrada y d_{ij} las distancias en el espacio de salida. Para minimizar (2.3) basta con un método de descenso por gradiente. Una de las desventajas de NLM es que no define una función de mapeo por lo que no se puede generalizar. Cada vez que se quiere agregar un nuevo punto es necesario correr el algoritmo nuevamente, además la optimización puede quedar estancada en mínimos locales. La carga computacional del mapa de Sammon es de $O\left(N^2\right)$.

El Mapa Auto Organizativo de Kohonen (SOM: *Self-Organizing Map*), desarrollado por el finlandés Teuvo Kohonen [8], es uno de los modelos de red neuronal no supervisada más populares, utilizado tanto como cuantizador vectorial como algoritmo de visualización. A diferencia del MLP (*Multilayer Perceptron*) o perceptrón multicapa, una red neuronal auto organizativa no requiere que los valores de salida deseada para cada entrada sean conocidos. El SOM corresponde a un mapeo de un espacio de dimensión D a un espacio de salida de menor dimensión d. Este mapa topográfico está basado en la forma en que las neuronas se relacionan espacialmente para responder a ciertos estímulos. El espacio de salida de la red de Kohonen consiste en una grilla d dimensional en la cual se distribuyen las neuronas. Por lo general la forma de la grilla es híper rectangular y

las neuronas se distribuyen equiespaciadamente. Esta grilla se usa para definir las relaciones de vecindad entre las neuronas. A cada neurona se le asocia un peso o vector prototipo w_i que posee la misma dimensión que los datos de entrada D. El mapa intenta preservar la estructura de la vecindad de los datos de entrada a través de las neuronas en el espacio de salida, no asi las distancias. El número de neuronas permanece fijo desde el comienzo hasta el final del entrenamiento, aunque existen variantes constructivas como GSOM (*Growing SOM*) [27], que parten de 2 neuronas y van agregando de acuerdo a cierta regla de ajuste. El algoritmo busca adaptar el valor de los pesos w_i asociados a cada neurona mediante una regla de adaptación heurística. SOM no optimiza ningún funcional explicito.

La regla de actualización de cada vector prototipo es:

$$\triangle w_i\left(t\right) = w_i\left(t+1\right) - w_i\left(t\right) = \alpha\left(t\right) h_{ij*}\left(t\right)\left[x\left(t\right) - w_i\left(t\right)\right] \tag{2.4}$$

donde $x\left(t\right)$ es el vector de entrada en la iteración t, h_{ij*} es la función de vecindad centrada en la neurona ganadora j^* y α es la tasa de aprendizaje. Generalmente h_{ij*} se define como una función gaussiana:

$$h_{ij*}(t) = e^{-\frac{\left\|p_i - p_{j*}\right\|^2}{L(t)^2}} \tag{2.5}$$

donde $\|.\|$ es la medida de distancia, p_i y p_{j*} son las posiciones de los vectores posición de las neuronas i y j^*. $L(t)$ es el ancho de la vecindad el cual decae con t.

Uno de los problemas de la representación de SOM es que utiliza una grilla fija, luego las relaciones de distancia entre neuronas no pueden ser apreciadas, por lo que se requiere algún tipo de post procesamiento para visualizar esto. Una de las técnicas más utilizadas es la Matriz Unificada de Distancias o Matriz U [28]. Esta consiste en representar la distancia a los nodos adyacentes a través de una escala de colores, donde un color oscuro representará una gran distancia entre los vectores prototipos asociados, en cambio un color claro indica que los vectores son cercanos en el espacio de entrada. En la figura 2.3 se puede apreciar un ejemplo de una proyección SOM utilizando la Matriz U. La proyección muestra un mapa generado en base a una encuesta realizada sobre gustos personales a 21 personas del departamento de Ingeniería Eléctrica de la Universidad de Chile, las cuales respondieron 11 preguntas, generando por cada una de ellas una característica. Como se puede observar dentro de la grilla se forman *clusters* o agrupaciones de individuos que comparten

ciertas características.

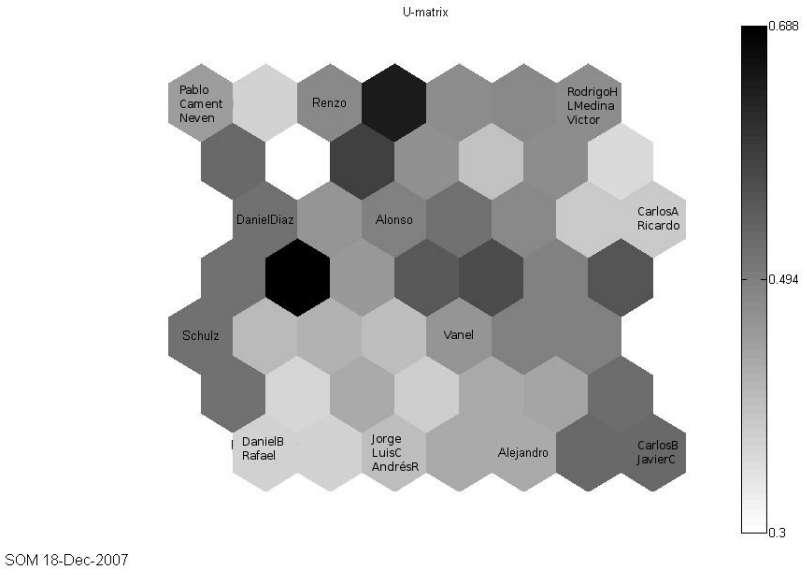


Figura 2.3: Ejemplo proyección SOM, utilizando matriz U. Los tonos oscuros indican mayor distancia a sus vecinos

Otras técnicas para visualizar la salida de SOM son Conexiones de *Cluster* [29] o matriz P [30]. TOPSOM [31] extiende SOM y reemplaza la grilla fija de salida por un espacio de salida continuo y dinámico.

PCA Probabilístico, proviene de la definición de PCA como la solución de máxima verosimilitud a un modelo probabilístico de variable latente [24]. En este modelo todas las distribuciones marginales y condicionales se definen como gaussianas. El modelo se puede formular a través de una variable latente explicita z que corresponde al subespacio definido por las componentes principales. Así se definen la distribución a priori sobre z, $p(z)$ y la distribución condicional $p(x|z)$, las cuales se restringen a ser gaussianas. La distribución $p(z)$ se define como una distribución de media cero y covarianza unitaria,

$$p(z) = \mathcal{N}(z|0, I), \tag{2.6}$$

mientras la distribución condicional se define como

$$p(x|z) = \mathcal{N}(x|Wz + \mu, \sigma^2 I), \tag{2.7}$$

donde se usa la condición de que la media es una función lineal de la variable z definida por la matriz de transformación W y el vector μ, y la varianza se define por el escalar σ^2. Para determinar los parámetros del modelo a través de una estimación de máxima verosimilitud se expresa la distribución marginal de la variable observada x como

$$p(x) = \int p(x|z)p(z)dz, \tag{2.8}$$

la cual también es gaussiana de la forma $p(x) = \mathcal{N}(x|\mu, C)$, donde la matriz de covarianzas C se define como

$$C = WW^T + \sigma^2 I. \tag{2.9}$$

Entre las ventajas que posee esta definición del modelo PCA está que al restringir el modelo a ser gaussiano el conjunto de parámetros a determinar puede ser limitado (i.e. no es necesario calcular todos los parámetros) y aun así se capturan las correlaciones dominantes en el conjunto de datos. Por otro lado se puede derivar un algoritmo del tipo EM *(Expectation Maximization)* para la optimización, lo que es mucho más eficiente que el algoritmo clásico ya que no se requiere hacer el cálculo completo de todos los vectores propios. Este modelo permite también tratar PCA desde el punto de vista Bayesiano, por lo que el subespacio principal se puede encontrar directamente desde los datos.

Stochastic Neighbor Embedding (SNE) [11], es un enfoque probabilístico para proyectar un conjunto de datos multidimensionales en un espacio de baja dimensión. Para esto se centra una función gaussiana en la posición de cada dato en el espacio de entrada, y se define una distribución sobre los potenciales vecinos. El objetivo es aproximar esta distribución con la que se genera al hacer la mima operación en el espacio de salida. En términos formales, para cada dato x_i, y cada vecino potencial x_j, se comienza computando la probabilidad asimétrica p_{ij}, que x_i pueda tener a x_j como vecino:

$$p_{ij} = \frac{\exp\left(-d_{ij}^2\right)}{\sum_{k \neq j} \exp\left(-d_{ik}^2\right)}. \tag{2.10}$$

La medida de disimilitud d_{ij} puede quedar como definición del problema, sin embargo por lo general se utiliza una medida de distancia euclidiana escalada definida como

$$d_{ij}^2 = \frac{\|x_i - x_j\|^2}{2\sigma_i^2}. \tag{2.11}$$

El valor de σ_i se puede definir a mano o a través de una busqueda binaria que hace la entropía de la distribución sobre los vecinos igual a $\log k$, donde k define el número efectivo de vecinos o "perplejidad". Este valor se define a mano por el usuario.

En forma análoga para el espacio de salida se define la probabilidad q_{ij}. Con el objetivo de minimizar la diferencia entre las distribuciones, se minimiza una función de costos basada en la suma de divergencias KL entre las distribuciones p_{ij} y q_{ij}. Definiendo la divergencia KL entre dos distribuciones P y Q como

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P_i \log \frac{P_i}{Q_i}, \tag{2.12}$$

la función de costos se define como

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} = \sum_i D_{KL}\left(P_i||Q_i\right). \tag{2.13}$$

El funcional (2.13) puede ser minimizado mediante el método del gradiente para ajustar las posiciones de los datos en el espacio de salida, según la siguiente regla,

$$\frac{\partial C}{\partial y_i} = 2\sum_j \left(y_i - y_j\right)\left(p_{ij} - q_{ij} + p_{ji} - q_{ji}\right). \tag{2.14}$$

Una característica de SNE, es que las observaciones directas de los datos de entrada no son requeridas, basta tener las relaciones de disimilitud entre los datos, al igual que en MDS.

Parametric Embedding (PE) [13] es una versión generalizada de SNE, donde además de proyectar los datos en sí, se proyecta la información de clase de los mismos si esta se encuentra disponible. El algoritmo recibe como entrada un conjunto de vectores con las probabilidades posteriores de las clases para cada dato. El espacio de salida se calcula minimizando una suma de divergencias KL bajo la premisa de que los datos son generados por una mixtura gaussiana con la misma covarianza en el espacio de salida. Una de las ventajas de PE sobre SNE (y otros métodos) es que la complejidad se escala por la multiplicación del número de datos por las clases y no es cuadrático con respecto al número de ejemplos.

Formalmente, dadas como entradas las probabilidades condicionales $p\left(c_k|x_i\right)$, PE busca las coordenadas de proyección y_i para los datos y ϕ_k para las clases, tal que $p\left(c_k|x_i\right)$ se aproxime lo más posible a la probabilidad a posteriori de una mezcla gaussiana de varianza unitaria en el

espacio de la proyección:

$$p\left(c_k|y_i\right) = \frac{p\left(c_k\right)\exp\left(-\frac{1}{2}\left\|y_i-\phi_k\right\|^2\right)}{\sum_{l=1}^{K} p\left(c_l\right)\exp\left(-\frac{1}{2}\left\|y_i-\phi_l\right\|^2\right)}. \tag{2.15}$$

Para medir la distancia entre las probabilidades se usa una suma de divergencias, una por cada dato

$$\sum_{i=1}^{N} KL\left(p\left(c_k|x_i\right)||p\left(c_k|y_i\right)\right). \tag{2.16}$$

La minimización de (2.16) con respecto a $p\left(c_k|y_i\right)$ es equivalente a minimizar el funcional

$$E\left(\{y_i\},\{\phi_k\}\right) = -\sum_{i=1}^{N}\sum_{k=1}^{K} p\left(c_k|x_i\right)\log p\left(c_k|y_i\right). \tag{2.17}$$

Como se tienen que encontrar tanto las coordenadas para los datos en el espacio de salida como para las clases, se utiliza un método de descenso por coordenada, minimizando iterativamente con respecto a y_i y ϕ_k,

$$\begin{aligned} \frac{\partial E}{\partial y_i} &= \sum_{k=1}^{K}\left(p\left(c_k|x_i\right)-p\left(c_k|y_i\right)\right)\left(y_i-\phi_k\right) \\ \frac{\partial E}{\partial \phi_k} &= \sum_{i=1}^{N}\left(p\left(c_k|x_i\right)-p\left(c_k|y_i\right)\right)\left(\phi_k-y_i\right). \end{aligned} \tag{2.18}$$

2.1.2. Medidas de calidad

En la actualidad no existe una forma estándar de medir la calidad de un mapa, muchas veces incluso la calidad del mapa queda a criterio de la calificación del observador, por lo mismo se han desarrollado tantas medidas como métodos dependiendo del enfoque. Dado que muchos métodos de proyección tanto lineales como no lineales se basan en la optimización de algún funcional, la medida de calidad más directa resulta ser la evaluación de tal funcional al término de la optimización. El problema salta a la vista y es que probablemente el método que optimice tal criterio tenga ventajas sobre otros métodos haciendo que la comparación sea injusta. Otro criterio que se ha manejado es la evaluación del error de reconstrucción. Si se tiene la función de mapeo $g(x)$ entonces el error de

reconstrucción se define como

$$Err = E\left\{\left(X - g^{-1}\left(g\left(X\right)\right)\right)^2\right\}. \tag{2.19}$$

Este error mide que tan bien se puede reconstruir el espacio de entrada utilizando la proyección. El problema del criterio (2.19) es que no siempre se puede tener la función inversa del mapeo. En algunos casos se pueden utilizar medidas como el error de clasificación de los datos utilizando la proyección. Pero esto es sólo aplicable a bases de datos donde se tienen las etiquetas de los mismos. Otra posibilidad es utilizar alguna medida como el estrés de Sammon (2.3), el cual mide que tan bien se preservan las distancias entre pares de puntos. Sin embargo estas medidas son dependientes de la distancia a utilizar y no necesariamente miden que tan bien se preserva la estructura de los datos. En este sentido se han desarrollado varias medidas que buscan establecer que tan bien se preserva la topología del espacio de entrada en el espacio de salida. Las primeras medidas estaban relacionadas a las proyecciones de SOM, el producto topográfico [32] y la función topográfica [33]. Posteriormente han aparecido medidas más generales que permiten una aplicación más amplia, las cuales miden como se preservan las vecindades sobre los K vecinos más cercanos a cada dato, siendo K variable. Para establecer estas vecindades se utilizan rankings basados en las distancias. En esta tesis se presentan cuatro diferentes medidas que utilizan este enfoque.

Medida Q_m

La medida de preservación topológica q_m [34] se basa en la comparación de los rankings de vecindad en los espacios de entrada y salida. Se calculan los n vectores más cercanos a cada vector j en el espacio de entrada $NN_{jiD}(i \in [1,n], j \in [1,N])$ y los n vectores más cercanos en el espacio de salida NN_{jid}. En base a estos rankings a cada dato se le asigna un puntaje de acuerdo a la siguiente regla, donde el parámetro k define una vecindad en torno a cada ejemplo:

$$q_{m_{ji}} = \begin{cases} 3, \text{ if } NN_{jiD} = NN_{jid} \\ 2, \text{ if } NN_{jiD} = NN_{jld}, l \in [1,n], i \neq l \\ 1, \text{ if } NN_{jiD} = NN_{jtd}, t \in [n,k], n < k \\ 0, \text{ otro caso} \end{cases} \tag{2.20}$$

Finalmente el q_m se calcula como:

$$q_m = \frac{1}{3nN}\sum_{j=1}^{N}\sum_{i=1}^{n} q_{m_{ji}} \tag{2.21}$$

El valor de q_m varía entre 0 y 1 dependiendo de la calidad de la preservación de vecindad, donde $q_m = 1$ implica una preservación perfecta.

Confiabilidad y Continuidad

Las medidas de confiabilidad (*trustworthiness*, M_T) y continuidad (*continuity,* M_C) [35, 36] también se basan en rankings de K-vecinos. M_T indica que tan confiable es la proyección midiendo si los datos proyectados cerca en el espacio de salida realmente son cercanos en el espacio de entrada. M_C en tanto indica que tan continuo es el mapa al medir si los datos cercanos en el espacio de entrada se mantienen cercanos en el espacio de salida. Sea N el número de datos y $r(i,j)$ el ranking del dato j ordenado de acuerdo a la distancia al dato i en el espacio de entrada. Sea $U_k(i)$ el conjunto de datos que se encuentran en la vecindad de tamaño k de i en el espacio de salida pero no en el de entrada. La medida de confiabilidad se define como:

$$M_T(k) = 1 - \frac{2}{Nk(2N-3k-1)}\sum_{i=1}^{N}\sum_{j\in U_k(i)}(r(i,j)-k). \tag{2.22}$$

Así mismo, sea $V_k(i)$ el conjunto de datos que pertenecen a la vecindad k-ésima en el espacio de entrada pero no en el de salida y sea $\hat{r}(i,j)$ el ranking del dato j ordenado con respecto a la distancia de i en el espacio de salida, la medida de continuidad se define como:

$$M_C(k) = 1 - \frac{2}{Nk(2N-3k-1)}\sum_{i=1}^{N}\sum_{j\in V_k(i)}(\hat{r}(i,j)-k) \tag{2.23}$$

Para hacer uso de estas medidas por lo general se utiliza un k variable con lo que se construyen curvas en función de k. Estas curvas pueden mostrar el énfasis local o global de la proyección.

Medida Q_{NX}

Sea la distancia entre los puntos x_i y x_j en el espacio de baja dimensión d_{ij} y la distancia en el espacio de alta dimensión δ_{ij}. El ranking de x_j con respecto a x_i en el espacio de alta dimensión se define como $p_{ij} = \{\, k : \delta_{ik} < \delta_{ij} \text{ ó } (\delta_{ik} = \delta_{ij} \text{ y } 1 \leq k < j \leq N)\}$, a su vez el ranking en el espacio de

baja dimensión se define como $r_{ij} = \{\, k : d_{ik} < d_{ij} \text{ ó } (d_{ik} = d_{ij} \text{ y } 1 \leq k < j \leq N)\}$. Los rankings reflexivos (p_{ii}, r_{ii}) se definen iguales a cero, y cada ranking además se define como único por lo que este varía entre 1 y $N-1$.

En [37], Lee y Verleysen introducen la llamada *matriz de co-ranking* como

$$Q = [q_{kl}]_{1 \leq k,l \leq N-1} \quad q_{kl} = \{(i,j) : p_{ij} = k \text{ y } r_{ij} = l\}\,. \tag{2.24}$$

A la diferencia $p_{ij} - r_{ij}$ se le llama *error de ranking*. Se definen también los conceptos de *intrusion* cuando existe un error de ranking positivo para algún par (i,j), es decir cuando el vector j-ésimo entra en la vecindad n_i^K sin estarlo en la vecindad original ν_i^K y *extrusión* cuando el error de ranking es negativo. Dependiendo del valor de K se pueden definir las K-intrusiones y K-extrusiones con lo que se pueden distinguir también entre intrusiones (extrusiones) leves o duras. Los diferentes tipos de intrusiones y extrusiones se pueden representar con bloques de la matriz de co-ranking. La primera gran separación en bloques de la matriz se hace a partir de las K-vecindades separando la matriz en cuatro bloques con las primeras K filas y columnas. De esta forma se definen $\mathbb{F}_K = \{1, ..., K\}$ y $\mathbb{L}_K = \{K+1, ..., N-1\}$, con lo que los conjuntos de indices que separan la matriz en los bloques arriba-izquierda (*upper-left*), arriba-derecha (*upper-right*), abajo-izquierda (*lower-left*) y abajo-derecha (*lower-right*) son $\mathbb{UL}_K = \mathbb{F}_K \times \mathbb{F}_K$, $\mathbb{UR}_K = \mathbb{F}_K \times \mathbb{L}_K$, $\mathbb{LL}_K = \mathbb{L}_K \times \mathbb{F}_K$ y $\mathbb{LR}_K = \mathbb{L}_K \times \mathbb{L}_K$. Asi mismo el bloque $\mathbb{UL}_K$ se puede subdividir en su diagonal principal $\mathbb{D}_K = \{(i,j) : 1 \leq i \leq K\}$ y los triángulos superiores e inferiores $\mathbb{UT}_K = \{(i,j) : 1 \leq i \leq K \;\; j > i\}$ y $\mathbb{LT}_K = \{(i,j) : 1 \leq i \leq K \;\; j < i\}$.

A partir de estos conjuntos se pueden definir las siguientes cantidades

$$\begin{aligned} U_N(K) &= \frac{1}{KN} \sum_{(k,l) \in \mathbb{UT}_K} q_{kl} \\ U_X(K) &= \frac{1}{KN} \sum_{(k,l) \in \mathbb{LT}_K} q_{kl} \\ U_P(K) &= \frac{1}{KN} \sum_{(k,l) \in \mathbb{D}_K} q_{kl} \end{aligned} \tag{2.25}$$

Las primeras dos (U_N y U_X) corresponden a las fracciones de intrusiones y extrusiones leves, mientras que U_P muestra la fracción de vectores que tienen el mismo ranking en v_i^K y n_i^K. La suma de estos tres términos define una nueva cantidad llamada Q_{NX}, la cual también depende del valor

de K

$$Q_{NX}(K) = U_P(K) + U_N(K) + U_X(K) \tag{2.26}$$

Índice de Dunn

El índice de Dunn [38] es una medida utilizada principalmente en *clustering* y que sirve para medir cuan compactos y separados son un conjunto de *clusters*. Dada una partición U, El índice de Dunn se define como:

$$\alpha(c, U) = \frac{\min_{1 \leq q \leq c} \min_{1 \leq r \leq c,\, r \neq q} \operatorname{dist}(C_q, C_r)}{\max_{1 \leq p \leq c} \operatorname{diam}(C_p)}, \tag{2.27}$$

donde C_i es el i-ésimo *cluster* , $\operatorname{dist}(C_q, C_r)$ es la distancia entre clusters y $\operatorname{diam}(C_p)$ es el diámetro del *cluster*. Un valor mayor del índice implica *clusters* más separados y compactos. Si bien por lo general este índice no se usa para medir calidad de proyección, pues no mide la preservación topológica, sirve como medida de la capacidad del método para preservar la separación de clases (*clusters*), lo que puede ser atractivo en ciertas aplicaciones.

Clasificador k-NN

Cuando existe información sobre las clases o etiquetas de los datos, se puede utilizar un clasificador para medir la capacidad de clasificación en el espacio de salida o proyectado. Una buena proyección separará correctamente las clases en el espacio de salida por lo que se pueden tener altos porcentajes de clasificación. En este caso se considera la utilización del clasificador k-NN (*k-Nearest Neighbor*s) [39], el cual utiliza la información sobre la clase de k vecinos para determinar la clase del dato a clasificar. Esto se hace mediante una votación por lo que el k escogido debe ser impar.

2.2. Teoría de la Información

La TI nació de una forma casi casual mientras los investigadores buscaban diseñar sistemas de comunicación eficientes y confiables por lo que sus orígenes fueron más bien prácticos. Años después la TI se ha convertido en una teoría matemática muy profunda explicando la verdadera esencia de las comunicaciones [40]. Esta teoría permite medir que tan bien se representa la información y las limitaciones existentes al transmitirla por cierto canal de comunicación, permitiendo establecer de

manera precisa los límites óptimos para el diseño de los sistemas de comunicación.

Uno de los mayores logros que ha permitido alcanzar la TI es la posibilidad de extraer información directamente de los datos, lo que es clave en sistemas de aprendizaje a través de ejemplos, ya sean biológicos o artificiales. Un sistema que aprende a través de ejemplos tiene como objetivo el capturar la información de un conjunto de observaciones de un evento real, a través de los parámetros de una máquina de aprendizaje. Hasta ahora las redes neuronales y los filtros adaptivos han usado principalmente medidas de correlación entre las señales y las salidas. Esto a pesar de que la correlación al ser un momento de segundo orden no permite hacer una medición completa sobre la equivalencia de información entre la salida del sistema y la salida deseada.

En las siguientes secciones se mostrarán los conceptos básicos de TI y como puede utilizar esta teoría para entrenar una máquina de aprendizaje, con el fin de obtener información directamente de los ejemplos.

2.2.1. Entropía de Shannon

Shannon [18, 41] extendió las ideas de Nyquist [42] y Hartley [43] y planteó una medida para la cantidad de información de una variable con cierta distribución de probabilidad a la que llamó *entropía de información*. Dada una variable aleatoria discreta x, se define un conjunto de posibles valores que x puede tomar como $A = \{a_1, ..., a_m\}$. Para cada uno de esos valores existe asociada una probabilidad p_k, que es la probabilidad de que la variable x tome el valor a_k. Dado esto, se cumple que $p_k \geq 0 \ (k = 1, \ldots, m)$, $\sum_{k=1}^{m} p_k = 1$ con m el número de eventos posibles de la variable x. La información asociada a un evento x_k se define como:

$$h = \log \frac{1}{p_k}. \tag{2.28}$$

La ec. (2.28) implica que aquellos eventos que tienen mayor probabilidad de ocurrencia entregan menos información que los menos probables. El logaritmo a utilizar es independiente de la definición, sin embargo dada su aplicación al mundo de las telecomunicaciones generalmente se le asocia con el logaritmo en base 2, con lo que la información queda medida en *bits*.

Un ejemplo típico es aquel en que la variable es equiprobable, por ejemplo en el lanzamiento de una moneda, los eventos {cara, sello} tienen la misma probabilidad $\{1/2, 1/2\}$, por lo que conocer el valor de cada uno de los eventos entrega la misma información $h = \log \frac{1}{1/2} = 1$. Ahora si se

supone una moneda con sesgo, donde los eventos {cara, sello} tuvieran la distribución $\{3/4, 1/4\}$, la información obtenida al observar el evento "cara", 0.42 bits, es mucho menor que la que se obtiene al observar el evento "sello", 4 bits.

Al extender la idea del contenido de información de un evento a una variable aleatoria, se define la entropía de una variable aleatoria como la esperanza del contenido de información (o contenido medio de información) de todos los posibles eventos:

$$H_s = E\{h\} = -\sum_{k=1}^{N} p_k \log p_k. \tag{2.29}$$

De acuerdo a esta definición, una variable equiprobable tendrá mayor información promedio que una donde existan eventos mucho más probables que otros. Si se vuelve al ejemplo anterior, el contenido medio de información para el caso de la moneda sin sesgo es de 1 bit, mientras que en el caso de la moneda con sesgo, el contenido medio de información es de 0,81 bits. Shannon demostró que la entropía de una variable aleatoria equivale a la cantidad mínima de bits requeridos para transmitir dicha señal.

En el campo de la TI una propiedad importante es que la entropía de una señal o variable contiene todos los estadísticos de la misma, no solo los de primer y segundo orden ya que trabaja directamente sobre la distribución de probabilidad de los datos. Esto implica que es una medida mucho más completa, pero a la vez mucho más difícil de calcular que las basadas sólo en la media o la varianza (primer y segundo orden), ya que requiere el conocimiento de la distribución o una estimación de la misma, mientras la media y la varianza son más simples de estimar directamente de los datos.

Existen muchas propiedades para la entropía de Shannon, dentro de las cuales se destacan:

1. La medida de entropía $H(p_1, ..., p_n)$ es una función continua de todas las probabilidades p_k.

2. $H(p_1, ..., p_n)$ es simétricamente permutable. La permutación de cualquier p_k no cambia la incerteza de la distribución por lo que no cambia la medida de entropía.

3. $H(1/n, ..., 1/n)$ es una función monótona creciente de n. En una distribución uniforme (o equiprobable), al aumentar el número de posibles elecciones n, la incerteza crece, por lo que la medida de entropía también lo hace.

4. Recursividad: implica que la entropía de n estados puede ser expresada en términos de la

entropía de $n-1$ estados posibles más una suma ponderada de las salidas combinadas,

$$H_n(p_1, p_2, ..., p_n) = H_{n-1}(p_1 + p_2, p_3, ..., p_n) + (p_1 + p_2)H_2\left(\frac{p_1}{p_1 + p_2}, \frac{p_2}{p_1 + p_2}\right) \tag{2.30}$$

5. Aditividad: si se tienen dos distribuciones de probabilidad independientes $p = (p_1, ..., p_n)$ y $q = (q_1, ..., q_n)$, entonces la entropía de la fdp conjunta es la suma de las entropías individuales $H(p, q) = H(p) + H(q)$.

La entropía de Shannon no es la única función que cumple con estas propiedades por lo que existen definiciones alternativas para la entropía como las que se presentan en la siguiente sección.

Dado que muchas veces las variables sobre las que se trabaja no son discretas si no que continuas, el concepto de entropía se puede extender definiendo la entropía diferencial de Shannon, conocida muchas veces simplemente como entropía. Para una variable aleatoria continua $x \in \Re$, la entropía diferencial se define como

$$\begin{aligned} H(x) &= \int f(x) \frac{1}{\log f(x)} dx, \\ &= -\mathbb{E}[\log f(x)] \end{aligned} \tag{2.31}$$

donde $f(x)$ se conoce como la función de distribución de probabilidad (fdp) la cual define la probabilidad de la ocurrencia de cada evento de la variable continua x.

Para el caso multivariable donde se tienen distribuciones conjuntas, se puede definir la entropía de la variable conjunta como

$$H(x, y) = \int \int f(x, y) \frac{1}{\log f(x, y)} dxdy, \tag{2.32}$$

con $f(x, y)$ la distribución conjunta de las variables. Así mismo, la entropía condicional se define como

$$H(x|y) = \int \int f(x, y) \frac{1}{\log f(x|y)} dxdy. \tag{2.33}$$

Con éstas dos definiciones en mano se puede definir una relación entre ambas entropías de la forma

$$H(x, y) = H(x) + H(x|y) = H(y) + H(y|x), \tag{2.34}$$

de lo que se desprende que cuando x e y son independientes entonces $H(x, y) = H(x) + H(y)$.

2.2.2. Entropía de Renyi

Debido a la dificultad que presenta trabajar con la entropía de Shannon cuando sólo se conocen algunos datos o muestras de una o más variables aleatorias y no se tiene información sobre la distribución de los datos (probabilidad de ocurrencia de cada evento), Principe et. al [17] propusieron una nueva forma de hacer estos cálculos a partir de la entropía de Renyi [19, 44].

En teoría, una media de la suma de n números reales $x_1, ..., x_n$, con pesos $p_1, ..., p_n$ tiene la forma

$$\varphi^{-1}\left(\sum_{i=1}^{n} p_k \varphi\left(x_k\right)\right) \tag{2.35}$$

donde $\varphi(x)$ es la función de Kolgomorov-Nagumo, una función cualquiera continua y estrictamente monótona definida en los reales. De esta forma se puede definir una expresión generalizada para la entropía como:

$$\varphi^{-1}\left(\sum_{i=1}^{n} p_k \varphi\left(I\left(p_k\right)\right)\right), \tag{2.36}$$

donde $I = -\sum_{k=1}^{N} p_k \log p_k$.

Dado que la información es aditiva, $\varphi()$ está restringida a ser $\varphi(x) = x$ ó $\varphi(x) = 2^{(1-\alpha)x}$. Al utilizar $\varphi(x) = x$ se recupera la entropía de Shannon. Al usar $\varphi(x) = 2^{(1-\alpha)x}$ se obtiene la entropía de Renyi de orden α que se define como:

$$H_\alpha = \frac{1}{1-\alpha} \log\left(\sum_{k=1}^{N} p_k^\alpha\right) \qquad \alpha > 0,\ \alpha \neq 1. \tag{2.37}$$

A partir de la entropía de Renyi se puede recuperar la entropía de Shannon bajo la siguiente relación:

$$\lim_{\alpha \to 1} H_\alpha = H_s, \tag{2.38}$$

siendo H_s la entropía de Shannon.

La entropía diferencial de Renyi, sale directamente de la expresión anterior al integrar sobre el espacio de la fdp. En especial es interesante observar el caso con $\alpha = 2$, que equivale a la entropía cuadrática de Renyi:

$$\begin{cases} H_\alpha(Y) = \frac{1}{1-\alpha} \log\left(\int f_Y(z)^\alpha \, dz\right) \\ H_2(Y) = -\log\left(\int f_Y(z)^2 \, dz\right) \end{cases} \tag{2.39}$$

Una observación importante es que la entropía de Renyi posee el mismo óptimo global que la entropía de Shannon en términos de maximización y minimización [45].

2.2.3. Estimadores de Densidad

Uno de los principales problemas para el cálculo de las medidas de TI es la necesidad de conocer la fdp de los datos, ya que la mayoría de las veces solo se tiene una muestra finita de los mismos y no se conoce la función generadora. Para resolver este problema se pueden usar estimadores ya sean paramétricos o no paramétricos.

2.2.3.1. Estimadores Paramétricos

En el caso paramétrico se asume que los datos son generados por algún tipo de distribución conocida o una mezcla de las mismas. Ejemplo de esto es el estimador de máxima verosimilitud [46] (del que se entregarán detalles más adelante en la sección 2.2.8). Por otro lado los modelos de mezclas [47–49], considerados también estimadores semi-paramétricos, asumen que la densidad desconocida es generada por una mezcla de k densidades

$$f(x) = \sum_{j=1}^{k} f(x|j) \, p_j, \tag{2.40}$$

dónde $\sum_{j=1}^{k} p_j = 1$. Este modelo asume que cada dato puede ser determinado por alguna de las k densidades con probabilidad p_j. Cada densidad es definida en forma paramétrica de la forma $f(x|j;w)$, por lo que los parámetros desconocidos w y p_j deben ser determinados a partir de los datos.

El principal problema de estos estimadores es que si se quieren usar como criterio para entrenar alguna máquina de aprendizaje, será necesario resolver el problema de optimización de los parámetros del estimador (ej.: w, p_j), dentro del problema de optimización más general de la máquina de aprendizaje. Un segundo problema es que la elección de las distribuciones puede ser compleja y es difícil encontrar un modelo de propósito general, es decir, que se adapte a cualquier distribución de

los datos.

2.2.3.2. Estimadores No-Paramétricos

A diferencia de los estimadores paramétricos, estos estimadores no requieren asumir algún tipo de distribución a priori ni tampoco requieren optimizar parámetros del mismo. Algunos de los estimadores más utilizados son:

Histogramas

Dado un origen al sistema x_0 y un ancho para cada compartimiento h, los compartimientos de cada histograma se definen como $[x_0 + mh, x_0 + (m+1)\,h]$ para valores enteros de m. Se define entonces la estimación del histograma como [50, 51]

$$\hat{f}\,(x) = \frac{1}{Nh} n_i, \tag{2.41}$$

donde n_i es el número de ejemplos de x que caen en el compartimiento y N es el número total de datos. El problema de este estimador es que no es continuo lo que limita su aplicabilidad. Por otro lado sufre de forma muy fuerte de la llamada maldición de la dimensionalidad, ya que al aumentar la dimensión de los datos, la división del espacio en compartimientos se vuelve más costosa. Además, la mayoría de los compartimientos quedan vacíos. La figura 2.4 muestra un ejemplo de aproximación de una distribución gaussiana a partir de un histograma.

Figura 2.4: Estimador de una distribución gaussiana, utilizando un histograma.

Estimador Naive

Se deriva como una extensión al uso de histogramas, donde la fdp evaluada en x puede ser estimada como

$$f(x) = \lim_{h \to 0} \frac{1}{2h} P\,(x - h < X < x + h)\,. \tag{2.42}$$

La probabilidad $P\left(x-h<X<hx+h\right)$ puede ser estimada contando el número de datos que caen dentro de una caja de ancho $2h$ centrada en x. Esto se puede expresar a través de una función de pesos

$$I(x)=\begin{cases}\frac{1}{2},\ |x|<1\\ 0,\ \text{otro caso}\end{cases} \tag{2.43}$$

Con esta función el estimador se puede expresar como [51]

$$\hat{f}\left(x\right)=\frac{1}{Nh}\sum_{i=1}^{N}I\left(\frac{x-x_i}{h}\right). \tag{2.44}$$

Este estimador evita la complicación de escoger un punto de origen, pero sigue teniendo el problema de no ser una función continua.

Estimador Ventana de Parzen

Si la función de pesos del estimador Naive se reemplaza por una función no negativa o kernel que satisfaga la condición

$$\int K\left(x\right)dx=1, \tag{2.45}$$

se obtiene un estimador basado en kernels llamado Ventana de Parzen [20]. Por lo general el kernel se escoge de modo que sea simétrico y unimodal. El estimador de densidad de Parzen se escribe como

$$\hat{f}\left(x\right)=\frac{1}{N\sigma}\sum_{i=1}^{N}K\left(\frac{x-x_i}{\sigma}\right)=\frac{1}{N}\sum_{i=1}^{N}K_\sigma\left(x-x_i\right). \tag{2.46}$$

Si el kernel escogido es gaussiano, entonces se puede escribir como

$$K_\sigma\left(x-x_i\right)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{\left(x-x_i\right)^2}{2\sigma^2}\right\}. \tag{2.47}$$

Esto implica que sobre cada dato x_i se ubica un kernel de varianza σ^2, que define el tamaño del mismo. Esta suma de gaussianas genera una función continua y diferenciable en cualquier orden lo que permite que sea utilizada en una gran cantidad de problemas [51]. La elección del tamaño del kernel σ no es simple y generará un *trade off* entre sesgo y varianza del estimador. La figura 2.5 muestra un ejemplo de estimación de una distribución gaussiana utilizando una sumatoria de kernels gaussianos.

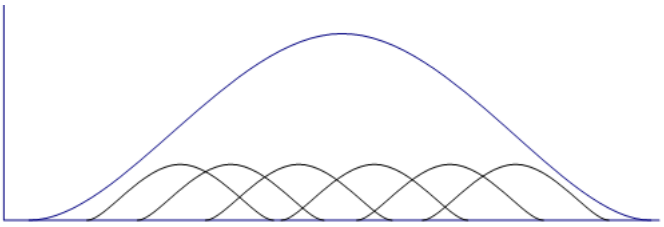


Figura 2.5: Estimador de una distribución gaussiana usando kernels gaussianos locales.

Para medir cuan cercana es la estimación de Parzen a la distribución real se utiliza el Error Cuadrático Medio Integrado (MISE),

$$MISE\left\{\hat{f}\left(x\right)\right\} = \int E\left\{\left[\hat{f}\left(x\right) - f(x)\right]^2\right\} dx. \tag{2.48}$$

Al descomponer esta ecuación en términos de sesgo y varianza se obtiene

$$MISE\left\{\hat{f}\left(x\right)\right\} = \int \left[E\left\{\hat{f}\left(x\right)\right\} - f(x)\right]^2 dx + \int Var\left\{\hat{f}\left(x\right)\right\} dx. \tag{2.49}$$

Si se analiza la ecuación (2.49) asintóticamente, cuando N es muy grande, asumiendo un kernel suave como el gaussiano y bajo la condición de que $\sigma(N)$ se aproxime a cero a una tasa menor a N^{-1}, se puede demostrar que el error cuadrático medio integrado asintótico (AMISE) está dado por [52]

$$AMISE\left\{\hat{f}\left(x\right)\right\} = \frac{\sigma^4 \mu_2^2(K) R(f'')}{4} + \frac{R(K)}{\sigma N}, \tag{2.50}$$

dónde $\mu_2 = \int z^2 K(z) dz$, $R(f") = \int \{f''(x)\}^2 dx$. Si se observa la parte derecha de la ecuación, se tienen primero los términos correspondientes al sesgo y luego a la varianza. De estos términos se deduce que el sesgo se minimiza al minimizar σ sin embargo la varianza se minimiza maximizando σ. Esto demuestra la existencia de un compromiso entre el sesgo y la varianza en el estimador de Parzen. Esta es la razón por la que no es simple escoger un buen valor de σ. Una forma que parece más o menos obvia es utilizar (2.50) directamente, por lo que si esta ecuación se deriva con respecto a σ e iguala a cero se obtiene la relación

$$\sigma_{AMISE} = \left[\frac{R(K)}{\mu_2^2(K) R(f") N}\right]^{\frac{1}{5}}, \tag{2.51}$$

donde $R(f")$ se puede estimar asumiendo una distribución normal. Al hacer esto se obtiene una aproximación para σ [51]

$$\hat{\sigma}_{AMISE} = 1{,}06 \cdot \hat{\sigma} N^{-\frac{1}{5}}, \tag{2.52}$$

con $\hat{\sigma}$ una estimación de la desviación estándar de la distribución normal supuesta. El problema al asumir una distribución gaussiana es que este resultado generalmente lleva a escoger un σ demasiado grande, haciendo la estimación demasiado suavizada.

Otra forma de estimar σ viene de la idea de la validación cruzada, donde el objetivo es encontrar el valor de σ que minimice el MISE, lo que es equivalente a minimizar

$$MISE\left\{\hat{f}(x)\right\} - \int f^2(x)dx = E\left[\hat{f}^2(x) - 2\int \hat{f}(x) f(x)dx\right]. \tag{2.53}$$

La expresión derecha puede ser estimada por el estimador sin sesgo [52]

$$LSCV(\sigma) = \int \hat{f}^2(x) - 2N^{-1}\sum_{i=1}^{N}\hat{f}_{-i}(x_i), \tag{2.54}$$

dónde $\hat{f}_{-i} = \frac{1}{N-1}\sum_{j\neq i} W_\sigma(x - x_j)$ es la densidad estimada al eliminar el ejemplo i-ésimo. Por lo tanto, minimizar (2.53) es equivalente a minimizar (2.54). La forma de minimizar (2.54) es conocida como validación cruzada de mínimos cuadrados (*least squares cross-validation*), ya que se utiliza información de algunos ejemplos para estimar información sobre otros. En [53] se encontró que el σ_{LSCV} tiene una varianza muy grande debido al hecho de ser un estimador sin sesgo, lo que provoca que el valor de σ sea por lo general muy pequeño [52].

La extensión del estimador de Parzen al caso d-dimensional es directa, donde se mantiene la condición de que la integración del kernel debe ser igual a uno.

$$\hat{f}(x) = \frac{1}{Nd^2}\sum_{i=1}^{N} K\left(\frac{x - x_i}{\sigma}\right) = \frac{1}{N}\sum_{i=1}^{N} W_\sigma(x - x_i), \tag{2.55}$$

dónde W_σ es generalmente un kernel gaussiano de la forma

$$W_\sigma(x - x_i) = \frac{1}{(2\pi\sigma^2)^{\frac{d}{2}}}\exp\left\{-\frac{\|x - x_i\|^2}{2\sigma^2}\right\}. \tag{2.56}$$

Otra de las ventajas, además de su continuidad y diferenciabilidad, que tiene el usar un kernel gaussiano es que su naturaleza híper-esférica en el caso d-dimensional permite utilizar un tamaño de kernel σ igual para todas las dimensiones. En el caso general se debería estimar una matriz de covarianzas completa, sin embargo esto trae consigo una dificultad mucho mayor que la estimación de σ. Al aplicar la regla de Silverman se obtiene la expresión para el caso multidimensional [51]

$$\sigma_{AMISE} = \hat{\sigma} \left[\frac{4}{(2d+1)\, N} \right]^{\frac{1}{d+4}}, \tag{2.57}$$

donde $\hat{\sigma}^2 = d^{-1} \sum_i^d \sigma_{ii}$, y σ_{ii} son los elementos de la diagonal de la matriz de covarianzas de los datos.

En el caso multidimensional el estimador de Parzen sufre de la maldición de la dimensionalidad por lo que en dimensiones altas se requiere una gran cantidad de datos para obtener una buena estimación.

2.2.4. Entropía Cuadrática

Para calcular la entropía diferencial cuadrática de Renyi (2.39) se utiliza el estimador de ventana de Parzen, (2.55) con un kernel gaussiano definido como $G\left(y, \Sigma\right) = \frac{1}{(2\pi)^{M/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} y^T \Sigma^{-1} y\right)$ en el espacio M-dimensional, donde Σ es la matriz de covarianza, $y \in \Re^M$. Cuando se combinan la definición de entropía de Renyi con el estimador de Parzen, se puede conseguir un estimador de entropía basado en las muestras discretas de datos $\{y\}$. Dado un conjunto de datos $Y = \{y_1, ..., y_n\}$, $y_i \in \Re^d$, la estimación de la fdp utilizando la ventana de Parzen y un kernel gaussiano se define como

$$\begin{cases} \hat{f}\left(\{y\}\right) = \frac{1}{N} \sum_{i=1}^{N} G\left(y - y_i, \sigma^2 I\right) \\ G\left(y, \Sigma\right) = \frac{1}{(2\pi)^{M/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} y^T \Sigma^{-1} y\right) \end{cases} \tag{2.58}$$

Sean y_i e y_j ambas $\in \Re^M$ dos muestras de datos, Σ_1 y Σ_2 las matrices de covarianza para dos kernel gaussianos, se puede demostrar que se cumple la siguiente relación:

$$\int G\left(y - y_i, \Sigma_1\right) G\left(y - y_j, \Sigma_2\right) dz = G\left(\left(y_i - y_j\right), \left(\Sigma_1 + \Sigma_2\right)\right). \tag{2.59}$$

De esta misma forma se podría conseguir la integración para el producto de tres kernel gaussianos. La igualdad (2.59) puede ser interpretada como la convolución entre dos kernel gaussianos centrados

en y_i e y_j donde el resultado es una función gaussiana con covarianza igual a la suma de las covarianzas individuales y centrada en $d_{ij} = (y_i - y_j)$.

Al reemplazar la estimación de Parzen 2.58 en la fórmula para la entropía diferencial cuadrática de Renyi (2.39), se obtiene

$$\begin{cases} H(\{y\}) = H_2(Y \mid \{y\}) = -\log\left(\int f(y)^2 dy\right) = -\log V(\{y\}) \\ V(\{y\}) = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\int G\left(y - y_i, \sigma^2 I\right) G\left(y - y_j, \sigma^2 I\right) dz = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G(y_i - y_j, 2\sigma^2 I). \end{cases} \tag{2.60}$$

2.2.4.1. Potencial de Información

La combinación de la ventana de Parzen con la entropía de Renyi nos lleva a una estimación de la entropía basada en la interacción de pares de muestras. Vale destacar que hasta ahora no se ha hecho ninguna aproximación además de la estimación de la fdp. La expresión

$$V(y) = \frac{1}{N^2}\sum_i\sum_j G\left(y_i - y_j, 2\sigma^2 I\right), \tag{2.61}$$

es llamada también Potencial de Información (PI) debido a la analogía existente con el potencial gravitatorio entre partículas, ya que ambas decaen exponencialmente con la distancia al cuadrado de los puntos de origen, en este caso las partículas de información (PCI). De acuerdo a esta relación, el maximizar la entropía de los datos es equivalente a minimizar el PI, lo que se asemeja al principio físico de mínima energía.

2.2.4.2. Fuerzas de Información

Al igual que en el caso gravitatorio, es posible derivar desde el PI una fuerza de información (FI). En este caso las fuerzas interactúan para ajustar las distribuciones de los datos y de esta forma optimizar la entropía. Si se deriva el PI con respecto a una partícula y_i se obtiene la fuerza aplicada sobre la partícula por el resto de los datos.

$$\frac{\partial V}{\partial y_i} = \frac{1}{N^2}\sum_i\sum_j \frac{\partial}{\partial y_i} G\left(y_i - y_j, 2\sigma^2 I\right) = -\frac{1}{N^2\sigma^2}\sum_j G\left(y_i - y_j, 2\sigma^2 I\right)(y_i - y_j). \tag{2.62}$$

2.2.5. Información Mutua

Considerando un sistema estocástico donde una variable de entrada X produce una salida llamada Y. La información o incerteza que se tiene sobre X, sin haber hecho ningun otro tipo de observación está definida por su entropía diferencial $H(X)$. Si ahora se observa la salida del sistema Y, entonces la entropía condicional $H(X|Y)$ es la incerteza que queda sobre la entrada después de observar la salida del sistema. La diferencia de entropía que se genera, $I(X,Y) = H(X) - H(X|Y)$, es lo que se llama Información Mutua entre las variables X e Y. En otras palabras, la IM mide la información que contiene una variable sobre la otra, lo que puede ser visto como una relación de dependencia mucho más fuerte que la correlación. Cuando las variables son independientes entonces la IM es cero.

La IM es capaz de cuantificar una relación de entropías entre pares de variables aleatorias por lo que es una medida más general que la simple entropía de una variable lo que permite que sea aplicada en una mayor cantidad de problemas.

Algunas de las propiedades de la IM son:

1. No negatividad:

$$I(X,Y) \geq 0 \tag{2.63}$$

2. Simetría:

$$I(X,Y) = I(Y,X) \tag{2.64}$$

   Basado en las dos propiedades anteriores es posible mostrar las siguientes relaciones existente entre la IM y la entropía de Shannon [54, 55]

$$\begin{cases} I_s = H_s(X) - H_s(X|Y) \\ I_s = H_s(Y) - H_s(Y|X) \\ I_s = H_s(X) + H_s(Y) - H_s(X,Y) \end{cases} \tag{2.65}$$

   donde $H_s(X,Y)$ es la entropía conjunta y $H_s(X|Y)$ es la entropía condicional de Shannon de X dado Y.

3. Invarianza: La IM es invariante ante transformaciones invertibles de variables aleatorias.

La figura 2.6 muestra la relación existente entre la entropía y la IM entre dos variables aleatorias.

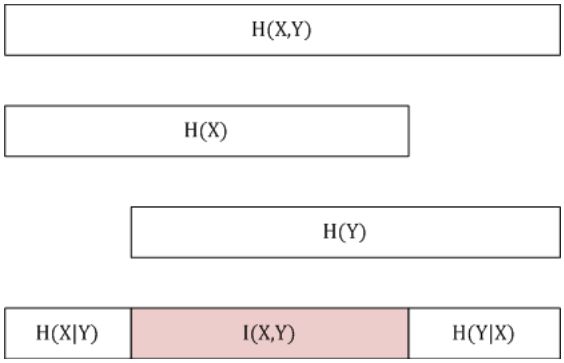


Figura 2.6: Relaciones entre entropía e información mutua

Según la definición hecha por Shannon [18, 41] la extensión de la IM a variables aleatorias continuas se establece como

$$I_s(X,Y) = \int\int f(x,y) \log \frac{f(x,y)}{f(x) f(y)} \tag{2.66}$$

donde $f(x,y)$ es la fdp de la variable conjunta (x,y), $f(x)$ y $f(y)$ son las fdp marginales de X e Y. Esta definición puede ser vista como la divergencia de Kullback-Leibler (KL) entre la fdp de la variable conjunta con las marginales [12, 54, 55], donde la divergencia de KL entre dos fdp $f(x)$ y $g(x)$ está definida como

$$KL(f,g) = \int f(x) \log \frac{f(x)}{g(x)} dx. \tag{2.67}$$

Esta divergencia mide la "distancia" entre dos fdps en el espacio de las funciones. Sin embargo el hecho de no ser simétrica y no cumplir con la desigualdad triangular, hace que no sea considerada una distancia como tal por lo que se habla más bien de una divergencia. El valor de la divergencia es siempre mayor o igual a cero y la igualdad se alcanza solo cuando $f(x) = g(x)$.

2.2.6. Información Mutua Cuadrática

Al igual que en el caso de la entropía, se pueden buscar definiciones alternativas para la IM que faciliten el cálculo de la misma. Sobre todo es interesante tener medidas que puedan utilizar las formas cuadráticas de las fdps.

Basados en la idea de que la IM es una medida de independencia estadística y que depende de la distancia entre la fdp conjunta y las marginales se pueden proponer otras alternativas que mantengan estas propiedades. La primera alternativa es utilizar la distancia euclidiana entre dos fdps lo que lleva a definir la *Información Mutua Cuadrática con Distancia Euclidiana* (IMC-DE, o ED-QMI en inglés).

$$\begin{cases} D_{DE}(f,g) = \int \left(f(x) - g(x)\right)^2 dx \\ I_{DE}\left(X_1, X_2\right) = D_{DE}\left(f(x_1,x_2), f_1(x_1)f(x_2)\right) \end{cases} \tag{2.68}$$

Esta medida es no negativa y alcanza el cero solo cuando la distribución conjunta es igual a la multiplicación de las marginales, por lo que cumple como medida de independencia.

Basado en la desigualdad de Cauchy-Schwartz [56]

$$\left(\int f\left(x\right)^2 dx\right)\left(\int g\left(x\right)^2 dx\right) \geq \left(\int f\left(x\right) g\left(x\right) dx\right)^2, \tag{2.69}$$

se puede definir la divergencia de Cauchy-Schwartz entre dos fdps como

$$D_{CS}\left(f,g\right) = \log \frac{\left(\int f\left(x\right)^2 dx\right)\left(\int g\left(x\right)^2 dx\right)}{\left(\int f\left(x\right) g\left(x\right) dx\right)^2}. \tag{2.70}$$

Al igual que en (2.68) se puede definir una medida de *Información Mutua Cuadrática de Cauchy-Schwartz* (IMC-CS o CS-QMI en inglés).

$$I_{cs}(X_1, X_2) = D_{CS}\left(f(x_1,x_2), f_1(x_1)f(x_2)\right). \tag{2.71}$$

En este caso también se cumple que la medida es no negativa y alcanza el cero sólo cuando la distribución conjunta es igual a la multiplicación de las marginales. Estas medidas representan una aproximación a la divergencia de Kullback-Leibler y sus propiedades cuadráticas serán de gran utilidad como se verá más adelante.

Si se tienen dos conjuntos de muestras correspondientes a dos variables aleatorias, $\{x\} = \{x_i, i = 1, \ldots, N\}$ para X y $\{y\} = \{y_i, i = 1, \ldots, N\}$ para Y, al utilizar la ventana de Parzen para estimar las fdps conjunta y marginales se obtiene:

$$\begin{cases} \hat{f}(x,y) = \frac{1}{N}\sum_{i=1}^{N} G\left(x - x_i, \sigma_x^2 I\right) G\left(y - y_i, \sigma_y^2 I\right) \\ \hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} G\left(x - x_i, \sigma_x^2 I\right) \\ \hat{f}(y) = \frac{1}{N}\sum_{i=1}^{N} G\left(y - y_i, \sigma_y^2 I\right) \end{cases} \tag{2.72}$$

Usando (2.72) se pueden definir tres términos que se repiten en las definiciones de ED-QMI y CS-QMI

$$\begin{cases} V_J = \iint f(x,y)^2 \, dxdy \\ V_M = \iint \left(f(x)f(y)\right)^2 dxdy \\ V_C = \iint f(x,y) \, f(x)f(y) dxdy \end{cases} \tag{2.73}$$

reemplazando las fdps por las estimaciones de Parzen

$$\begin{cases} V_J = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, \sigma_x^2 I\right) G\left(y_i - y_j, \sigma_y^2 I\right) \\ V_C = \frac{1}{N}\sum_{i=1}^{N}\left\{ \frac{1}{N}\sum_{j=1}^{N} G\left(x_i - x_j, \sigma_x^2 I\right) \cdot \frac{1}{N}\sum_{j=1}^{N} G\left(y_i - y_j, \sigma_y^2 I\right) \right\} \\ V_M = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, \sigma_x^2 I\right) \cdot \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(y_i - y_j, \sigma_y^2 I\right) \end{cases} \tag{2.74}$$

Cada uno de estos términos puede ser interpretado como un potencial de información tal como (2.61). Se puede definir entonces V_J como el potencial de información en el espacio conjunto. V_M es el potencial de información marginal y finalmente V_C es el potencial de información cruzado entre la fdp conjunta y las marginales. Con esto se pueden reescribir las medidas cuadráticas ya definidas como

$$\begin{cases} V_{DE} = V_J - 2V_c + V_M \\ V_{CS} = \log V_J - 2\log V_C + \log V_M \end{cases} \tag{2.75}$$

La medida V_{DE} es llamada Potencial de Información Cruzado con Distancia Euclidiana (ED-CIP) y V_{CS} Potencial de Información Cruzado de Cauchy-Schwartz.

Experimentalmente Principe [17] encontró que I_{DE} tiene un mejor comportamiento cuando se

desea maximizar la IMC que I_{CS}, mientras ambas ofrecen resultados similares cuando se desea minimizar.

2.2.6.1. Fuerzas de Información Cruzada

Tal como en el caso de la entropía, es posible derivar fuerzas de los potenciales de información. En este caso, las fuerzas provienen de varias fuentes o potenciales al mismo tiempo. La fuerza compuesta por la interacción de los tres potenciales se conoce como fuerza de información cruzada (FIC). Sean

$$\begin{cases} V_{ij}^{l} = G(y_i - y_j, 2\sigma^2 I), & l = 1,2 \\ V_{i}^{l} = \sum_{i=1}^{N} G(y_i - y_j, 2\sigma^2 I), & l = 1,2 \\ V^{l} = \sum_{i=1}^{N}\sum_{j=1}^{N} G(y_i - y_j, 2\sigma^2 I), & l = 1,2 \end{cases} \tag{2.76}$$

para el caso de $V_{ED}(y_1, y_2)$, la FIC, $\frac{\partial V_{ED}}{\partial y_i}$ se puede calcular como

$$\begin{gathered} c_{ij}^{k} = V_{ij}^{k} - V_{i}^{k} - V_{j}^{k} + V^{k}, \quad k = 1,2 \\ F_i^l = \frac{\partial V_{ED}}{\partial y_i^k} = -\frac{1}{N^2\sigma^2}\sum_{j=1}^{N} c_{ij}^{k} V_{ij}^{l} d_{ij}^{l} \\ i = 1,\ldots,N \qquad l \neq k. \quad l = 1,2 \end{gathered} \tag{2.77}$$

si se utiliza $V_{CS}(y_1, y_2)$, entonces la FIC se calcula como

$$\begin{aligned} F_i = \frac{\partial V_{CS}}{\partial y_i^k} &= \frac{1}{V_J}\frac{\partial V_J}{\partial y_i^k} - 2\frac{1}{V_C}\frac{\partial V_C}{\partial y_i^k} + \frac{1}{V_M}\frac{\partial V_M}{\partial y_i^k} \\ &= -\tfrac{1}{\sigma^2}\left[\frac{\sum_{j=1}^{N} V_{ij}^{1} V_{ij}^{2} d_{ij}^{k}}{\sum_{i=1}^{N}\sum_{j=1}^{N} V_{ij}^{1} V_{ij}^{2}} + \frac{\sum_{j=1}^{N} V_{ij}^{k} d_{ij}^{k}}{\sum_{i=1}^{N}\sum_{j=1}^{N} V_{ij}^{k}} - \frac{\sum_{j=1}^{N} (V_i^l + V_j^l) V_{ij}^{k} d_{ij}^{k}}{N\sum_{i=1}^{N}\sum_{j=1}^{N} V_i^1 V_i^2}\right] \end{aligned} \tag{2.78}$$

2.2.7. Aprendiendo a partir de los ejemplos

Una máquina de aprendizaje puede ser definida como un mapeo entre una señal de entrada y una de salida, la cual tiene la capacidad de adaptarse para cumplir con cierto objetivo. Un tipo de máquina de aprendizaje de particular interés son las Redes Neuronales, las cuales pueden resumir

la mayoría de los tipos existentes de modelos adaptivos. Estos modelos pueden ser lineales o no lineales, estáticos o dinámicos, etc. El aprendizaje en este tipo de redes se define como la capacidad de adaptar los parámetros de la red de modo de optimizar cierta función objetivo utilizando una cantidad finita de ejemplos disponibles. A este tipo de aprendizaje se le llama *aprender a través de ejemplos.*

Los criterios de optimización o aprendizaje pueden ser variados y su forma más clásica es minimizar el *error cuadrático medio* (MSE), sin embargo en este trabajo se desea utilizar alguno de los criterios de TI que se verán con más detalle en la sección 2.2.8. Cada uno de esos criterios puede ser utilizado con diversos modelos de aprendizaje.

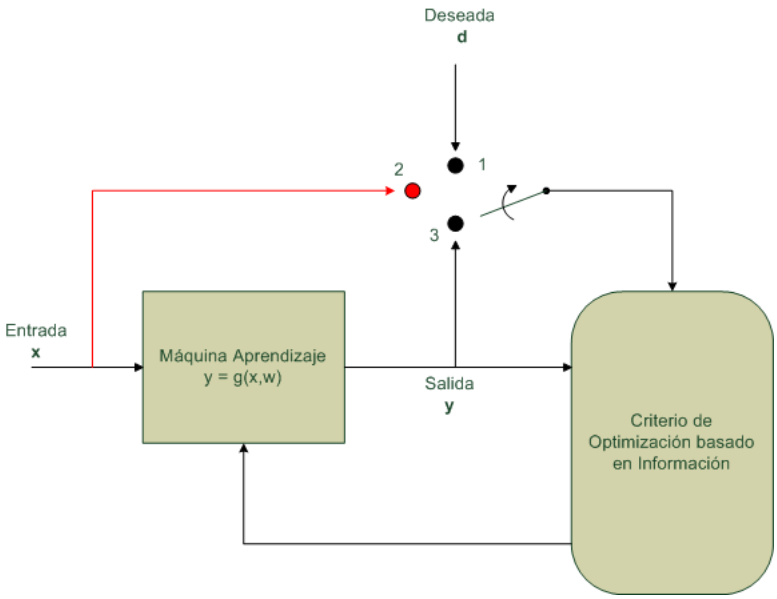


Figura 2.7: Esquema aprendizaje utilizando ITL

La figura 2.7 resume algunos modelos de aprendizaje que pueden ser optimizados utilizando algún criterio de TI. Dependiendo de la posición del selector se pueden distinguir tres tipos diferentes de aprendizaje. La posición 1 representa un sistema con aprendizaje supervisado, donde se desea que la salida de la red sea igual a cierta señal deseada, por lo que el criterio deberá comparar esta salida deseada con la salida de la red. Por lo general se utiliza el MSE para medir este error o diferencia, sin embargo también se puede utilizar una medida de divergencia como la IM, tal como hace Torkkola en [22] para realizar una proyección de datos basado en la clase de cada uno de ellos.

Cuando el selector está en las posiciones 2 o 3, se tiene un sistema de aprendizaje no supervi-

sado donde lo que cambia es la fuente de la información. La posición 3 indica que se utiliza sólo información de la salida y. En estos casos se pueden utilizar criterios como la maximización de la entropía de la salida o como en el modelo de análisis componentes independientes (ICA), se podría querer minimizar la dependencia entre diferentes componentes de la salida [57]. En el caso de la posición 2, se utiliza tanto la información de la entrada como de salida de la red. Un criterio de TI que se puede utilizar es una medida de divergencia entre la entrada y la salida, como la divergencia de KL o la IM. Este último caso equivale al criterio Infomax de Linsker [58].

Para entender mejor como funciona una máquina de aprendizaje se revisarán los modelos estáticos de los más conocidos [59]:

1. Modelo Lineal: El modelo de red más simple es el modelo lineal, que puede ser definido como una transformación lineal de un conjunto de vectores $\{x_i\} \in \Re^D$ a otro conjunto $\{y_i\} \in \Re^d$ con $d < D$.

$$Y = W^T X \tag{2.79}$$

donde W es una matriz de $D \times d$ parámetros. Se define X como la variable del espacio de entrada e Y como el espacio de salida. Las ventajas del modelo lineal son su simpleza y utilidad en un amplio rango de problemas, además de ser fácil de interpretar (figura 2.8). Las columnas de la matriz W definen los ejes de proyección.

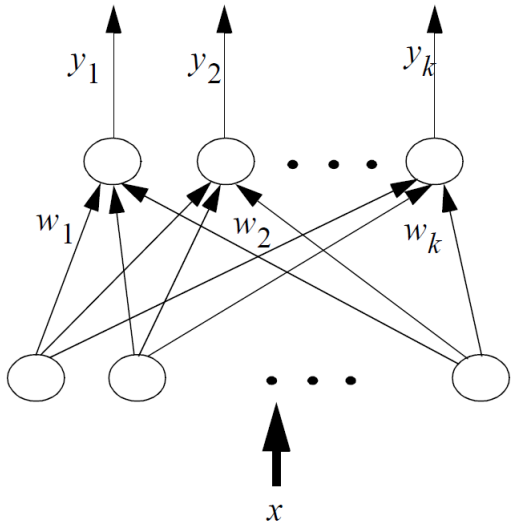


Figura 2.8: Red Lineal

2. Perceptrón: Modelo simplificado del funcionamiento de una neurona, presentado por Rosenblatt [60–62]. Este modelo se diferencia del caso lineal en que el nodo de la red es reemplazado por una función no lineal como la función escalón. Se considera como la primera máquina que aprende directamente de los ejemplos [63]. Su capacidad de aprendizaje se ve limitada a discriminaciones entre dos clases mediante hiperplanos, por lo que sólo resuelve problemas linealmente separables [62].

3. Perceptrón Multicapa (MLP): Es una extensión del modelo del perceptrón donde se mezclan varios perceptrones en distintas capas de neuronas formando una red neuronal. Las funciones de activación en cada nodo son funciones continuas diferenciables y no lineales, como por ejemplo, la función sigmoide $f(x) = 1/(1+e^{-x})$. En este modelo, además de la capa de entrada y salida, se tienen capas ocultas que ayudan a encontrar soluciones más complejas que un perceptrón no es capaz de hallar, como separaciones no lineales (figura 2.9). Cada nodo en la MLP es una unidad de procesamiento que simula el funcionamiento de una célula neuronal [55]. La propiedad más importante que tiene la MLP es su capacidad de aproximar cualquier tipo de función si es que se tiene la cantidad suficiente de nodos, por lo tanto si a esto se le suma la cantidad suficiente de datos para entrenar la MLP, esta puede aproximar cualquier mapeo [55, 64]. Esto se explica ya que el bloque básico de cada red neuronal es un hiperplano que es la proyección de cada nodo representada por la suma de las entradas. La función no lineal asociada permite curvar estos hiperplanos rígidos para obtener mejores aproximaciones. Como la MLP es finalmente una combinación de estos bloques básicos, una gran cantidad de ellos es capaz de aproximar cualquier mapa.

4. Funciones de Base Radial (RBF): Son redes de dos capas, donde la capa oculta es no lineal con una función de base radial como relación entre entrada y salida (figura 2.10). La función base más típicamente utilizada es una gaussiana. La capa de salida combina linealmente las salidas de la capa oculta. Este tipo de red comparte con la MLP la capacidad de ser un aproximador universal [55, 65, 66]. La diferencia con la MLP es que el bloque básico de construcción aproxima localmente en vez de globalmente como lo hace la MLP.

Además de estas redes estáticas existen variadas maquinas de aprendizaje dinámicas, es decir, redes que guardan información temporal, como Redes Feed Forward (FFNN), Time Delay Neural Networks (TDNN) [67, 68], Modelos Gamma [69, 70], etc.

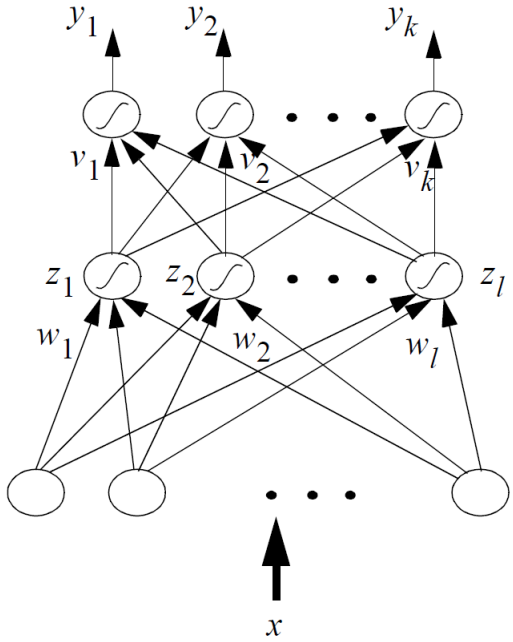


Figura 2.9: Red Neuronal MLP

2.2.8. Principios de Optimización de la Información

Considere un mapa paramétrico $g : \Re^D \rightarrow \Re^d$ de una variable aleatoria X con N ejemplos $\{x_i\} \in \Re^D, i = 1...N$, con $d < D$, descrito por $y_i = g(x_i, W)$, siendo Y una variable aleatoria de N ejemplos $\{y_i\} \in \Re^d, i = 1...N$, y W un conjunto de parámetros. El objetivo de ITL es determinar el valor de los parámetros W basados en la optimización de alguna medida de TI. La optimización es realizada utilizando sólo la información contenida en los valores de x_i e y_i y no se deben hacer supuestos sobre las fdp de los datos. El mapeo puede ser tanto lineal como no lineal, estático o dinámico, y el aprendizaje puede ser supervisado si se añade alguna información externa como la salida deseada o la etiqueta de las clases, o no supervisado si solamente se tiene la información de los datos de entrada. Para encontrar el conjunto de parámetros deseado existen variados criterios que pueden ser optimizados dentro de los cuales se encuentran los basados en el contenido de información en los datos. Cada uno de estos criterios puede ser aplicado a un problema diferente por lo que es importante tenerlos en consideración. A continuación se entrega un breve resumen de algunos criterios importantes.

Error Cuadrático Medio (MSE): Suponiendo que se tiene una red con entradas x_i y a cada

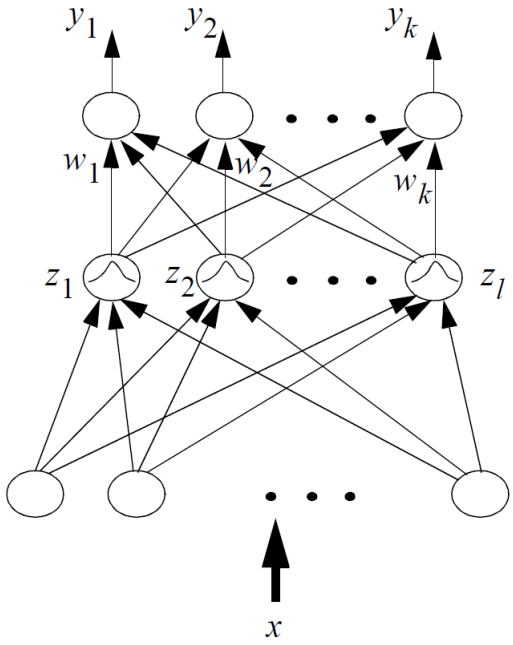


Figura 2.10: RBF

una de ellas se le asocia una salida y_i mediante algún mapeo $y_i = g(x_i, W)$, además se tiene una señal de salida deseada d_i como objetivo, se puede describir el error en la salida de la red como la diferencia $e_i = (d_i - y_i)$. El error cuadrático medio se define entonces como

$$MSE = \frac{1}{2}\sum_{i=1}^{N} e_i^2. \tag{2.80}$$

Como se puede observar el MSE corresponde a la distancia euclidiana entre la señal de salida y la señal deseada. Si se asume que el error es una señal gaussiana de media cero, entonces el minimizar el MSE es equivalente a minimizar la entropía del error.

Razón Señal a Ruido: Este principio busca maximizar la razón señal a ruido en la salida de una red. Un ejemplo de uso es PCA ya que utiliza este criterio como motivación para encontrar las proyecciones de mayor varianza. En el caso lineal y asumiendo una señal gaussiana, el criterio es equivalente a maximizar la diferencia de entropías entre la señal y el ruido.

Máxima Verosimilitud (MLE): La estimación por máxima verosimilitud [46, 71] (*likelihood* en ingles) es una de las técnicas estadísticas más utilizadas para estimar los parámetros desconocidos de un modelo a través de los ejemplos o datos disponibles. Sea $g(x, w)$ un mo-

delo estadístico de la variable x, con w un conjunto de parámetros. Sea $f(x)$ la distribución real de x, se quiere optimizar el conjunto de parámetros w tal que $g(x,w)$ sea lo más parecida posible a $f(x)$. Para medir esta diferencia se puede utilizar la divergencia de KL tal como se definió en (2.67).

$$D_{KL}(w) = \int f(x) \log \frac{f(x)}{g(x,w)} dx = -E\left[\log g(x,w)\right] + H_s\left(x\right). \tag{2.81}$$

Como la entropía de Shannon no depende del conjunto de parámetros, minimizar la divergencia equivale a maximizar la función de verosimilitud de $g(x,w)$, $L(w) = E\left[\log g(x,w)\right]$.

Si bien algunos de estos criterios se pueden asociar con elementos de TI, éstos no están motivados directamente en la información contenida en los datos en el sentido de Shannon. A continuación se presentan algunos criterios desarrollados dentro del marco de la TI.

MaxEnt: Este principio propuesto por Jaynes [72, 73], busca la distribución de la variable de salida Y que maximice la entropía de Shannon sujeto a alguna restricción. Si se supone que se posee un sistema con cierto conjunto de variables de las cuales no se conoce su probabilidad pero si ciertas restricciones sobre su distribución, el problema se reduce a encontrar el modelo que optimice cierto criterio mientras se cumplen las condiciones establecidas. Dado que los posibles modelos son infinitos, Jaynes establece el principio de *máxima entropía*:

> *Cuando una inferencia es hecha sobre la base de información incompleta, esta debe ser establecida sobre la base de la distribución de probabilidad que maximiza la entropía, sujeto a las restricciones sobre la distribución.*

Este principio ha sido aplicado como ejemplo al problema conocido como Separación Ciega de Fuentes, *(Blind Source Separation, BSS)* [74]. También se ha utilizado para el aprendizaje en redes neuronales [75, 76].

MinEnt: Consiste en minimizar la entropía de la variable de salida Y. Puede ser utilizado para resolver problemas como deconvolución ciega, clasificación o filtros de información.

MinXEnt: Principio conocido como Kullback's MinXEnt [73], busca una distribución de la variable de salida Y que minimice la distancia en el espacio de las probabilidades con una distribución dada (ej. la distribución de la entrada X, o la distribución de las clases). La

medida más utilizada para medir esta diferencia es la divergencia de Kullback-Leibler. Una de las ventajas de este principio es la invarianza a las transformaciones de coordenadas. Este esquema se ha utilizado ampliamente para resolver el problema de BSS [54, 77–79].

Infomax: También conocido como el principio de máxima preservación de información de Linsker [58], es un caso especial del principio de pérdida de información de Plumbey [80]. El objetivo es determinar el conjunto de parámetros W, tal que la salida Y contenga la mayor cantidad de información posible sobre la entrada, X. Inspirado en los sistemas biológicos auto-organizativos, Linsker propuso como solución al problema, el maximizar la información mutua promedio entre la entrada X y la salida Y. En una MLP, este concepto se aplica capa a capa.

> *La transformación de un vector aleatorio X, observado en la capa de entrada de un sistema neuronal a un vector aleatorio Y producido en la capa de salida del sistema debe ser escogida de modo que las actividades de la neuronas en la capa de salida conjuntamente maximicen la información sobre las actividades en la capa de entrada. La función objetivo a ser maximizada es la información mutua $I(X, Y)$ entre los vectores X e Y.*

Si se asumen distribuciones gaussianas y un mapeo lineal, entonces la información mutua es maximizada al maximizar la varianza del espacio de salida [58], lo que sería equivalente al resultado de PCA. Cuando el sistema es determinístico, la maximización de la IM es equivalente a maximizar la entropía de salida [58, 74]. Esto se puede demostrar de la relación $I(X, Y) = H(Y) - H(Y|X)$. Si el modelo es determinístico entonces la entropía condicional $H(Y|X)$ es cero, por lo que sólo queda la entropía de la salida. Sin embargo el principio de Infomax puede aplicarse también a cualquier par de variables aleatorias como la salida del mapeo con alguna variable aleatoria externa. El problema es que obtener una solución analítica es muy difícil a menos que se asuman restricciones sobre la fdp como gaussianidad y mapeo lineal [54].

2.3. Problema Espacio Vacío

El problema abordado es esta tesis es el de proyectar datos multidimensionales en dos o tres dimensiones, sin embargo, es válido preguntarse si existe un límite de dimensiones sobre el cual el

algoritmo es incapaz de proyectar. Verleysen en [81] entrega un análisis sobre los problemas que pueden tener algunos métodos cuando la dimensionalidad de los datos aumenta sobre cierto límite. En general las redes neuronales actuan como interpoladores entre los datos de entrenamiento, por lo que mientras más datos existan en cierta zona del espacio es más fácil para la red neuronal ajustarse a los datos, asi mismo, si la cantidad de datos es muy baja la red neuronal tendrá problemas para ajustarse. Por otro lado intuitivamente podemos deducir que una función compleja necesitará más datos de muestra para ser bien aproximada por una red que una función más simple. El problema de la falta de datos en alta dimensión es llamado "fenómeno del espacio vacío".

Si se considera el volumen de una esfera de radio r en dimensión d se puede describir la siguiente recurrencia

$$V(d) = V(d-2)\frac{2\pi}{d}r^2, \tag{2.82}$$

con $V(1) = 2$ y $V(2) = \pi$. Se puede apreciar que el volumen de la esfera decrece a cero a medida que se incrementa el valor de d. Esto se puede ver en la figura 2.11 para una esfera de radio unitario. La caída del volumen es tan drástica que en dimensión 20 éste es practicamente igual a cero.

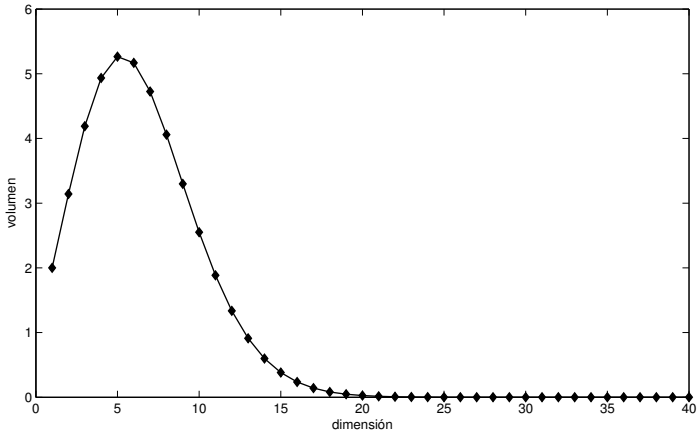


Figura 2.11: Volumén esfera versus dimensión

Otra forma de considerar el efecto es comparándolo con el volumen de una figura más familiar como es el volumen de un cubo. En la figura 2.12 se muestra la razón entre el volumen de una esfera y el volumen de un cubo cuyo lado es igual al diametro de la esfera. Se observa que el volumen de la esfera es menor al 10 % del volumen del cubo ya en dimensión 6.

Ya que el método propuesto en esta tesis se basa en el cálculo de kernels gaussianos, es importante

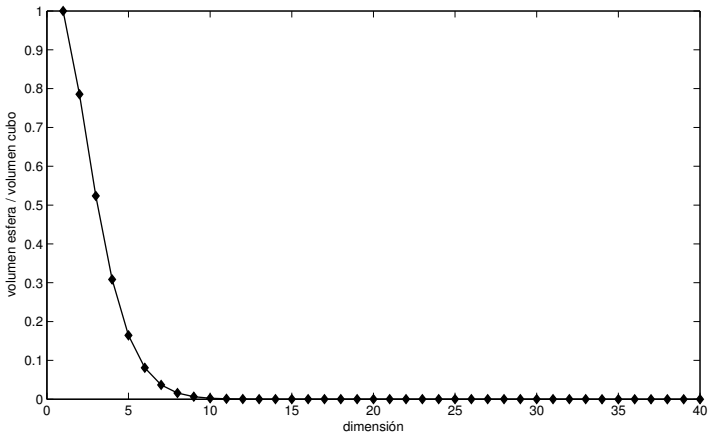


Figura 2.12: Razón entre volumen esfera y volumen de un cubo versus la dimensión

saber qué pasa con ellos en espacios de alta dimensión. El kernel gaussiano se utiliza principalmente debido a sus propiedades locales, que intuitivamente señalan que la mayor parte del volumen del mismo se encuentra en una zona cercana al centro. El 90 % de los datos en una distribución gaussiana escalar normalizada ($\mu = 0, \sigma = 1$) cae dentro del intervalo [-1.65, 1.65], el problema es que este porcentaje decrece a cero rápidamente a medida que la dimensión aumenta, lo que no es tan intuitivo. Si se repite el ejercicio anterior para estimar el porcentaje de muestras que caen dentro de la esfera de radio 1.65 en distintas dimensiones, en la figura 2.13 se observa que a una dimensión igual a 10 este porcentaje es menor al 1 %, lo que implica que la mayor parte de las muestras se encuentran en los extremos de la función en vez de en el centro como ocurre en una dimensión, lo que podría significar que la cantidad de ejemplos disponibles tiene que ser mucho mayor para que una red pueda hacer buenas aproximaciones en alta dimensión. Silverman [51] dedujo que la cantidad de datos necesarios para aproximar una distribución gaussiana a través de kernels gaussianos es aproximadamente

$$\log_{10} N(d) \cong 0{,}6(d - 0{,}25) \tag{2.83}$$

lo que implicaría que la cantidad de datos necesarios aumenta en forma exponencial a medida que aumenta la dimensión.

En el mundo real no siempre se tiene una gran cantidad de datos como para cumplir con el requerimiento de Silverman, sin embargo como apunta Verleysen [81], los problemas del mundo real no sufren tan dramáticamente de la maldición de la dimensionalidad. La explicación para esto

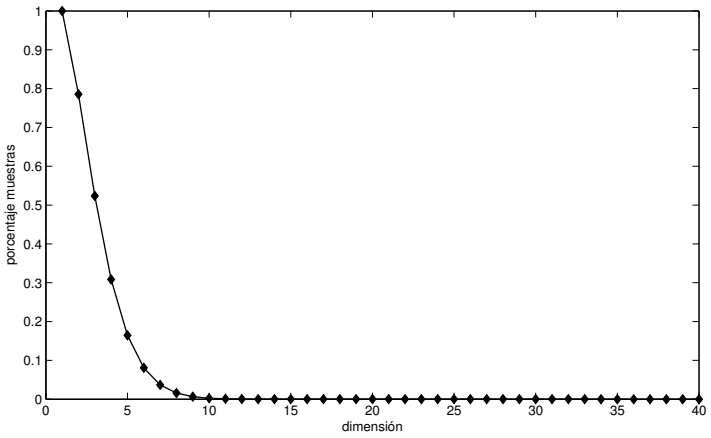


Figura 2.13: Porcentaje de muestras de una distribución gaussiana que caen dentro de una esfera de radio 1.65 versus la dimensión

es que en la mayoría de los casos los datos viven en un subespacio de dimensión mucho menor al espacio total, por lo que lo importante es tener la cantidad de datos suficiente para aproximar bien la función en ese subespacio y no en el espacio multidimensional original.

Capítulo 3

Metodología: Proyección Utilizando Teoría de la Información

3.1. Introducción

Después de haber revisado los diferentes modelos de máquinas de aprendizaje y algunos de los criterios de optimización existentes, en este capítulo se describe como se utiliza la Teoría de la Información para desarrollar un algoritmo que sea capaz de hacer una proyección de datos multidimensionales a una dimensión menor.

Primero se describe un modelo de proyección basado en alguno de los modelos de aprendizaje ya presentado. El modelo planteado se optimiza utilizando un criterio de TI que permite generar una proyección óptima de los datos en términos de información. La elección de un modelo adecuado y de un criterio o funcional continuo y derivable, permiten la optimización numérica del sistema. Un esquema gráfico del modelo deseado se puede observar en la figura 3.1.

3.2. Modelo de Proyección Lineal

Sean $\{x_i\} \in \Re^D$, N muestras de la variable aleatoria X, se desea encontrar un mapeo $Y = g(X, W)$, tal que Y sea una transformación de las muestras de X a un nuevo conjunto $\{y_i\} \in \Re^d$ con $d < D$. Se dirá que la variable X pertenece al espacio de entrada y la variable Y al de salida.

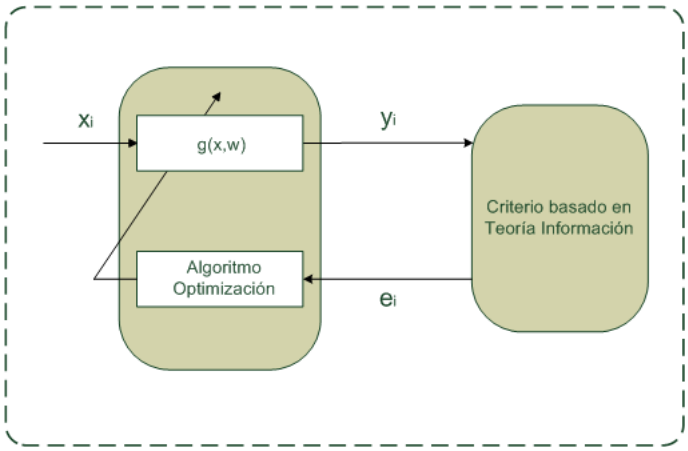


Figura 3.1: Esquema representando el mapeo o proyección

Esta transformación a un espacio de menor dimensión debe preservar cierta estructura de los datos de entrada. El tipo de estructura o información que se preserve dependerá fuertemente del mapeo $g(,)$ y del conjunto de parámetros W. Como se vio en la sección 2.2.7, existen variados modelos de mapeo que se pueden aplicar, cada uno con diferentes características. En este caso se ha escogido como modelo de proyección una transformación lineal entre el espacio de entrada y el de salida, es decir

$$Y = W^T X, \tag{3.1}$$

donde W es una matriz de $D \times d$ parámetros. La elección se baso en la fácil interpretación del modelo y la derivación de las reglas de optimización. Este tipo de transformación genera una combinación lineal de las características de los datos en el espacio de entrada para hacer la proyección. Cada una de las d columnas de la matriz W corresponde a un eje de proyección. Al ser lineal la transformación, no es posible extraer estructuras no lineales de los datos, pero por otro lado, la transformación tiene la propiedad de ser reversible y fácil de interpretar cuando se requiere una visualización del espacio de salida.

Ya que las columnas de la matriz W representan los ejes de proyección, es posible imponer la condición de ortonormalidad de W con el fin de que los ejes de proyección sean ortogonales y de dimensión unitaria, como se observa en la figura 3.2, lo que permite que la matriz de transformación sea una base en el sentido algebraico para el espacio $\Re^d$.

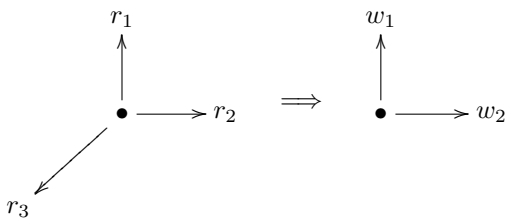


Figura 3.2: Ejemplo de Base Ortonormal. Cada eje es perpendicular a otro eje de la base.

3.3. Infomax e Información Mutua Cuadrática

Para encontrar los parámetros de la transformación W, se pueden utilizar los criterios descritos en 2.2.8, siendo el que mejor encaja con los objetivos planteados en esta tesis el criterio de Infomax. Esto porque se desea transmitir la mayor cantidad de información posible desde la entrada X a la salida Y de forma no supervisada, es decir, sin hacer uso de información externa como podría ser la clase a la que pertenecen los datos. Según el criterio de Infomax, para resolver el problema se requiere maximizar la IM entre X e Y sujeto al mapeo $Y = g(X, W)$.

Tal como se dijo en el capítulo anterior, utilizar el principio de Infomax para obtener una solución analítica es muy difícil, a menos que se asuman restricciones sobre la fdp como gaussianidad y un mapeo lineal [54], lo que restringe mucho su aplicabilidad en situaciones reales. Dado que los datos que se tienen son muestras finitas de las variables aleatorias, una solución apropiada es utilizar las medidas de entropía e información mutua cuadráticas como criterios a optimizar, ya que estos permiten hacer las estimaciones correspondientes directamente desde los datos, sin restricciones sobre las fdp. Como la ED-QMI (2.68) tiene un mejor comportamiento para la maximización de la IMC [17], ésta fue escogida para ser aplicada como estimación de la IM en el criterio Infomax en desmedro de la CS-QMI.

Formalmente el problema en términos del criterio Infomax se define como:

> *Sean* $\{x_i\} \in \Re^D$, N *muestras de la variable aleatoria* X, *sea* Y *una transformación lineal de las muestras de* X *a un nuevo conjunto* $\{y_i\} \in \Re^d$ *con* $d < D$, *de la forma* $Y = W^T X$. *Para encontrar el conjunto de parámetros definidos en la matriz de transformación* W *de modo que* Y *contenga la mayor cantidad de información posible sobre* X, *y que sus componentes sean ortogonales, es necesario maximizar* $I_{ED}(X, Y)$, *restringiendo la matriz* W *a ser ortonormal.*

El algoritmo propuesto, para resolver este problema se llama de ahora en adelante MDP-ITL por sus siglas en inglés (*Multidimensional Data Projection using Information Theoretic Learning*).

3.4. Algoritmo

Recordando la formulación de la I_{ED} como suma de potenciales:

$$V_{ED} = V_J - 2V_C + V_M, \tag{3.2}$$

donde

$$\begin{aligned} V_J &= \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) G\left(y_i - y_j, 2\sigma_y^2 I\right) \\ V_C &= \frac{1}{N}\sum_{i=1}^{N}\left\{\frac{1}{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \frac{1}{N}\sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right)\right\}, \\ V_M &= \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \end{aligned} \tag{3.3}$$

y

$$G\left(x_i - x_j, 2\sigma_x^2 I\right) = \frac{1}{\left(2\pi(2\sigma_x^2)\right)^{D/2}} \exp\left(-\frac{1}{2}\frac{\left(x_i - x_j\right)^2}{2\sigma_x^2}\right) \tag{3.4}$$

$$G\left(y_i - y_j, 2\sigma_y^2 I\right) = \frac{1}{\left(2\pi(2\sigma_y^2)\right)^{d/2}} \exp\left(-\frac{1}{2}\frac{\left(y_i - y_j\right)^2}{2\sigma_y^2}\right). \tag{3.5}$$

Para maximizar I_{ED} (V_{ED}) se puede utilizar cualquier método numérico para optimizar funciones continuas, como ascenso por gradiente, o métodos de segundo orden como Newton-Rhapson. En este caso utilizaremos un método de ascenso por gradiente ya que a pesar de ser más lento, es más simple de implementar y ajustar. La regla de actualización de los parámetros de la matriz W es la siguiente

$$W\left(t+1\right) = W\left(t\right) + \eta\frac{\partial V_{ED}}{\partial W}, \tag{3.6}$$

donde η representa la tasa de aprendizaje del sistema.

Utilizando la regla de la cadena, se puede reescribir la derivación del potencial con respecto a los parámetros como:

$$\frac{\partial V_{ED}}{\partial W} = \frac{\partial V_{ED}}{\partial\left(y_i - y_j\right)}\frac{\partial\left(y_i - y_j\right)}{\partial W} \tag{3.7}$$

además, utilizando (3.2),

$$\frac{\partial V_{ED}}{\partial W} = \frac{\partial V_J}{\partial W} - 2\frac{\partial V_C}{\partial W} + \frac{\partial V_M}{\partial W}. \tag{3.8}$$

De este modo la regla de actualización queda descrita de la siguiente forma

$$W(t+1) = W(t) + \eta \left\{ \frac{\partial V_J}{\partial W} - 2\frac{\partial V_C}{\partial W} + \frac{\partial V_M}{\partial W} \right\}. \tag{3.9}$$

Cabe notar que la derivada de V_{ED} debe ser calculada con respecto a $d_{ij} = (y_i - y_j)$, ya que el funcional V_{ED} depende de la interacción entre partículas y no de la posición específica de cada una de ellas. Por otro lado, la derivada $\frac{\partial(y_i - y_j)}{\partial W}$ depende del mapeo escogido, en este caso la transformación lineal $Y = W^T X$. De este modo, se puede llegar a la siguiente relación:

$$\frac{\partial (y_i - y_j)}{\partial W} = (x_i - x_j)^T. \tag{3.10}$$

Para desarrollar (3.9) por partes, se comienza por el potencial conjunto V_J:

$$\frac{\partial V_J}{\partial W} = \frac{\partial V_J}{\partial (y_i - y_j)} \frac{\partial (y_i - y_j)}{\partial W} \tag{3.11}$$

$$\begin{cases} \dfrac{\partial V_J}{\partial W} = \dfrac{1}{N^2} \displaystyle\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \frac{\partial G\left(y_i - y_j, 2\sigma_y^2 I\right)}{\partial (y_i - y_j)} \cdot \frac{\partial (y_i - y_j)}{\partial W} \\ \dfrac{\partial V_J}{\partial W} = \dfrac{1}{N^2} \displaystyle\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot G\left(y_i - y_j, 2\sigma_y^2 I\right) \left\{ -2 \cdot \frac{(y_i - y_j)}{2\sigma_y^2} \right\} \cdot (x_i - x_j)^T \end{cases} \tag{3.12}$$

Por lo tanto

$$\frac{\partial V_J}{\partial W} = -\frac{1}{N^2 \sigma_y^2} \sum_{i=1}^{N}\sum_{j=1}^{N} \left\{ G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot G\left(y_i - y_j, 2\sigma_y^2 I\right) (y_i - y_j) \cdot (x_i - x_j)^T \right\} \tag{3.13}$$

Para el potencial marginal V_M:

$$\frac{\partial V_M}{\partial W} = \frac{\partial V_M}{\partial (y_i - y_j)} \frac{\partial (y_i - y_j)}{\partial W} \tag{3.14}$$

$$\begin{cases} \dfrac{\partial V_M}{\partial W} = \dfrac{\partial}{\partial (y_i - y_j)} \left\{ \dfrac{1}{N^2} \displaystyle\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \tfrac{1}{N^2} \sum_{i=1}^{N}\sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \right\} \cdot \dfrac{\partial (y_i - y_j)}{\partial W} \\ \dfrac{\partial V_M}{\partial W} = \dfrac{1}{N^4} \displaystyle\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \sum_{i=1}^{N}\sum_{j=1}^{N} \dfrac{\partial}{\partial (y_i - y_j)} G\left(y_i - y_j, 2\sigma_y^2 I\right) \cdot \dfrac{\partial (y_i - y_j)}{\partial W} \\ \dfrac{\partial V_M}{\partial W} = \tfrac{1}{N^4} \displaystyle\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \sum_{i=1}^{N}\sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \cdot \left\{ -2 \cdot \dfrac{(y_i - y_j)}{2\sigma_y^2} \right\} (x_i - x_j)^T \end{cases} \tag{3.15}$$

$$\frac{\partial V_M}{\partial W} = -\frac{1}{N^4 \sigma_y^2} \sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \sum_{i=1}^{N}\sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \cdot (y_i - y_j)(x_i - x_j)^T \tag{3.16}$$

Finalmente se desarrolla el potencial de información cruzado:

$$\frac{\partial V_C}{\partial W} = \frac{\partial V_C}{\partial (y_i - y_j)} \frac{\partial (y_i - y_j)}{\partial W} \tag{3.17}$$

$$\begin{cases} \dfrac{\partial V_C}{\partial W} = \dfrac{\partial}{\partial (y_i - y_j)} \tfrac{1}{N} \displaystyle\sum_{i=1}^{N} \left\{ \tfrac{1}{N} \sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \tfrac{1}{N} \sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \right\} \cdot \dfrac{\partial (y_i - y_j)}{\partial W} \\ \dfrac{\partial V_C}{\partial W} = \tfrac{1}{N} \displaystyle\sum_{i=1}^{N} \left\{ \tfrac{1}{N} \sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \tfrac{1}{N} \dfrac{\partial}{\partial (y_i - y_j)} \sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \right\} \cdot \dfrac{\partial (y_i - y_j)}{\partial W} \\ \dfrac{\partial V_C}{\partial W} = \tfrac{1}{N^3} \displaystyle\sum_{i=1}^{N} \left\{ \sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) \left\{ -2 \cdot \dfrac{(y_i - y_j)}{2\sigma_y^2} \right\} (x_i - x_j)^T \right\} \end{cases} \tag{3.18}$$

$$\frac{\partial V_C}{\partial W} = -\frac{1}{N^3 \sigma_y^2} \sum_{i=1}^{N} \left\{ \sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) \cdot \sum_{j=1}^{N} G\left(y_i - y_j, 2\sigma_y^2 I\right) (y_i - y_j)(x_i - x_j)^T \right\} \tag{3.19}$$

Como resultado se tiene que el algoritmo de optimización es de orden cuadrático, ya que depende de las interacciones entre pares de datos. Por otro lado, los únicos parámetros a ajustar son la tasa de aprendizaje η y el tamaño del kernel en los espacios de entrada y salida σ_x y σ_y.

3.4.1. Determinación de Parámetros

La determinación del tamaño del kernel es un paso importante para obtener buenos resultados, ya que tal como se describió en 2.2.3.2, este determina que tan ajustada es la estimación de Parzen.

Un kernel mal ajustado puede llevar a soluciones sub-óptimas o degeneradas. Para la determinación del tamaño de los kernels lo más recomendable es utilizar la regla de Silverman (2.57) como primera aproximación. Posteriormente se puede hacer un ajuste más fino de acuerdo al resultado de los experimentos. La ventaja de esta regla es que se aplica directamente sobre el conjunto de datos y no requiere mayores cálculos complejos.

La determinación de la tasa de aprendizaje η es más bien heurística, y requiere generalmente hacer varios experimentos de prueba para encontrar un valor adecuado. Lo importante es encontrar un valor que permita una convergencia suave hacia el óptimo sin que esta sea muy lenta.

3.4.2. Annealing

Para mejorar la convergencia del algoritmo se puede utilizar un proceso llamado *annealing* (recocido) para el tamaño del kernel, el cual se basa en el proceso del mismo nombre utilizado en la industria metalúrgica para cambiar propiedades como la resistencia o ductilidad de ciertos materiales. Este consiste en llevar la temperatura del material a un nivel por sobre la temperatura de re-cristalización del mismo y luego permitir que se enfríe lentamente. En este caso el *annealing* consiste en variar el tamaño del kernel desde un valor por sobre el óptimo hasta otro valor menor por debajo del óptimo. La idea detrás de este proceso es que en las primeras etapas de la convergencia el kernel sea mayor lo que permite obtener estimaciones suavizadas de la superficie de optimización evitando caer en mínimos locales. Como el tamaño del kernel se va empequeñeciendo hacia el final, esto permite obtener estimaciones más precisas cerca del óptimo. Sea σ^+ el tamaño del kernel al inicio del proceso y σ^- el valor final, formalmente el tamaño del kernel en cada iteración i, σ_i, se calcula como:

$$\sigma_i = \sigma^+ \cdot \left(\frac{\sigma^-}{\sigma^+}\right)^{\frac{i}{i_{max}}} . \tag{3.20}$$

La elección de los valores para σ^+ y σ^- no son fijos pero generalmente son cercanos a 1.5 y 0.5 veces el "óptimo de Silverman" (2.57) respectivamente. La tasa de aprendizaje por otro lado también puede ser tratada con el mismo proceso al variarla desde un valor máximo η^+ a un valor mínimo η^- siguiendo el mismo esquema que el tamaño del kernel. La idea de mover la tasa de aprendizaje es favorecer una búsqueda hacia el óptimo más rápida y más gruesa en un comienzo para terminar con una búsqueda más fina y local. La figura 3.3 muestra la forma en que decrece el valor de sigma a lo largo de la optimización.

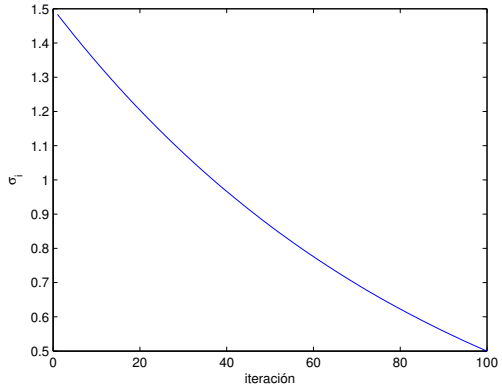


Figura 3.3: Gráfico *annealing* de σ desde un valor de 1,5 a 0,5

3.5. Preprocesamiento

Una base de datos cualquiera por lo general requiere de algún pre procesamiento antes de ser utilizada, ya sea para eliminar algún tipo de ruido o para evitar problemas derivados del escalamiento de las distintas variables. Este paso es fundamental para el funcionamiento de algunos métodos, mientras que para otros sirve como una simplificación que ayuda a obtener mejores resultados.

3.5.1. Centrado

Centrado es una forma de procesamiento simple y necesaria para algunos métodos como PCA. El centrado consiste en restar la media de cada componente tal que la nueva media sea cero. Este es un paso fundamental para el funcionamiento de PCA y una simplificación para ICA ya que se elimina la necesidad de estimar la media.

$$\tilde{x} = x - E\{x\}. \tag{3.21}$$

3.5.2. Escalamiento

Muchas veces las bases de datos están hechas en base a mediciones de distinto tipo tomadas en ambientes reales, por lo que cada una de estas variables puede estar en una escala muy diferente dependiendo de su naturaleza. Por ejemplo, una variable puede indicar la temperatura de un proceso y otra la presión, encontrándose ambas medidas en escalas muy diferentes, lo que puede traer ciertas complicaciones a algunos métodos, sobre todo cuando se requiere ajustar parámetros. En este caso,

una forma simple de evitar problemas es escalar cada variable a un intervalo acotado, generalmente [0, 1] ó [-1, 1].

3.5.3. Blanqueo

El proceso de blanqueo consiste en transformar los datos linealmente de modo que los vectores de características de los datos transformados ($\tilde{x}$) no esten correlacionados entre ellos y tengan varianzas unitarias. Es decir, que la matriz de covarianzas de los datos transformados sea igual a la matriz identidad [82],

$$E\{\tilde{x}\tilde{x}^T\} = I. \tag{3.22}$$

En los métodos clásicos basados en kernels, se utiliza el mismo valor de ancho de banda σ para todas las características, por lo que un proceso de blanqueo puede ayudar a que la estimación del tamaño del kernel sea más precisa y óptima. Una forma simple de hacer el blanqueo es utilizar la matriz de covarianzas de los datos originales Σ y hacer la siguiente transformación [83]:

$$\tilde{x_i} = \Sigma^{-1/2}\left(x_i - \bar{x}\right). \tag{3.23}$$

El resultado es un conjunto de datos centrado y con varianzas unitarias. La matriz de covarianzas puede ser descompuesta de la forma $\Sigma = UDU^T$, conocida como descomposición EVD (*eigenvalue decomposition*), donde U es una matriz ortonormal y D una matriz diagonal formada por los valores propios de Σ, con lo que la transformación puede ser reescrita como

$$\tilde{x_i} = UD^{-1/2}U^T\left(x_i - \bar{x}\right). \tag{3.24}$$

En el caso de MDP-ITL se propone utilizar el blanqueo como un paso necesario para un mejor funcionamiento del método de proyección, la idea es que el blanqueo elimine ruido existente y además elimine la dependencia de los métodos sobre la varianza de los datos, permitiendo un mejor desempeño de kernels simétricos como los utilizados.

Capítulo 4

Resultados de Simulaciones

En el presente capítulo se entregan los detalles sobre las pruebas realizadas para probar la efectividad del algoritmo propuesto. Se describen las bases de datos utilizadas con sus principales características y procedencia. Se describen los experimentos detalladamente, incluyendo los parámetros utilizados. Finalmente se entregan los resultados de la aplicación del algoritmo sobre las bases de datos seleccionadas y las comparaciones con otros algoritmos de proyección.

4.1. Bases de Datos

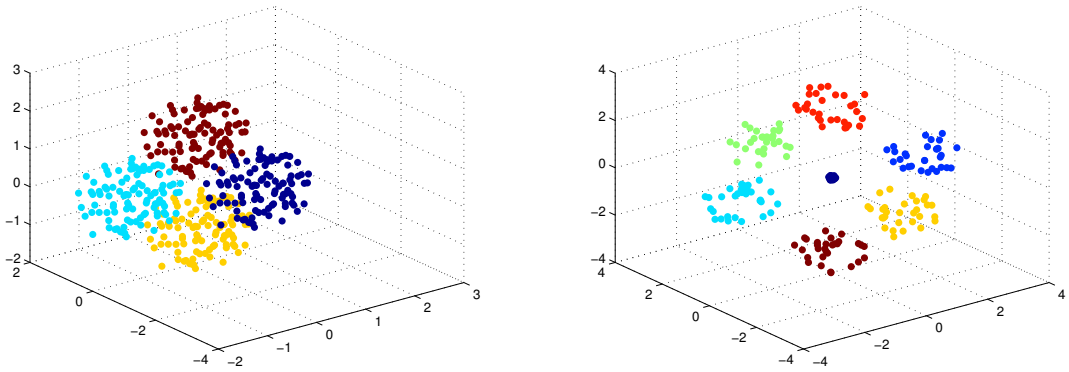
La elección de las bases de datos utilizadas para probar el funcionamiento del algoritmo propuesto se hizo en base a distintos criterios. Primero se escogieron bases artificiales simples cuyas estructuras son conocidas, de forma de tener la certeza de que el algoritmo funciona apropiadamente. En segundo lugar se escogió una base de datos sintetica pero de mayor complejidad. Finalmente se escogió una base real pero ampliamente usada en la literatura. A continuación la descripción de cada una de ellas.

Tetra: Conjunto que contiene información sobre cuatro esferas tridimensionales, cada una formada por 100 elementos y con la misma distribución. Este conjunto forma parte del *Fundamental Clustering Problem Suite* [1], conjunto de problemas artificialmente creados para testear el funcionamiento de métodos de clustering. En la figura 4.1 se puede apreciar la base en su espacio original. La información de la clase se utiliza como referencia para colorear el mapa.

[1] http://www.uni-marburg.de/fb12/datenbionik/

Hepta: Conjunto de 7 *clusters* tridimensionales con 178 datos en total, parte del *Fundamental Clustering Problem Suite* (figura 4.1).

Pipeline: Esta base de datos modela ciertas mediciones hechas sobre una tubería que transporta una mezcla de agua, aceite y gasolina. En total se tienen mil datos para el conjunto de entrenamiento que es el usado para la proyección. La mezcla de fluidos puede ir en tres configuraciones dependiendo de la proporción de cada elemento. Si bien cada medición posee 13 características, existen sólo dos grados de libertad para la configuración, la fracción de agua y aceite pues el tercero dependerá de la suma de los dos primeros. De acuerdo a esto se podría decir que los datos viven en un espacio localmente bi-dimensional. Estos datos se pueden obtener desde el sitio web del *Neural Computing Research Group* [2].



(a) Tetra (b) Hepta

Figura 4.1: Bases Artificiales tomadas de FCPS

Iris: Este conjunto de datos contiene información sobre tres variedades de la flor llamada *Iris,* conocidas como Iris Setosa, Iris Versicolor e Iris Virginica, donde cada elemento ha sido descrito utilizando cuatro características: largo y ancho del cépalo, y largo y ancho del pétalo. De cada clase se tienen 50 datos donde una clase es linealmente separable de las otras dos,

[2] http://www.ncrg.aston.ac.uk/GTM/3PhaseData.html

mientras estas últimas no se pueden separar linealmente entre ellas. Este conjunto de datos fue extraído del *UCI Machine Learning Repository* [3].

4.2. Métodos a comparar

Entre los métodos descritos en 2.1.1 se escogieron los siguientes para fines de comparación con el método propuesto :

PCA: es el método de proyección lineal más conocido y utilizado, por lo que la comparación con este método es necesaria.

NLM: por ser un método clásico de proyección, es interesante compararse con él, a pesar de su naturaleza no-lineal que en principio podría ser una ventaja sobre los métodos lineales.

SNE: compararse con SNE es conveniente ya que este posee una naturaleza probabilística y utiliza como función de costos una medida de divergencia, específicamente una suma de divergencias KL, mientras que el método propuesto en esta tesis utiliza una medida de divergencia cuadrática para estimar la IM.

4.3. Experimentos

Los experimentos consistieron en ejecutar el algoritmo en las bases seleccionadas para posteriormente medir la calidad de la proyección utilizando las medidas de calidad vistas en la sección 2.1.2. El primer paso, sin embargo, fue establecer los parámetros a utilizar en cada simulación. La lista de parámetros a definir en cada experimento para MDP-ITL es la descrita en Tabla 4.1.

Tabla 4.1: Lista de parámetros a definir en cada experimento

Parámetro	**Descripción**
i_{max}	número máximo de iteraciones a ejecutar el algoritmo
σ^+	tamaño del kernel en la primera iteración
σ^-	tamaño del kernel en la última iteración
η^+	tasa de aprendizaje en la primera iteración
η^-	tasa de aprendizaje en la última iteración

[3] http://archive.ics.uci.edu/ml/index.html

La forma de definir los parámetros se basó en diferentes criterios tal como se detalla a continuación:

σ^+, σ^- fueron escogidos dependiendo del σ_{AMISE} calculado con (2.57). Empíricamente se estableció que un buen rango de valores para σ va entre 1,5 hasta 0,5 veces el óptimo, por lo que se establecieron estos valores por defecto. Para cada prueba sin embargo se hicieron múltiples simulaciones con el fin de ajustar los valores de la forma más óptima posible a cada base de datos.

i_{max} se escogió por defecto un valor de cien iteraciones para la mayoría de las pruebas ya que empíricamente se observó que era una cantidad suficiente de iteraciones para que el método convergiera en la mayoría de las pruebas. En las bases donde se requería un número mayor de iteraciones se ajustó en base a múltiples pruebas experimentales.

η^+, η^- habiendo establecido los valores anteriores se escogieron las tasas de aprendizaje de forma de obtener una convergencia suave hacia el óptimo en el número de iteraciones indicada. La diferencia entre η^+ y η^- se mantuvo en un orden de magnitud para todas las pruebas.

Además es importante recordar que el blanqueo se incluye como parte del método MDP-ITL, es decir, todas las simulaciones consisten en utilizar la etapa de blanqueo como pre-procesamiento, lo que no implica que no se pueda añadir algún otro tipo de pre-procesamiento adicional.

Para el resto de los métodos, los parámetros a ajustar se describen en la tabla 4.2.

Tabla 4.2: Lista Parámetros otros métodos

Método	**Parámetro**
PCA	dimensión salida
NLM	dimensión salida número iteraciones
SNE	dimensión salida número iteraciones perplejidad $\sigma_{mínimo}$ $\sigma_{máximo}$

Para determinar los valores de los parámetros se realizaron pruebas variando el valor del parámetro deseado dentro del intervalo predefinido en la tabla 4.2, con lo que se genera una familia

de curvas para cada medida de calidad. La configuración que entrega los mejores indicadores es la escogida.

Los valores para k-NN y el índice de Dunn corresponden al promedio conseguido sobre 20 simulaciones. Para el caso de k-NN se utilizó un $k = 5$ y en cada simulación se utilizó un 30 % de la base proyectada como entrenamiento y un 70 % para evaluar la clasificación.

Con el objeto de mostrar los resultados de la forma más clara posible, los mapas de las proyecciones incluirán información de las clases en forma de colores. Esto no implica que la información de las clases se utilice como parte de los métodos si no que se utilizan como referencia para una mejor observación.

4.3.1. Tetra

La tabla 4.3 muestra un resumen de las características de la base de datos.

Tabla 4.3: Características base Tetra

Característica	**Valor**
Dimensión	3
N° datos	400
Clases	4

Tetra - Prueba 1

La primera prueba consiste en proyectar la base de datos sin ningún pre-procesamiento salvo el blanqueo en el caso del método MDP-ITL. La lista de parámetros utilizados para MDP-ITL se muestra en la tabla 4.4. Para los otros métodos (PCA, SNE y NLM) se detallan los parámetros en la tabla 4.5.

Tabla 4.4: Lista Parámetros MDPITL. Prueba 1.

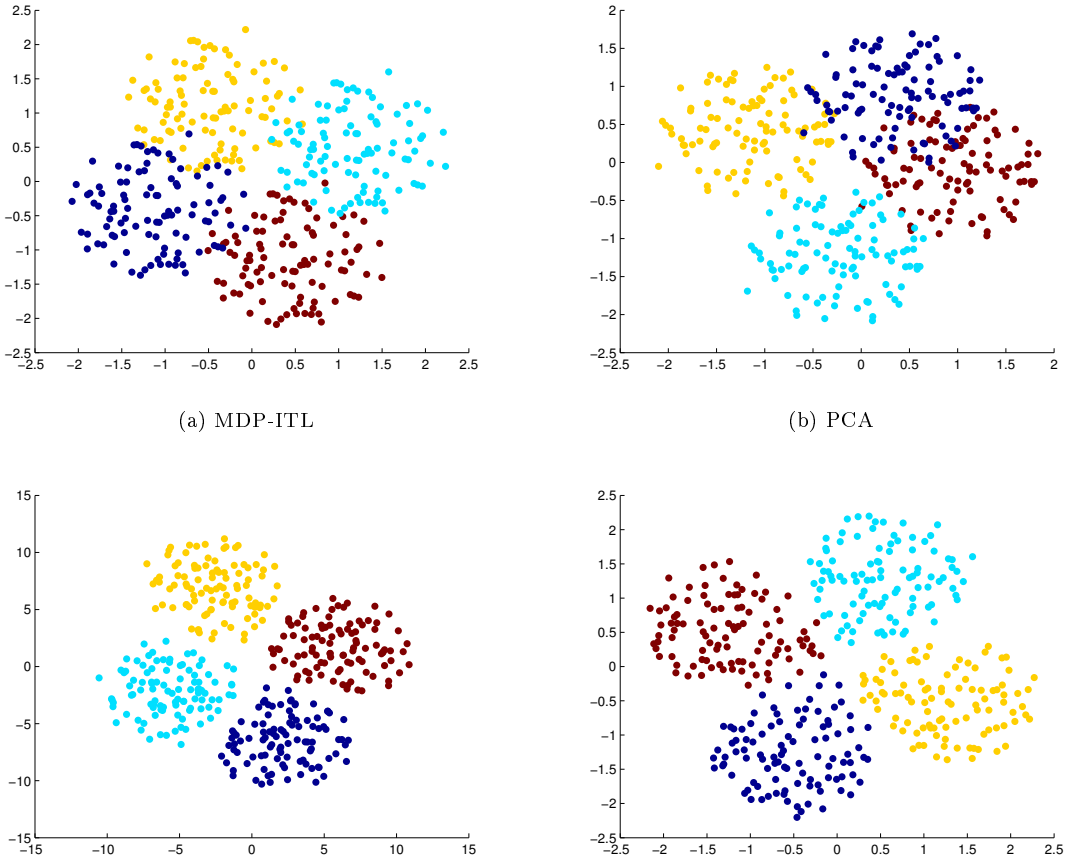
Parámetro	**Valor**
i_{max}	200
σ^+	1.5
σ^-	0.5
η^+	1e6
η^-	1e5

Tabla 4.5: Lista Parámetros proyección base Tetra. Prueba 1.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	100
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	10
	$\sigma_{mínimo}$	0.21
	$\sigma_{máximo}$	0.39

Los valores para la tasa de aprendizaje η son de varios ordenes de magnitud debido a que los valores obtenidos del gradiente son muy pequeños, por lo que se necesita un valor alto para que el algoritmo converja.

Los resultados de la proyección para los cuatro métodos utilizados se muestran en la figura 4.2. Se puede apreciar que los métodos no lineales hacen una buena proyección separando cada una de las clases. Por otro lado PCA parece traslapar en un porcentaje no menor 2 clases. Mientras tanto MDP-ITL logra mantener la estructura simétrica de los datos, manteniendo la poca distancia entre las clases que se tiene en el espacio de entrada.



(a) MDP-ITL

(b) PCA

(c) SNE

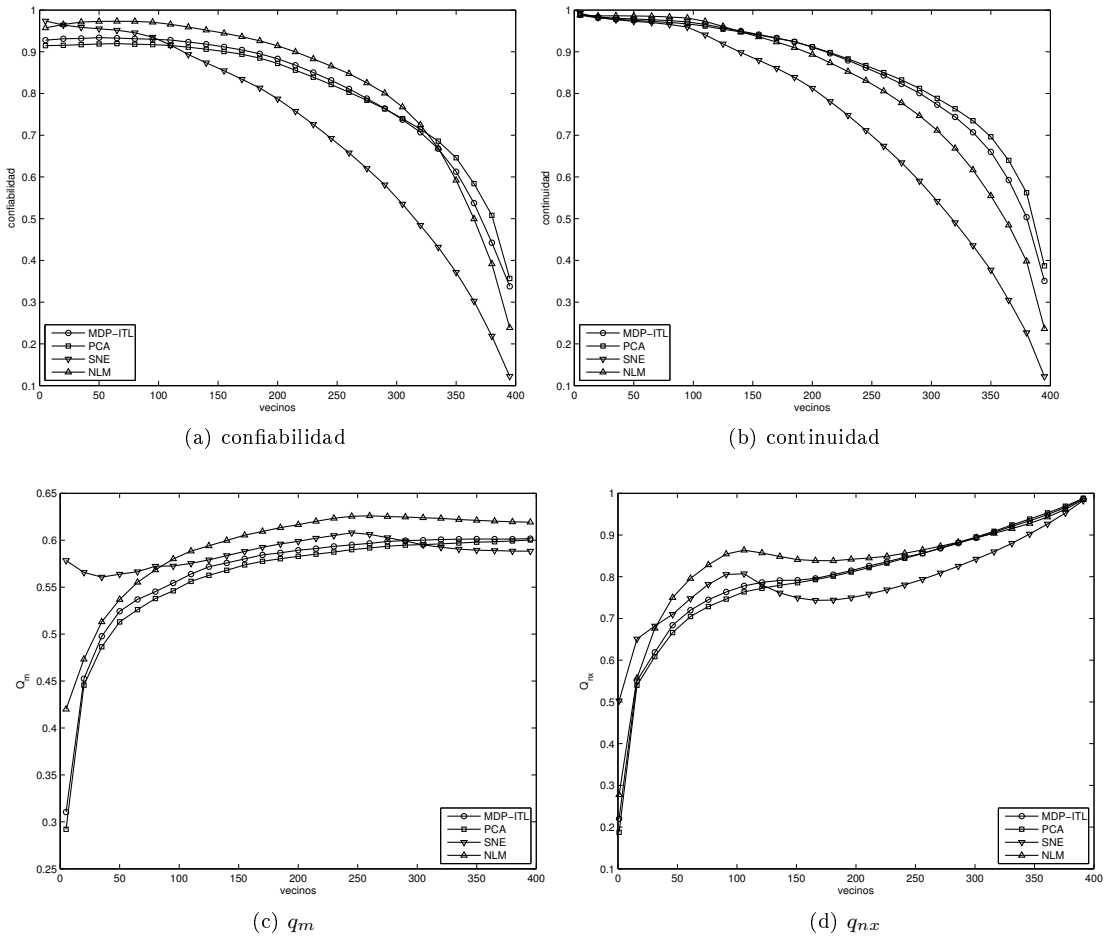
(d) NLM

Figura 4.2: Proyecciones base Tetra. Prueba 1.

El resultado de computar las medidas de calidad sobre las proyecciones se puede apreciar en la figura 4.3, donde SNE y NLM obtienen los mejores resultados en las vecindades "locales" (bajo numero de vecinos, se toman en cuenta sólo los más cercanos a cada dato). PCA obtiene resultados más bajos en terminos generales lo que coincide con la baja calidad de su proyección (efecto del traslape).

Tabla 4.6: Índice Dunn y k-NN, base Tetra. Prueba 1.

	MDP-ITL	PCA	SNE	NLM
dunn	0.023	0.022	0.157	0.076
k-NN (%)	91.4	89.7	100	99.2



(a) confiabilidad (b) continuidad

(c) q_m (d) q_{nx}

Figura 4.3: Medidas de Calidad base Tetra. Prueba 1.

Los métodos no lineales logran mejores resultados en las medidas de desempeño basadas en topología, en las zonas que pueden ser llamadas "locales" (bajo número de vecinos, debido a que son las que toman en cuenta los vecinos más próximos). Sobre todo se destaca el valor de la medida q_m para SNE. Al aumentar el tamaño de la vecindad los métodos no lineales bajan su desempeño sobre todo en confiabilidad y continuidad. En el caso de SNE esto se podría explicar debido a su naturaleza de preservación local, lo que se ve maximizado por el hecho de escoger una perplejidad baja. NLM sigue un comportamiento similar por lo que se podría deducir que el efecto viene también dado por la nolinealidad de la proyección. Los métodos lineales por su parte presentan un comportamiento muy similar con ventaja de MDP-ITL sobre PCA.

Los resultados para el índice de dunn y la clasificación usando k-NN publicados en 4.6 son

concordantes con las otras medidas, mostrando una superioridad de los métodos no lineales y una ventaja de MDP-ITL sobre PCA.

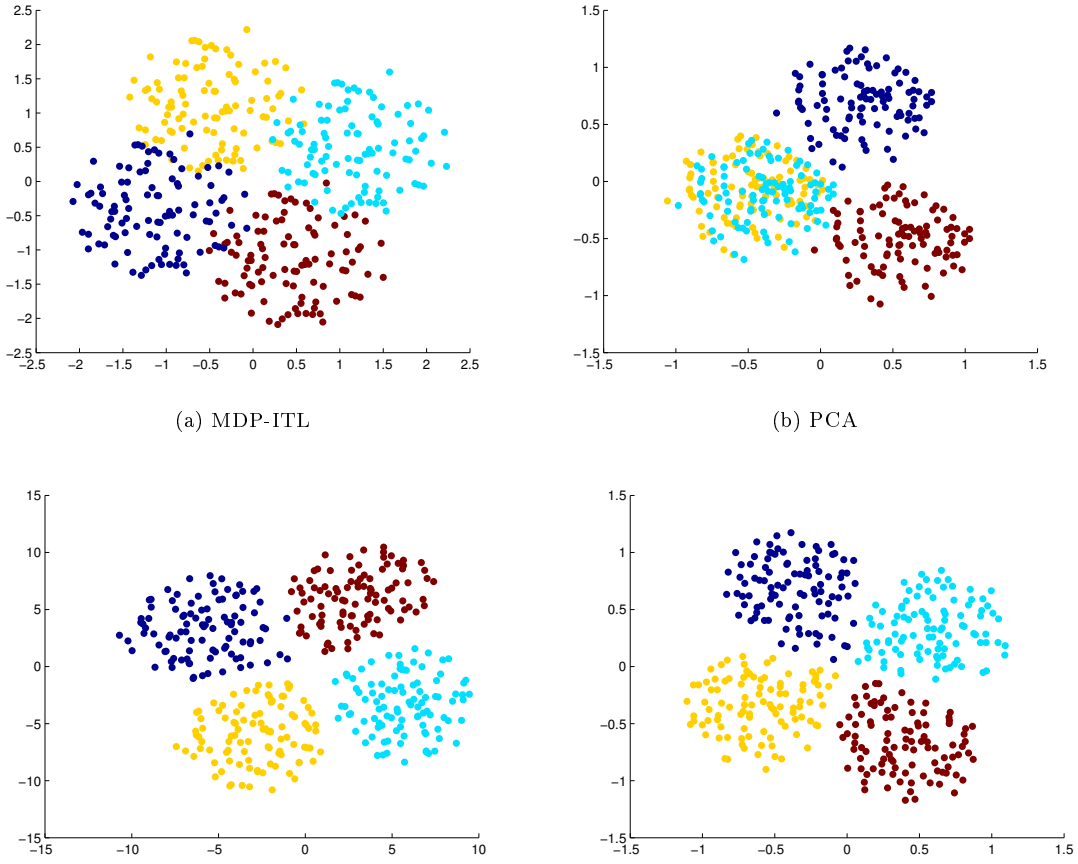
Tetra - Prueba 2

En este caso las bases son escaladas por cada característica en el intervalo $[-1, 1]$ antes de ser proyectadas. La lista de parámetros utilizadas es idéntica a las mostradas en la tabla 4.4 en el caso de MDP-ITL, incluyendo el blanqueo. Para los otros métodos los parámetros se indican en la tabla 4.7.

Tabla 4.7: Lista Parámetros proyección base Tetra. Prueba 2.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	100
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	10
	$\sigma_{mínimo}$	0.11
	$\sigma_{máximo}$	0.21

Los resultados a esta prueba muestran un comportamiento similar al caso anterior para todos los métodos excepto PCA, que traslapa completamente dos clases, como puede ser visto en la figura 4.4. En la figura 4.5 se observa además como PCA baja su desempeño sobre todo en lo que respecta a confiabilidad y q_m. El resultado para la clasificación con k-NN cae drásticamente para PCA como se ve en 4.8. MDP-ITL no cambia su desempeño comparado con el caso anterior.



(a) MDP-ITL

(b) PCA

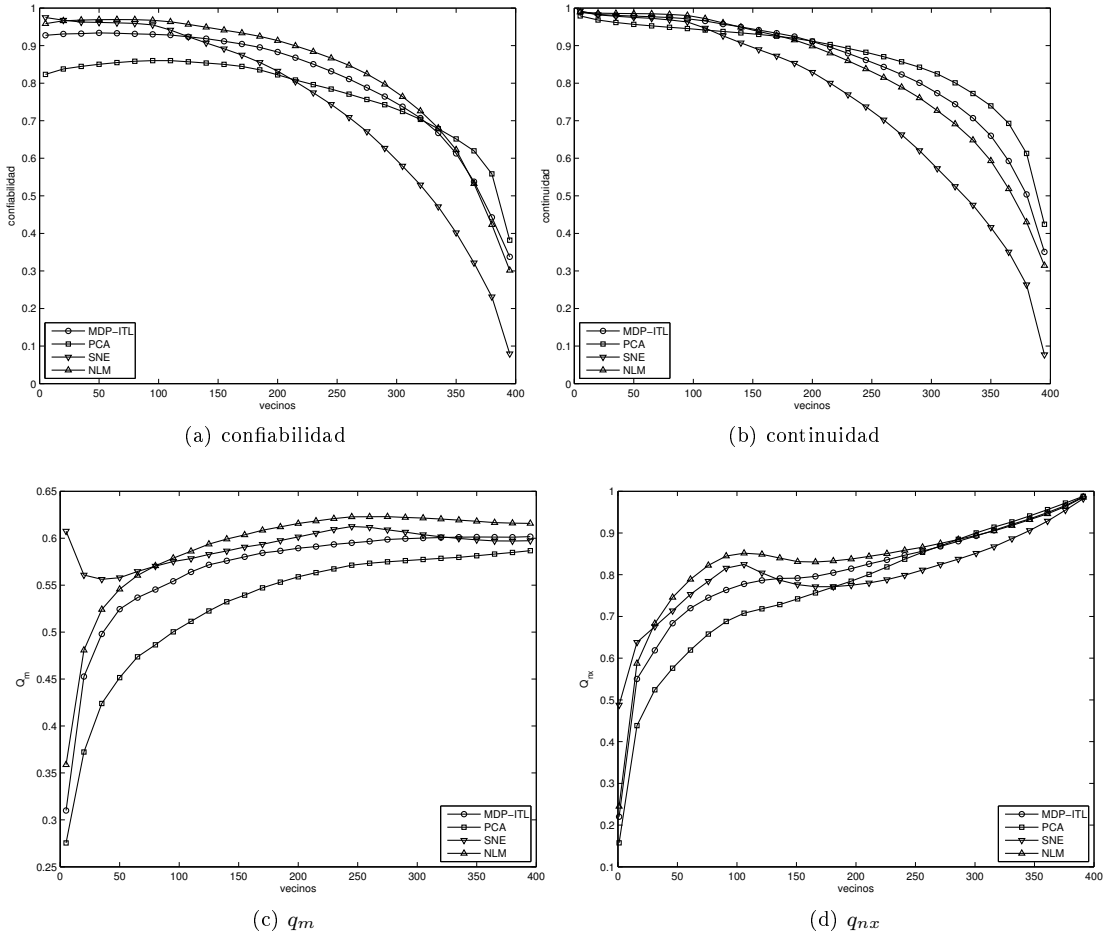
(c) SNE

(d) NLM

Figura 4.4: Proyecciones Tetra, base escalada. Prueba 2.

Tabla 4.8: Índice Dunn y k-NN, base Tetra. Prueba 2

	MDP-ITL	PCA	SNE	NLM
dunn	0.023	0.024	0.136	0.085
k-NN (%)	91.1	71	100	99.3



(a) confiabilidad (b) continuidad

(c) q_m (d) q_{nx}

Figura 4.5: Medidas de Calidad Tetra, base escalada. Prueba 2.

4.3.2. Hepta

El resumen de las características de la base de datos se muestra en la tabla 4.9.

Tabla 4.9: Características base Hepta

Característica	**Valor**
Dimensión	3
N° datos	212
Clases	7

Hepta - Prueba 1

Para la primera prueba se proyectan las bases de datos sin preprocesamiento adicional utilizando los parámetros mostrados en las tablas 4.10 y 4.11.

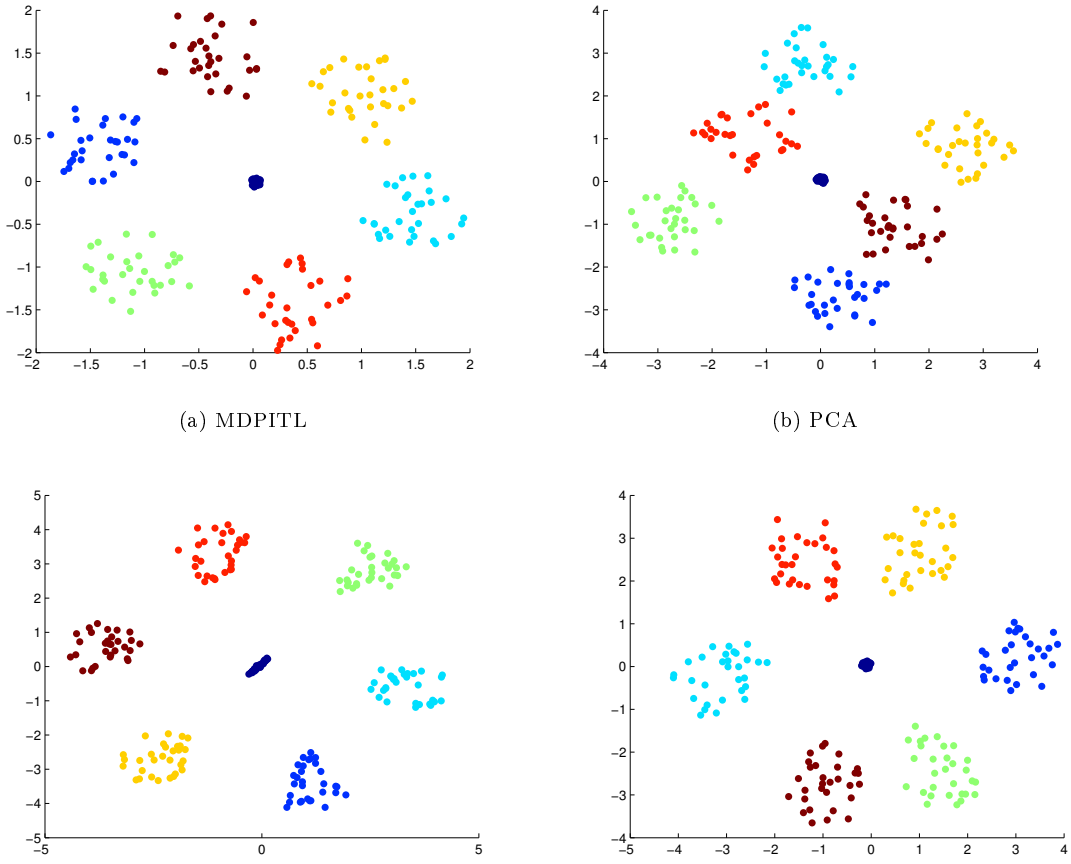
Tabla 4.10: Lista Parámetros MDPITL. Prueba 1.

Parámetro	**Valor**
i_{max}	120
σ^+	1.5
σ^-	0.5
η^+	1e5
η^-	1e4

Tabla 4.11: Lista Parámetros proyección base Hepta. Prueba 1.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	100
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	30
	$\sigma_{mínimo}$	0.1
	$\sigma_{máximo}$	1.7

Los resultados de la proyección sobre esta base, como se ven en la figura 4.6, muestran una proyección muy similar para todos los métodos. PCA sin embargo pierde parte de la simetría de la base obteniendo una proyección más achatada. Las medidas de confiabilidad y continuidad (figura 4.7) muestran un alto valor para todos los métodos en la zona más local, sin embargo SNE cae rápidamente al aumentar la vecindad. MDPITL supera el desempeño de PCA en la mayor parte del espacio para todas las medidas y es la que presenta una mayor continuidad en todo el espacio. Los métodos no lineales presentan valores para q_m mucho mayores a los métodos lineales. La medida q_{nx} es parecida para todos los métodos en las zonas más locales, SNE cae a medida que se aumenta el número de vecinos ($\gtrsim$ 75). Los resultados de la tabla 4.12 son muy similares para todas las proyecciones en terminos de clasificación. El índice de dunn muestra una superioridad en SNE.



(a) MDPITL

(b) PCA

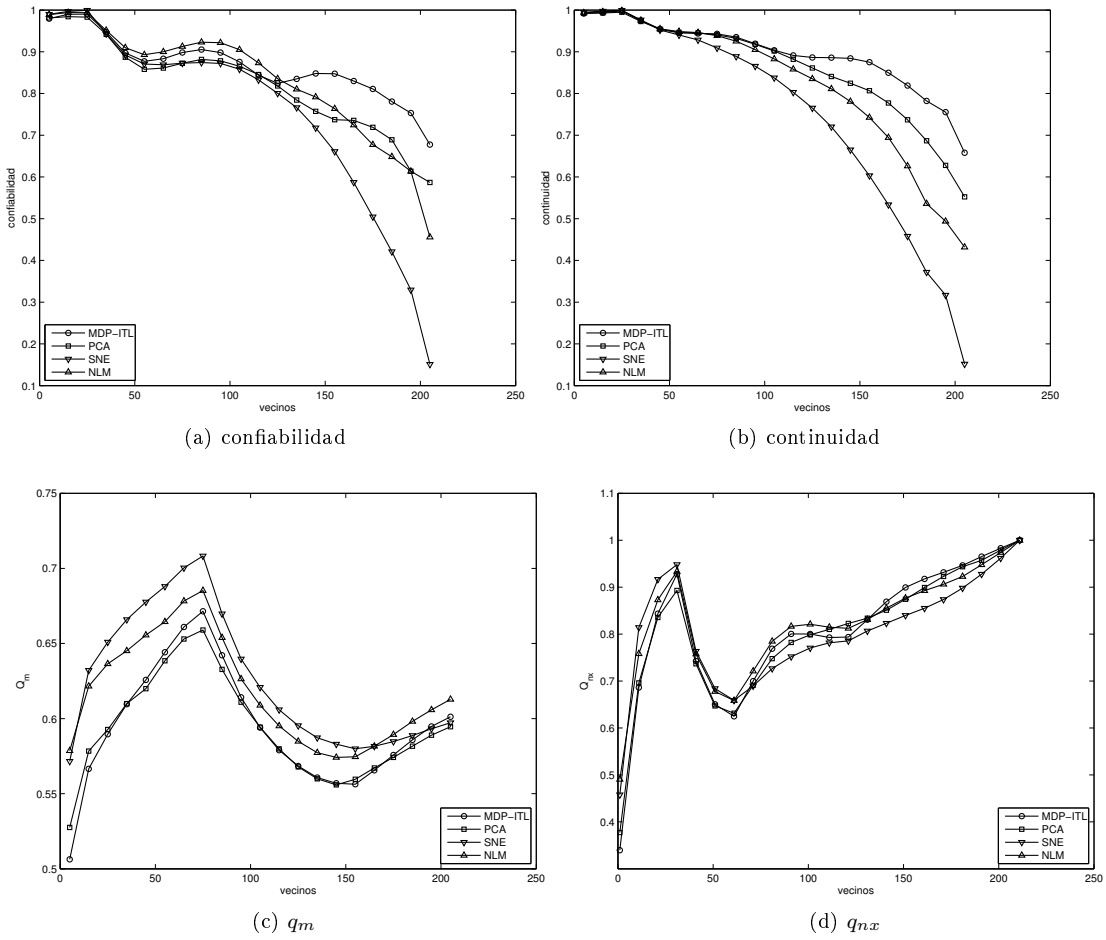
(c) SNE

(d) NLM

Figura 4.6: Proyecciones base Hepta. Prueba 1.

Tabla 4.12: Índice Dunn y k-NN, base Helpa. Prueba 1

	MDP-ITL	PCA	SNE	NLM
dunn	0.217	0.06	0.693	0.177
k-NN (%)	100	99.8	100	100



(a) confiabilidad (b) continuidad

(c) q_m (d) q_{nx}

Figura 4.7: Medidas de Calidad base Hepta. Prueba 1.

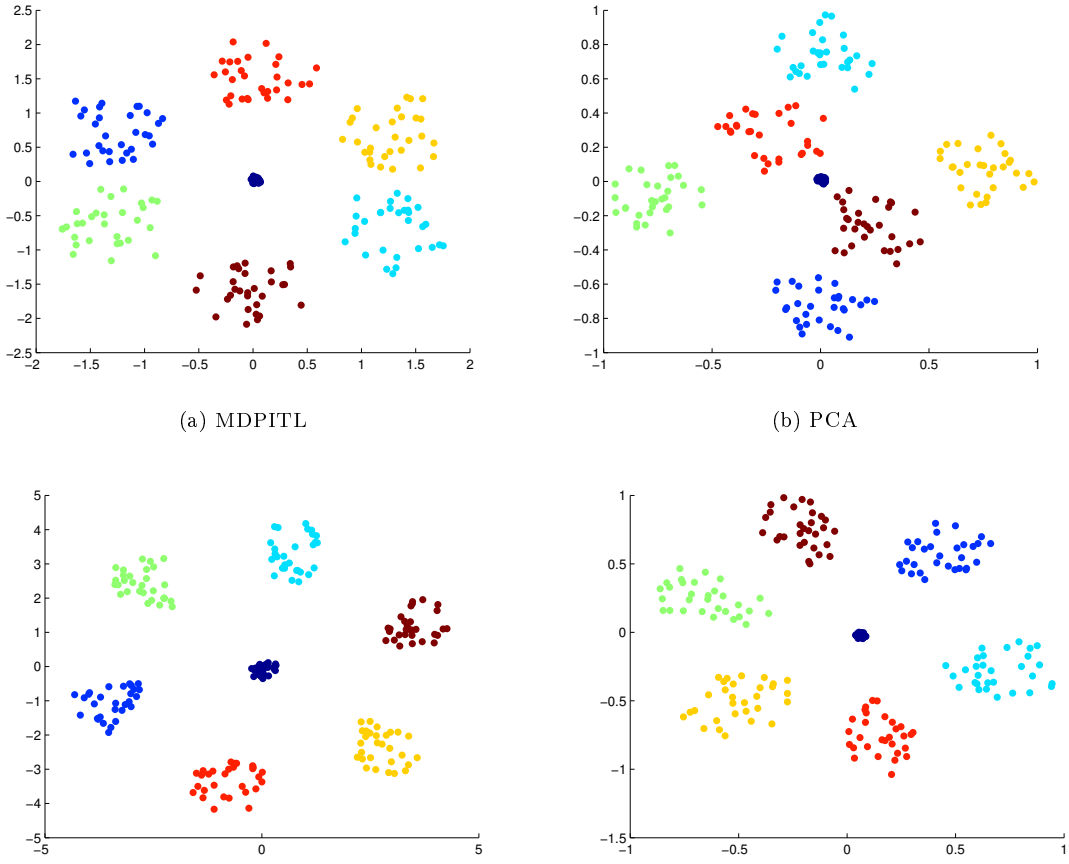
Hepta - Prueba 2

La base de datos se escala en cada una de sus componentes en el intervalo $[-1, 1]$ antes de ser procesada por los algoritmos de proyección. La lista de parámetros utilizados se describen en las tablas 4.10 y 4.13.

Tabla 4.13: Lista Parámetros proyección base Hepta. Prueba 2.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	100
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	30
	$\sigma_{mínimo}$	0.03
	$\sigma_{máximo}$	0.44

Los resultados de las proyecciones se muestran en la figura 4.8 mientras los gráficos con las medidas de calidad se encuentran en la figura 4.9. En ambas figuras no se observan grandes diferencias con respecto al caso no escalado, por lo que se deduce que esta base no se ve afectada por el proceso de escalamiento.



(a) MDPITL

(b) PCA

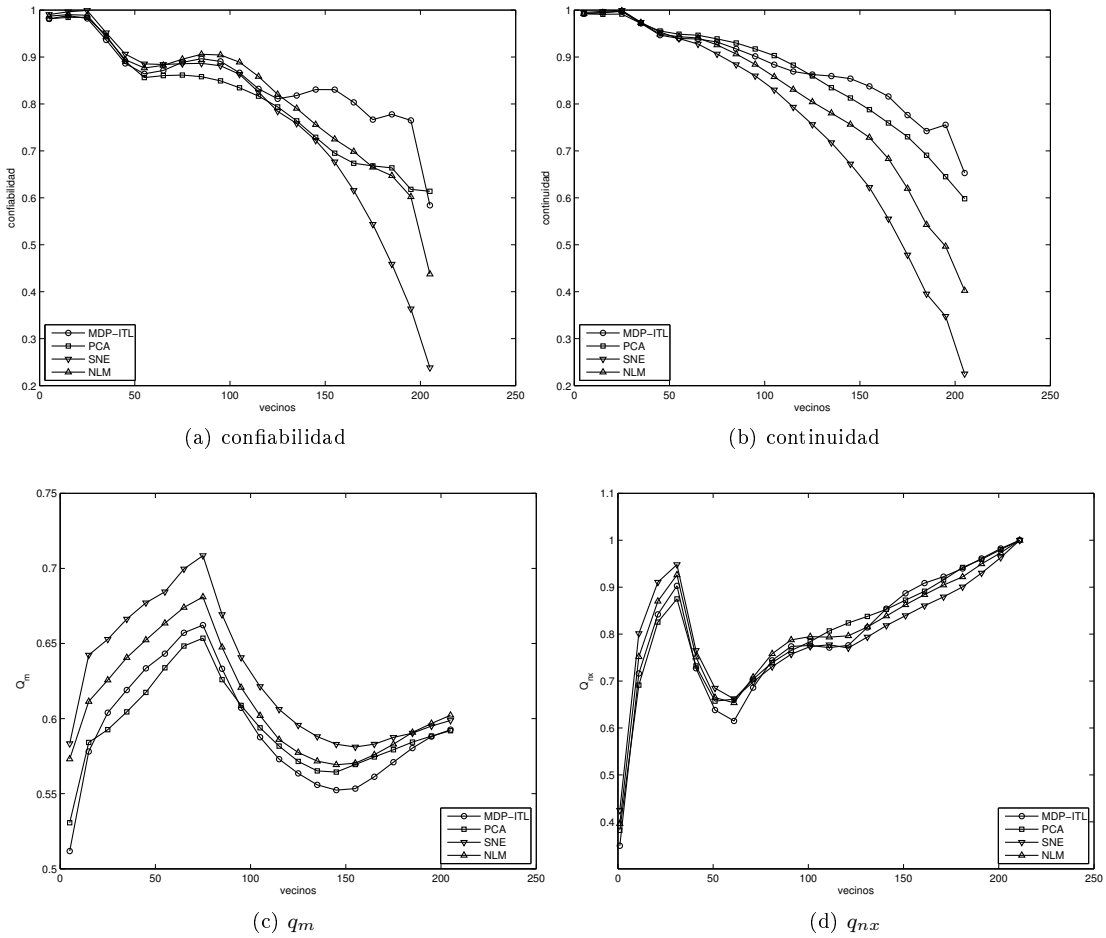
(c) SNE

(d) NLM

Figura 4.8: Proyecciones Hepta, base escalada. Prueba 2.

Tabla 4.14: Índice Dunn y k-NN, base Hepta. Prueba 2

	MDP-ITL	PCA	SNE	NLM
dunn	0.166	0.061	0.369	0.128
k-NN (%)	100	100	100	100



(a) confiabilidad (b) continuidad

(c) q_m (d) q_{nx}

Figura 4.9: Medidas de Calidad base Hepta escalada. Prueba 2.

4.3.3. Pipeline

El resumen de las características de la base de datos se muestra en la tabla 4.15.

Tabla 4.15: Características base *Pipeline*

Característica	**Valor**
Dimensión	13
N° datos	1000
Clases	3

Pipeline - Prueba 1

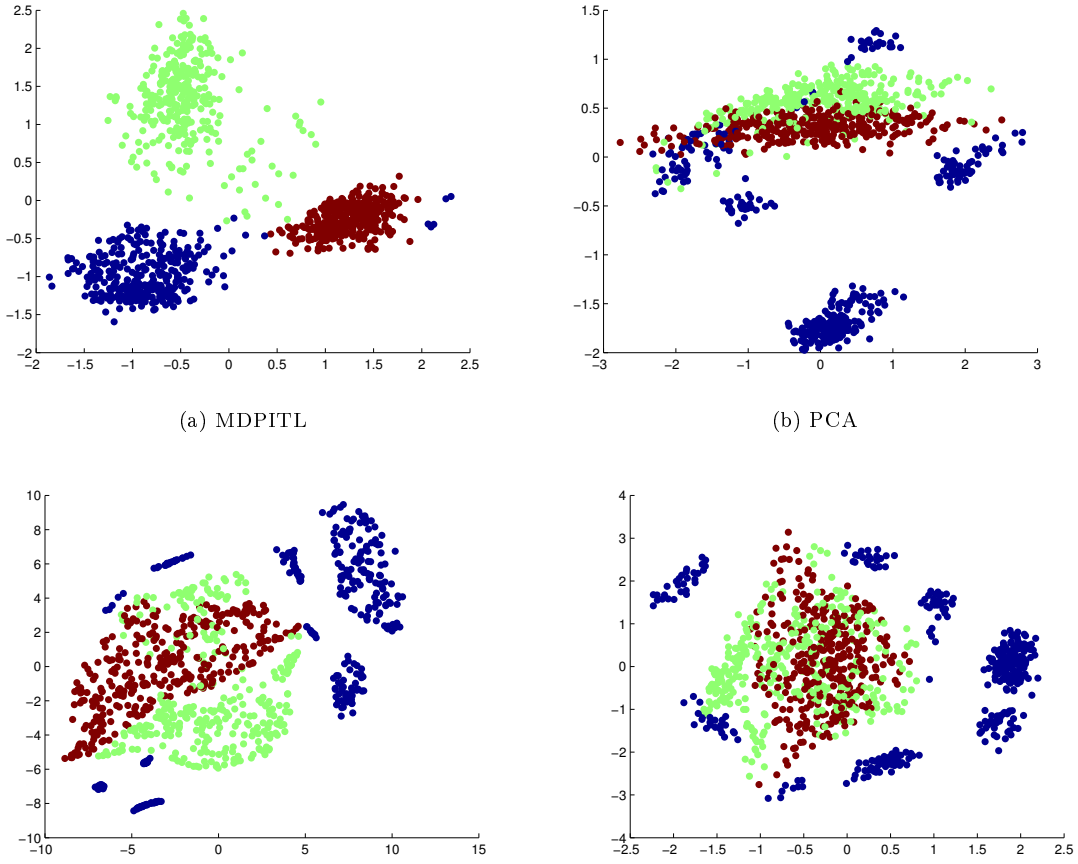
Se proyecta la base utilizando los parámetros descritos en 4.16 y 4.17. Esta base se presenta más compleja que las anteriores debido a la cantidad de datos y el número de dimensiones del espacio de entrada. La figura 4.10 muestra que todos los métodos hacen una proyección muy distinta entre ellos. Las proyecciones de PCA, NLM y SNE muestran un mapa que no entrega mucha información sobre la estructura de la base, dos de las clases se encuentran muy juntas (incluso traslapadas) por lo que resulta muy difícil identificarlas sin la información que dan los marcadores del mapa indicando las clases. Por otro lado, MDP-ITL entrega una visualización que se distancia bastante de las proyecciones de los otros métodos. En el mapa se logra observar claramente como se forman tres clusters de datos donde sólo aparecen algunos *outliers* (datos del conjunto "azul") cerca del conjunto "café". Este tipo de proyección es comparable con la presentada en [22] para el mismo conjunto de datos, con la diferencia de que Torkkola utiliza información de la clase para lograr tal proyección, es decir, hace una proyección supervisada. Como se dijo en el capitulo anterior, el espacio intrínseco de los datos es bi-dimensional, por lo que se puede observar que sólo MDP-ITL fue capaz de recuperar ese espacio.

Tabla 4.16: Lista Parámetros MDPITL base Pipeline. Prueba 1.

Parámetro	**Valor**
i_{max}	150
σ^+	1.0
σ^-	0.5
η^+	1e8
η^-	1e7

Tabla 4.17: Lista Parámetros proyección otros métodos, base Pipeline. Prueba 1.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	400
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	40
	$\sigma_{mínimo}$	0.16
	$\sigma_{máximo}$	1.03



(a) MDPITL

(b) PCA

(c) SNE

(d) NLM

Figura 4.10: Proyecciones *Pipeline*

En este caso las medidas de calidad de la figura 4.11 muestran un comportamiento poco intuible de la observación de los mapas. MDP-ITL tiene un mal desempeño en todas las medidas al comparar con los otros métodos. SNE muestra un buen desempeño en las zonas más locales. Este resultado lleva a la pregunta sobre si estas medidas utilizadas para medir la calidad de los mapas son las mejores. Por otro lado en la tabla 4.18 se observa como el porcentaje de clasificación para MDP-ITL es el más alto demostrando la buena separación de las clases. NLM entrega resultados bastante pobres.

Tabla 4.18: Índice Dunn y k-NN, base Pipeline. Prueba 1.

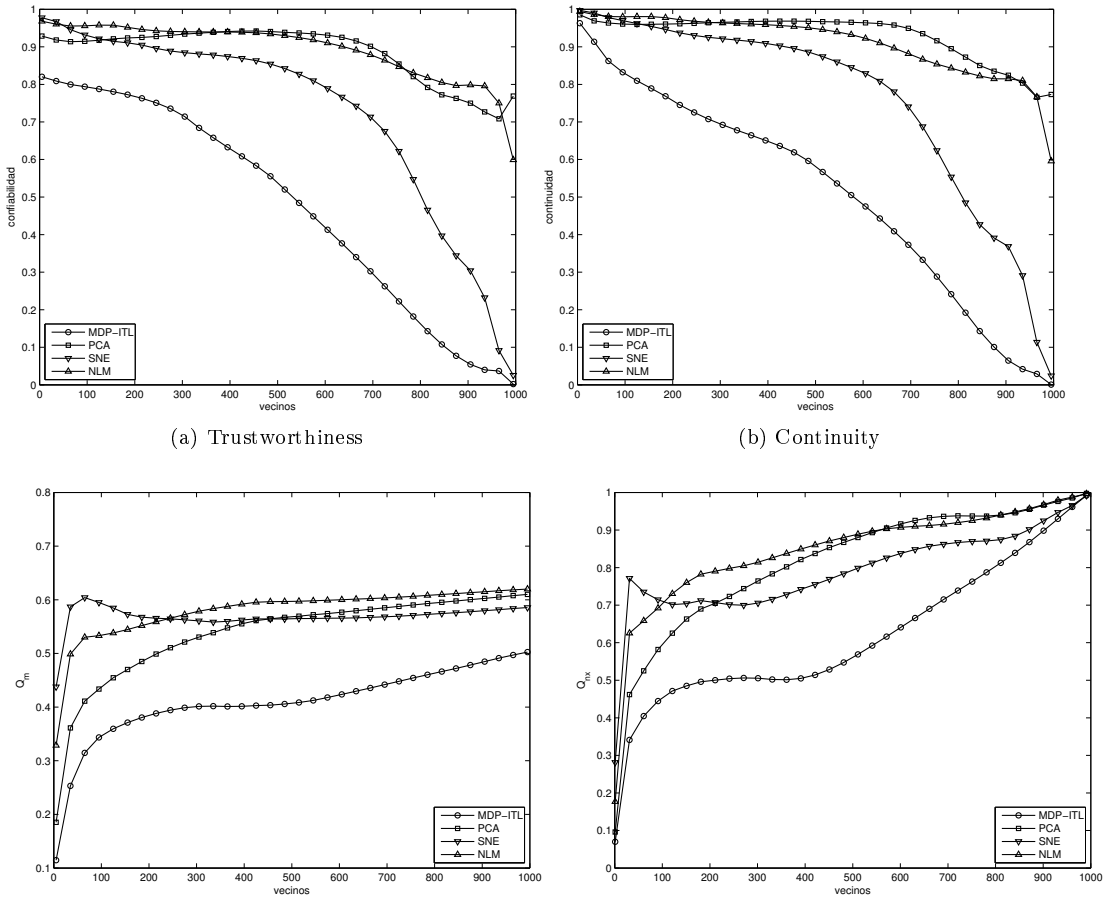
	MDP-ITL	PCA	SNE	NLM
dunn	0.032	0.01	0.009	0.009
k-NN (%)	99.3	86.3	97.1	81.3

Pipeline - Prueba 2

La base se procesa previamente a la proyección de datos escalando cada componente en el intervalo $[-1, 1]$. Los parámetros utilizados por los métodos se describen en las tablas 4.16 y 4.19. Los resultados de las proyecciones se muestran en la figura 4.12 y las medidas de calidad en la figura 4.13 y en la tabla 4.20.

Tabla 4.19: Lista Parámetros proyección base Pipeline. Prueba 2.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	400
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	40
	$\sigma_{mínimo}$	0.18
	$\sigma_{máximo}$	1.02



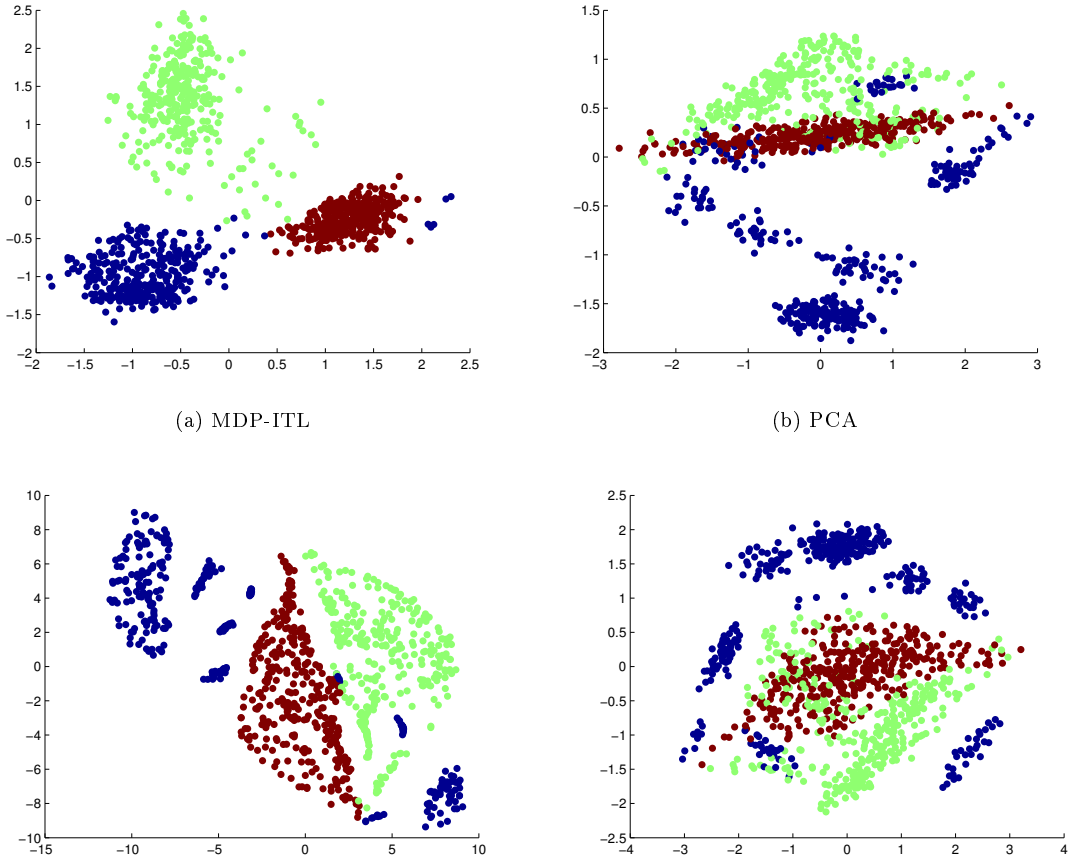
(a) Trustworthiness

(b) Continuity

(c) q_m

(d) q_{nx}

Figura 4.11: Medidas de Calidad Pipeline. Prueba 1.



(a) MDP-ITL

(b) PCA

(c) SNE

(d) NLM

Figura 4.12: Proyecciones base Pipeline. Prueba 2.

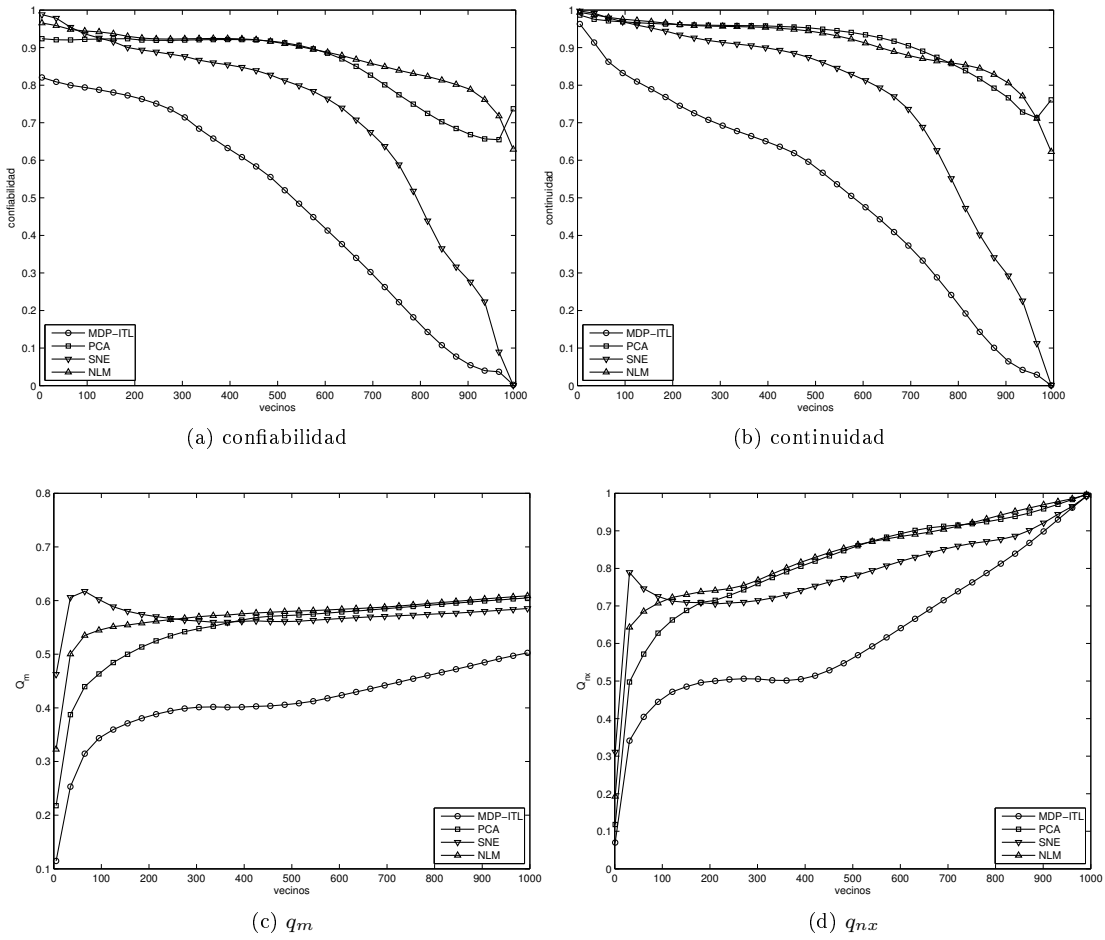
Los resultados no varían mucho del caso no escalado, salvo talvez NLM cuya mejoría se observa sobre todo en el porcentaje de clasificación mostrado en la tabla 4.20.

Tabla 4.20: Índice Dunn y k-NN, base Pipeline. Prueba 2

	MDP-ITL	PCA	SNE	NLM
dunn	0.033	0.011	0.007	0.005
k-NN (%)	99.4	87	95.7	89.3

4.3.4. Iris

Las características de la base de datos se muestra en la tabla 4.21.



(a) confiabilidad (b) continuidad

(c) q_m (d) q_{nx}

Figura 4.13: Medidas de Calidad Pipeline. Prueba 2.

Iris - Prueba 1

Los parámetros utilizados para esta prueba se muestran en las tablas 4.22 y 4.23.

Tabla 4.22: Lista Parámetros MDP-ITL, base Hepta. Prueba 1.

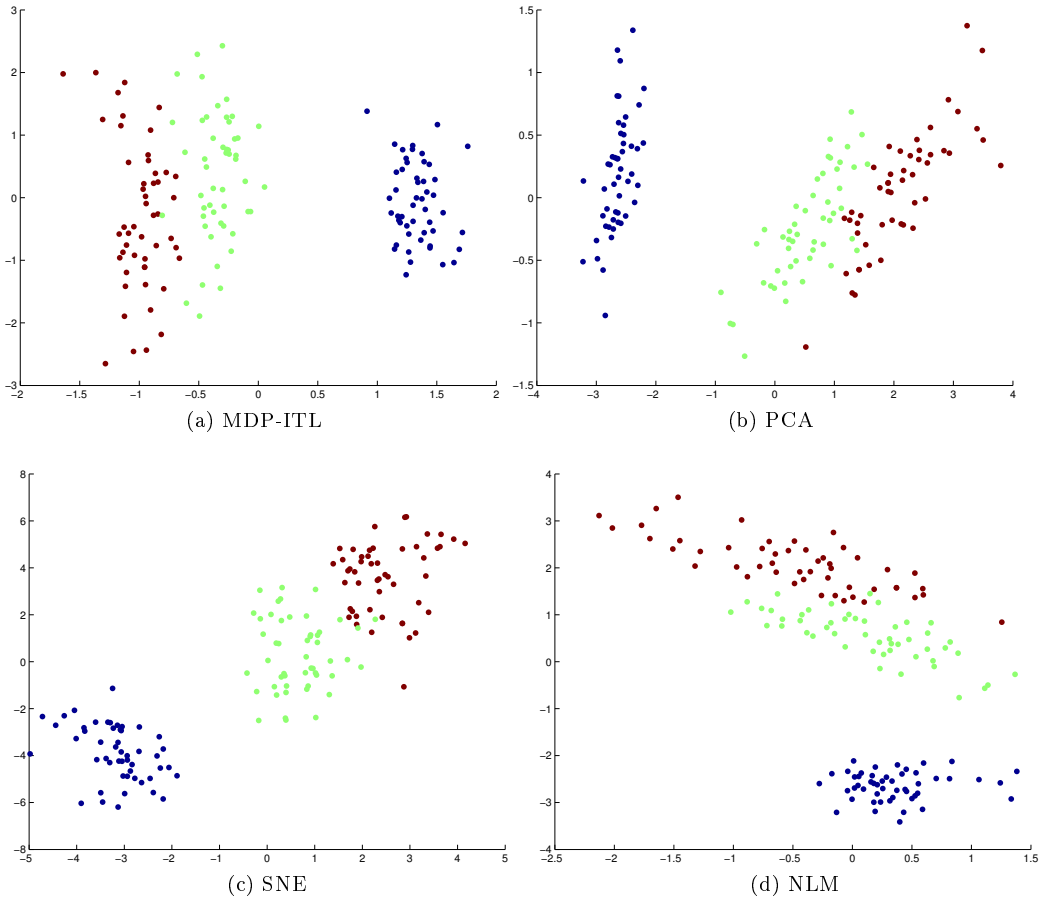
Parámetro	**Valor**
i_{max}	100
σ^+	1.5
σ^-	0.5
η^+	1e5
η^-	1e4

Tabla 4.21: Características base Iris

Característica	**Valor**
Dimensión	4
N° datos	150
Clases	3

Tabla 4.23: Lista Parámetros otros métodos proyección base Hepta. Prueba 1.

Método	**Parámetro**	**Valor**
PCA	dimensión salida	2
NLM	dimensión salida	2
	número iteraciones	100
SNE	dimensión salida	2
	número iteraciones	4000
	perplejidad	30
	$\sigma_{mínimo}$	0.1
	$\sigma_{máximo}$	1.7

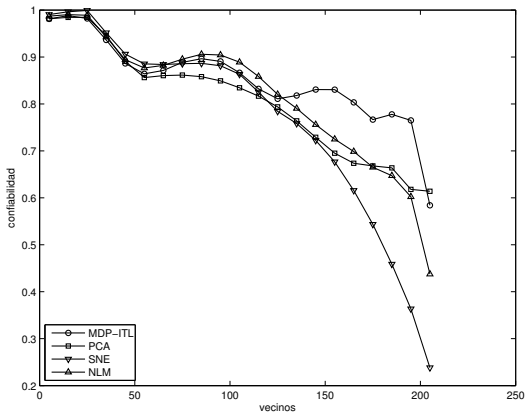


(a) MDP-ITL (b) PCA

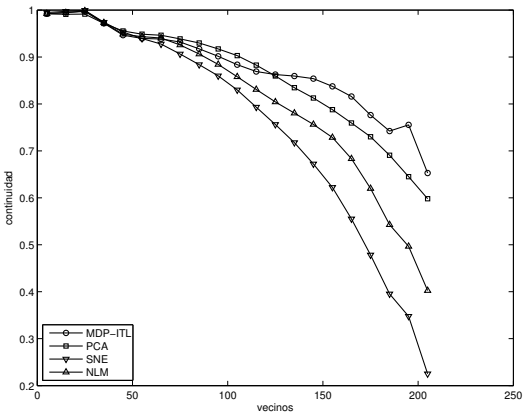
(c) SNE (d) NLM

Figura 4.14: Proyecciones base Iris. Prueba 1.

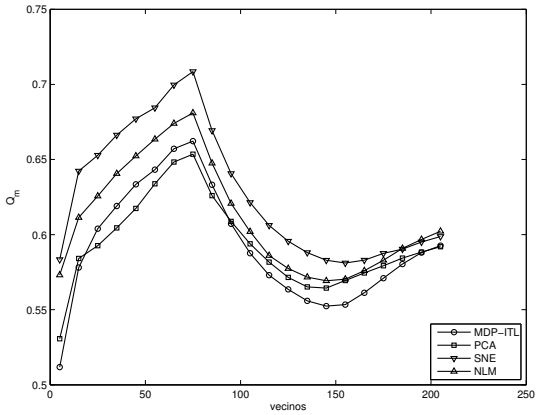
Las medidas de calidad en la figura 4.15 muestran que en las zonas locales de todas las medidas excepto q_m los métodos estan parejos. MDP-ITL destaca en confiabilidad y continuidad en las zonas más globales.



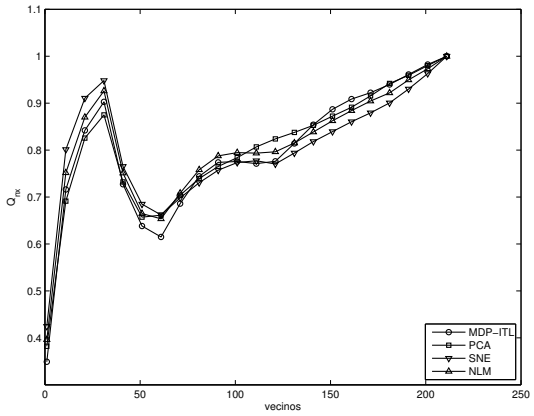
(a) confiabilidad



(b) continuidad



(c) q_m



(d) q_{nx}

Figura 4.15: Medidas de Calidad base Iris. Prueba 1.

Tabla 4.24: Índice Dunn y k-NN, base Iris, prueba 1

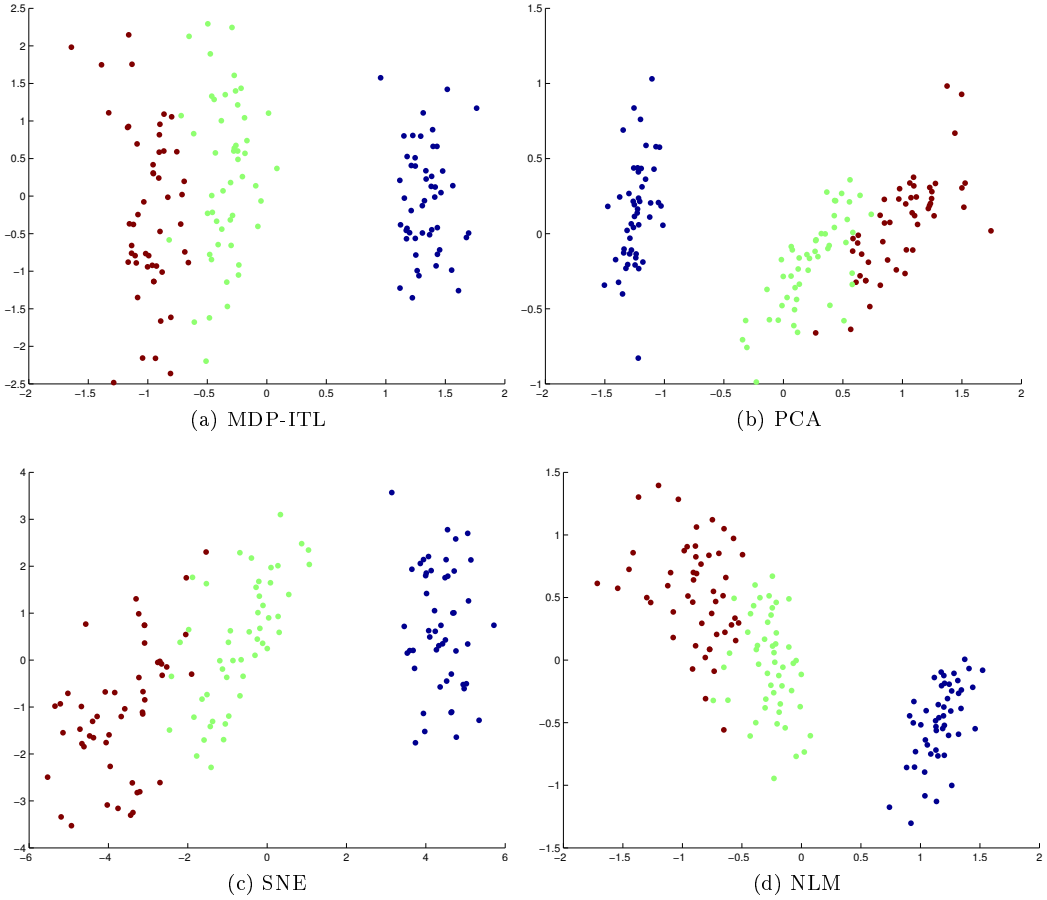
	MDP-ITL	PCA	SNE	NLM
dunn	0.032	0.045	0.032	0.039
k-NN (%)	97.1	96.8	96.5	97.3

Las medidas de clustering de la tabla 4.24 muestran resultados muy parejos para todos los métodos.

Iris - Prueba 2

En esta prueba se escala la base de datos entre $[-1, 1]$, los parámetros utilizados son los mismos que en el caso anterior y estan descritos en las tablas 4.22 y 4.23.

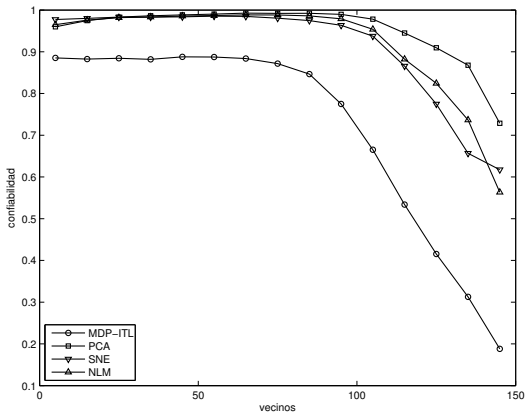
Las medidas de topología para MDP-ITL bajan, hecho que no se ve a simple vista en las proyecciones. Por otro lado la medida de clasificación muestra una pequeña baja pareja para todos los métodos.



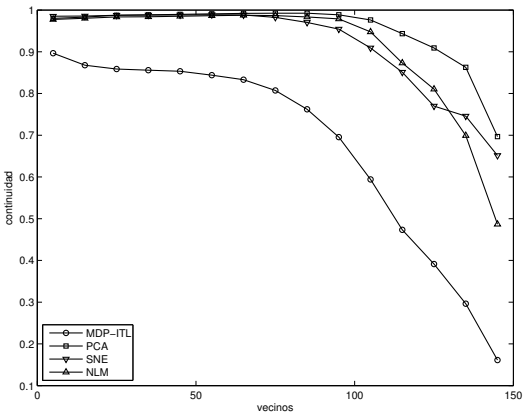
(a) MDP-ITL (b) PCA

(c) SNE (d) NLM

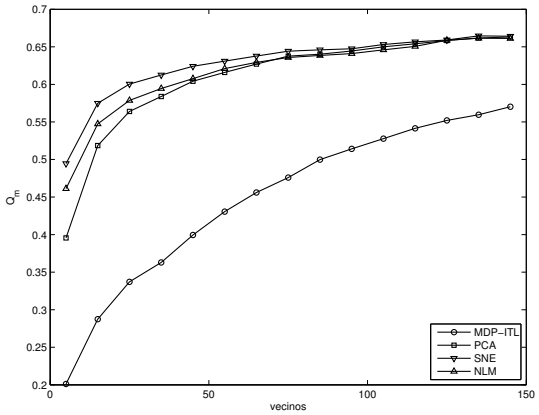
Figura 4.16: Proyecciones base Iris. Prueba 2



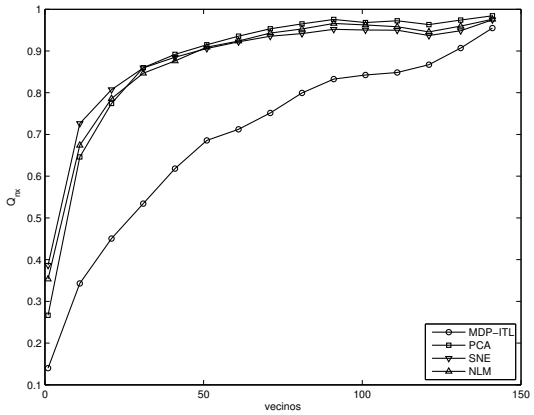
(a) confiabilidad



(b) continuidad



(c) q_m



(d) q_{nx}

Figura 4.17: Medidas de Calidad base Iris. Prueba 2.

Tabla 4.25: Índice Dunn y k-NN, base Iris. Prueba 2

	MDP-ITL	PCA	SNE	NLM
dunn	0.037	0.028	0.065	0.041
k-NN (%)	95.9	94	94.4	94.1

Capítulo 5

Discusión

En este capítulo se analizarán los resultados obtenidos en las simulaciones sobre las bases de datos descritas en el capítulo anterior. Además, se pretende entregar una caracterización o análisis sobre las restricciones del método en términos de cantidad de datos, costos computacionales y dimensionalidad del espacio de entrada con el propósito de definir de la mejor forma posible el campo de aplicación.

Como paso previo al análisis es importante recordar las características principales de los métodos con los que se compara MDP-ITL en este trabajo. Tanto SNE como NLM son métodos no lineales por lo que en teoría son capaces de encontrar estructuras que los métodos lineales no son capaces de encontrar. SNE tiene una característica de preservación de vecindades locales, mientras que NLM intenta preservar las distancias entre los datos de forma global. Por otro lado PCA es un método de proyección lineal donde la preservación de topología se hace en forma global al igual que MDP-ITL.

5.1. Análisis de Resultados

Una cosa que se desprende de los resultados es la dificultad de encontrar una métrica única para medir las proyecciones, por lo que siempre es necesario observar visualmente la proyección, además de los resultados de las métricas. Desde este punto de vista se observa que MDP-ITL es mejor que PCA ya que genera mejores mapas, lo que se refleja la mayoria de las veces en las medidas de preservación topológica y en las medidas de clustering. En las bases Tetra y Hepta los resultados muestran que MDP-ITL es capaz de hacer la proyección de una alta dimensión a una menor sin

afectar la estructura de los datos.

El caso de la base Pipeline es claro, el único método que logra presentar una proyección con cierto nivel de representatividad de la dimensión intrínseca de los datos es MDP-ITL. La separación de las clases es tan buena que k-NN obtiene un porcentaje de clasificación superior al 99 %. Lo que llama la atención es que estos resultados no se correlacionan con lo que se observa en los gráficos de continuidad, confiabilidad, q_m y q_{nx}. Como se mencionó antes, el resultado es muy similar visualmente al que se obtiene en [22] utilizando información de la clase para hacer la separación.

5.2. Dependencia Parámetro σ

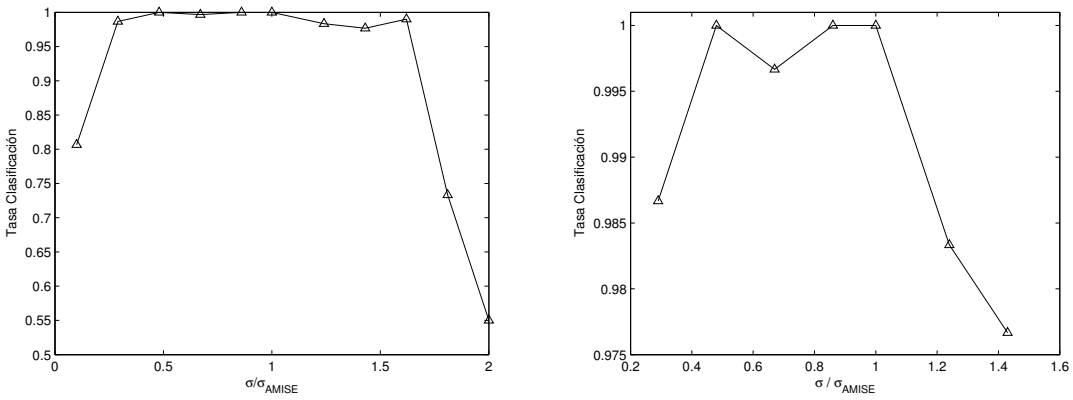
MDP-ITL depende principalmente de un parámetro, el ancho de banda de los kernel gaussianos, σ. En la estimación de Parzen, es el parámetro que define que tan suave es la aproximación a la distribución. Un valor de σ muy grande puede hacer que el algoritmo no converja, mientras que un valor muy chico hace que la superficie de optimización contenga muchos mínimos locales. Para evitar estos problemas se presentó el procedimiento del *annealing*, el cual mueve el valor de σ desde un valor amplio a uno más pequeño lo que permite una mejor convergencia. Para estimar el valor adecuado de σ se ha utilizado la regla de Silverman 2.57, la cual en teoría da una aproximación al valor optimo.

La figura 5.1 muestra como afecta la tasa de clasificación de k-NN al mover σ en la proyección de la base *Pipeline*. Para esta prueba se hicieron varias proyecciones variando σ entre 0.1 y 2 veces el σ_{AMISE} de Silverman. En este caso no se hizo *annealing*, es decir, el σ se mantuvo fijo en cada prueba.

Se observa como los mejores resultados se obtienen con los valores cercanos al óptimo de Silverman. Este hecho sumado al proceso de *annealing*, hacen que se alcancen buenos resultados en términos de σ. También es importante notar que fuera de cierto radio en torno al óptimo el desempeño cae drásticamente.

5.3. Caracterización del Modelo

La mayoría de los métodos de proyección, y de aprendizaje en general, poseen caracteristicas que los hacen funcionar bien en cierto subconjunto del espacio total de problemas. Lo importante



(a) Variación de σ entre 0.1 y 2 veces σ_{AMISE} (b) Acercamiento a zona con valores maximos

Figura 5.1: Dependencia clasificación k-nn con parámetro σ base Pipeline

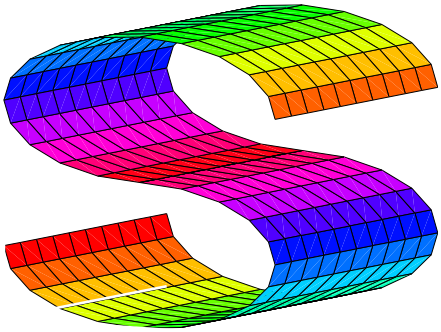
es conocer las restricciones del modelo con el fin de reconocer este subconjunto de problemas donde el método es aplicable o donde alcanza su máximo desempeño, así como los problemas que no es capaz de resolver. Por esta razón se entrega un análisis sobre ciertas características del método como la carga computacional.

5.3.1. *Manifold Learning*

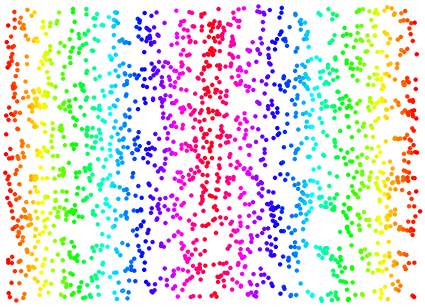
El método desarrollado al ser globalmente lineal no es capaz de encontrar algunos subespacios de baja dimensión embebidos en el espacio de alta dimensión como lo hacen algoritmos del tipo ISOMAP o LLE. Esto se puede observar en el ejemplo de la figura 5.2. En ese caso, ISOMAP logra "desenrollar" el subespacio, estirando el *manifold* que se encontraba curvado, por otro lado MDP-ITL sólo hace una proyección global de los datos sin la posibilidad de desenrollar el subespacio bidimensional.

5.3.2. Orden Computacional

MDP-ITL está basado en el cómputo de la información mutua en base a interacciones de pares de datos, por lo que en cada iteración o cálculo del gradiente es necesario hacer $O(N^2)$ operaciones lo que implica que para llegar al óptimo es necesario hacer $O(I \cdot N^2)$ operaciones con I el número total de iteraciones. Este hecho puede hacer que el método sea inaplicable en bases de datos demasiado grandes ya que aparecen problemas de memoria para guardar los datos, y de velocidad al aumentar



(a) *Scurve*, subespacio bi-dimensional embebido en un espacio de tres dimensiones



(b) Proyección de *Scurve* por ISOMAP

(c) Proyección de *Scurve* por MDPITL

Figura 5.2: Diferencia proyecciones ISOMAP y MDPITL en base *Scurve*

el número de cálculos en forma cuadrática. Sin embargo, existen formas de alivianar la carga computacional del sistema debido a las características especiales de los cálculos. Tomando como ejemplo el cálculo del potencial de información conjunto

$$V_J = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} G\left(x_i - x_j, 2\sigma_x^2 I\right) G\left(y_i - y_j, 2\sigma_y^2 I\right), \tag{5.1}$$

se oberva que este depende de la doble suma de kernels gaussianos. Es bien sabido que el 90 % de los datos en una distribución gaussiana escalar normalizada ($\mu = 0, \sigma = 1$) cae dentro del intervalo $[-1{,}65, 1{,}65]$, intervalo que se reduce drásticamente a medida que se aumenta la dimensión del kernel. Esto quiere decir que a cierta distancia del centro del kernel, la influencia de éste tiende a

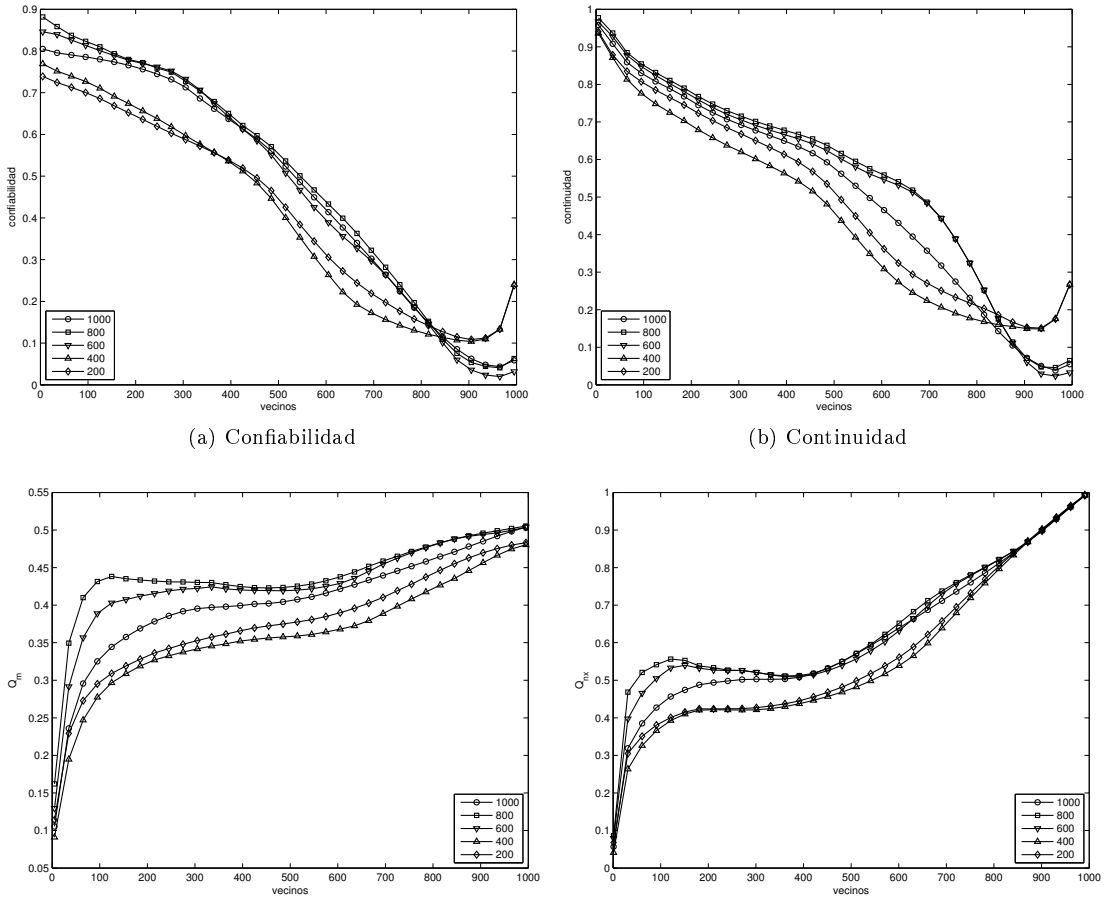
cero por lo que hacer los cálculos es una perdida de recursos debido a que éstos no afectan en mayor medida el resultado final. Bajo esta premisa es posible establecer una zona de acción o híper esfera, definida por cierta distancia al centro del kernel (posición del dato o ejemplo) fuera de la cual no se harán los cálculos. Análogamente se puede establecer como criterio el número de k vecinos más cercanos sobre el cual se harán los cálculos. Tomando este último caso como ejemplo, el cálculo del potencial de información conjunto quedaría como

$$V_J = \frac{1}{NK}\sum_{i=1}^{N}\sum_{j=1}^{K} G\left(x_i - x_j, 2\sigma_x^2 I\right) G\left(y_i - y_j, 2\sigma_y^2 I\right), \tag{5.2}$$

con lo que se reduce el orden de operaciones en cada iteración de $O(N^2)$ a $O(NK)$, donde K puede ser mucho menor a N, lo que puede traer un gran beneficio en términos de eficiencia computacional.

Para analizar como se ve afectada la proyección de una base de datos si sólo se utilizan los K vecinos más cercanos, en la figura 5.3 se muestran las medidas de calidad para diversas proyecciones de la base *Pipeline* utilizando {200, 400, 600, 800, 1000} vecinos.

El gráfico muestra como al utilizar 200 y 400 vecinos que corresponen a un 20 y 40 % del total de los datos respectivamente, se tiene una caída en términos de calidad, sin embargo, la proyección de todos modos es posible. El hecho a destacar es que al utilizar 600 y 800 vecinos se tiene incluso un mejor desempeño que al utilizar la base completa. Estos resultados muestran que es completamente posible acelerar el proceso de proyección utilizando menos datos sin grandes pérdidas de desempeño.



(a) Confiabilidad

(b) Continuidad

(c) Q_m

(d) Q_{nx}

Figura 5.3: Medidas de calidad para proyección base *Pipeline* con distinta cantidad de vecinos.

Capítulo 6

Conclusiones

En este trabajo de tesis se desarrolló un método para proyectar datos multidimensionales a un espacio de menor dimensión, de forma no supervisada, utilizando como criterio de optimización la Teoria de la Información, en particular el criterio Infomax propuesto por Linsker [58]. El método propuesto utiliza el criterio de Infomax para optimizar los parámetros de un mapeo lineal representado por una matriz de transformación W, es decir, busca los parámetros que maximicen la Información Mutua entre la entrada y la salida de esta red lineal. Esta matriz de transformación define la proyección de los datos de entrada al espacio de salida de la forma $\mathrm{Y} = \mathrm{W}^{\mathrm{T}}\mathrm{X}$, donde además se estableció la restricción de la ortonormalidad de W para mejorar la interpretación de la visualización.

Los resultados expuestos demuestran que el método es capaz de resolver el problema de proyectar datos multidimensionales a dos dimensiones, y que la utilización de ITL como criterio para proyectar o visualizar datos multidimensionales es una alternativa a los métodos clásicos basados en distancias o MSE. La fuerte base matemática con la que se desarrollan los criterios de ITL hace que la optimización de los parámetros del sistema pueda ser desarrollada utilizando algoritmos clásicos de optimización sin necesidad de recurrir a heurísticas o a la llamada fuerza bruta. La utilización de las divergencias cuadráticas propuestas por Principe [17] como estimadores de la Información Mutua, permitió en la práctica la implementación del criterio Infomax para la proyección de datos, el cual resulta imposible de implementar bajo los conceptos clásicos de Shannon, tanto en el sentido matemático como computacional, sin asumir distribuciones a priori de los datos. Esto permite que

el método sea aplicable en bases de datos con distintas distribuciones de las que no es necesario tener conocimiento previo por lo que se puede calificar el método como "no paramétrico". Por otro lado el hecho de que los criterios de ITL presentados sean independiente del mapeo o modelo de máquina de aprendizaje utilizada, abre un campo de investigación donde los resultados pueden ser mejorados si se encuentra o desarrolla un mapeo más sofisticado sin tener que hacer cambios a los criterios ni a la forma de implementarlos.

Las comparaciones del método basado en ITL con el clásico método lineal PCA, muestran que MDP-ITL obtiene mejores resultados en las bases de datos probadas. Las proyecciones resultantes son diferentes ya que se basan en distintos criterios de optimización aunque el modelo de proyección sea similar. El hecho de utilizar la información de la distribución de los datos y no sólo la varianza permite que MDP-ITL pueda hacer mejores proyecciones y pueda resolver problemas que PCA no puede, como sucede por ejemplo cuando las varianzas de las diferentes componentes son iguales o muy similares. Al ver las diferencias entre los resultados de MDP-ITL con los métodos no lineales se observa que los resultados son comparables, a pesar de que con ciertas medidas el desempeño es más bajo en algunas zonas, los mapas resultantes entregan información importante sobre el espacio de entrada.

El modelo y las restricciones impuestas hacen que el método desarrollado sea catalogado como un método del tipo lineal y de proyección global lo que restringe su aplicabilidad cuando los datos existen en espacios altamente no lineales como es el caso de un *manifold* bidimensional "enrollado" y embebido en un espacio de dimensión superior.

El modelo propuesto considera un parámetro sensible de definir, el valor del ancho de banda de los kernel gaussianos, σ, el cual es razonablemente aproximado por la regla de Silverman (2.57), lo que combinado con el proceso de *annealing* hacen que la elección del σ no sea tan compleja como podría ser sin tener una aproximación inicial. El ajuste de la tasa de aprendizaje depende principalmente del tamaño de la base de datos y su dimensionalidad.

El orden computacional del método $O(N^2)$ puede presentar un problema al querer aplicarlo en bases de datos muy grandes, sin embargo gracias a su naturaleza basada en kernels es posible reducirlo sin perder demasiado desempeño, incluso mejorándolo en ciertos casos al eliminar de los cálculos datos muy lejanos que pueden agregar más ruido que información relevante al sistema.

6.1. Recomendaciones para Futura Investigación

La principal limitación del método desarrollado en esta tesis es su característica lineal, por lo que el paso natural a seguir es utilizar algún modelo de máquina de aprendizaje que permita hacer proyecciones no lineales a un costo no prohibitivo. El modelo no lineal más conocido y utilizado son las RNA, sin embargo, una RNA por si sola no resuelve el problema, es necesario agregar algún criterio para que la red mantenga cierto orden en la proyección. Un modelo utilizable son las redes neuronales *autoencoders* (autocodificables) basadas en la utilización de una RNA como codificador (*encoder*), la cual se encarga de transformar los datos de alta a baja dimensión, y otra red similar llamada decodificador (*decoder*) que intenta recuperar los datos codificados. En [84], Hinton y Salakhutdinov proponen entrenar una red *autoencoder* utilizando como criterio de optimización la entropía cruzada entre la entrada y la salida. Dado los conocimientos que ya se tienen sobre teoría de la información no es difícil darse cuenta de que una posibilidad es utilizar alguna medida de divergencia sobre las distribuciones de los datos de entrada y salida, no limitándose sólo a la IM. Lo complicado del modelo es establecer las restricciones para que la proyección resultante mantenga cierta estructura topológica sobre los datos.

Otra posibilidad es utilizar directamente las fuerzas de información cuadrática (2.78) para mover la posición de los datos de salida. De acuerdo a pruebas preliminares realizadas, el algoritmo converge a ciertos máximos de la distribución de los datos por lo que es necesaria una etapa intermedia que se encargue de mantener las restricciones para que el mapa de salida entregue información topológica representativa del espacio de entrada. Lo ideal sería desarrollar un modelo que permita al sistema darle una mayor importancia a las proyecciones locales lo que permite además disminuir el costo computacional ya que sólo bastaría con hacer cálculos por vecindades sin necesidad de utilizar todos los datos para cada cálculo.

El calculo de un σ óptimo es también un tema de investigación por si solo. El desarrollar un criterio para calcular un σ fijo o adaptivo puede ser bastante beneficioso no sólo para este problema si no que para cualquier algoritmo basado en kernels.

Por último, la TI se podria utilizar para construir una medida de desempeño que sea más adecuada que las existentes para medir la calidad de las proyecciones. Probablemente puede que alguna medida basada en la fdp de los datos sea más adecuada que una basada en rankings o vecinos cercanos.

Bibliografía

[1] R. Shepard, "The analysis of proximities: multidimensional scaling with an unknown distance function i and ii," *Psychometrika*, vol. 27, pp. 125–140, 219–246, 1962.

[2] J. Sammon, J. W., "A nonlinear mapping for data structure analysis," pp. 401–409, May 1969.

[3] P. Demartines and J. Herault, "Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets," vol. 8, pp. 148–154, Jan. 1997.

[4] J. A. Lee, A. Lendasse, N. Donckers, and M. Verleysen, "A robust nonlinear projection method," in *In Proc. 8th European Symp. Artificial Neural Networks (ESANN2000)*, (Bruges, Belgium), Apr. 2000.

[5] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction.," *Science*, vol. 290, pp. 2319–2323, December 2000.

[6] L. K. Saul and S. T. Roweis, "Think globally, fit locally: unsupervised learning of low dimensional manifolds," *J. Mach. Learn. Res.*, vol. 4, pp. 119–155, 2003.

[7] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, December 2000.

[8] T. Kohonen, *Self-organizing maps*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.

[9] V. De Silva and J. B. Tenenbaum, "Global versus local methods in nonlinear dimensionality reduction," in *Advances in Neural Information Processing Systems 15*, vol. 15, pp. 705–712, 2003.

[10] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical magazine*, vol. 6, pp. 559–572, 1901.

[11] G. Hinton and S. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems 15*, vol. 15, pp. 833–840, 2003.

[12] S. Kullback, *Information Theory and Statistics.* New York: Dover Publications, 1968.

[13] T. Iwata, K. Saito, N. Ueda, S. Stromsten, T. L. Griffiths, and J. B. Tenenbaum, "Parametric embedding for class visualization," *Neural Comput.*, vol. 19, no. 9, pp. 2536–2556, 2007.

[14] P. A. Estevez, C. J. Figueroa, and K. Saito, "Cross-entropy approach to data visualization based on the neural gas network," in *Proc. IEEE International Joint Conference on Neural Networks IJCNN '05*, vol. 5, pp. 2724–2729, 31 July–4 Aug. 2005.

[15] P. A. Estévez, C. J. Figueroa, and K. Saito, "Cross-entropy embedding of high-dimensional data using the neural gas model," *Neural Netw.*, vol. 18, no. 5-6, pp. 727–737, 2005.

[16] K. J. S. T. M. Martinetz, "A neural-gas network learns topologies," *Artificial Neural Networks*, pp. 397–402, 1991.

[17] J. Principe, J. F. III, and D. Xu, *Information theoretic learning. In Simon Haykin, editor.* New York, NY: Unsupervised Adaptive Filtering, Wiley, 2000.

[18] C. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–653, 1948.

[19] A. Renyi, "On measures of entropy and information," in *Proc. 4th Berkeley Symp. Math. Stat. and Prob.*, vol. 1, pp. 547–561, 1961.

[20] E. Parzen, "On the estimation of a probability density function and mode," *Annals of Mathematical Statistics*, vol. 33, pp. 1065–1076, 1962.

[21] K. Fukunaga, *Introduction to Statistical Pattern Recognition, Second Edition (Computer Science and Scientific Computing Series).* Academic Press, September 1990.

[22] K. Torkkola, "Feature extraction by non parametric mutual information maximization," *J. Mach. Learn. Res.*, vol. 3, pp. 1415–1438, 2003.

[23] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, pp. 1373–1396, 2002.

[24] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, August 2006.

[25] H. Hotelling, "Analysis of a complex of statistical variables into principal components," *Journal of Educational Psychology*, vol. 24, pp. 417–441, 1933.

[26] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[27] D. Alahakoon, S. K. Halgamuge, and B. Sirinivasan, "Dynamic self organizing maps with controlled growth for knowledge discovery," *IEEE Transactions on Neural Networks, Special Issue on Knowledge Discovery and Data Mining*, vol. 11, pp. 601–614, 2000.

[28] A. Ultsch and H. P. Siemon, "Kohonen's self-organizing feature maps for exploratory data analysis," in *Proceeding of International Neural Networks Conference (INNC '90)*, (Dordrecht, Netherlands, Kluwer), pp. 305–308, 1990.

[29] D. Merkl and A. Rauber, "Alternative ways for cluster visualization in self-organizing maps," in *Proceeding onWorkshop of the Self-Organizing Map (WSOM '97)*, (Finland: Helsinki University Technology, HUT, T. Kohonen), pp. 106–111, Ed. Espoo, June 4-6 1997.

[30] A. Ultsch, "Maps for the visualization of high-dimensional data spaces," in *Proceeding of the Workshop Self-Organizing Map (WSOM '03)*, pp. 225–230, September 2003.

[31] C. J. Figueroa and P. A. Estevez, "A new visualization scheme for self-organizing neural networks," in *Proc. IEEE International Joint Conference on Neural Networks*, vol. 1, 25–29 July 2004.

[32] H. U. Bauer and K. R. Pawelzik, "Quantifying the neighborhood preservation of self-organizing feature maps," vol. 3, pp. 570–579, July 1992.

[33] T. Villmann, R. Der, M. Herrmann, and T. M. Martinetz, "Topology preservation in self-organizing feature maps: exact definition and measurement," vol. 8, pp. 256–266, March 1997.

[34] A. Konig, "Interactive visualization and analysis of hierarchical neural projections for data mining," vol. 11, pp. 615–624, May 2000.

[35] J. Venna and S. Kaski, "Neighborhood preservation in nonlinear projection methods: An experimental study," pp. 485–491, 2001.

[36] J. Venna and S. Kaski, "Local multidimensional scaling," *Neural Networks*, vol. 19, no. 6, pp. 889–899, 2006.

[37] J. Lee and M. Verleysen, "Rank-based quality assessment of nonlinear dimensionality reduction.," *Proceedings of ESSAN 2008, 16th European Symposium on Artificial Neural Networks*, pp. 49–54. d–side, April 2008.

[38] R. Xu and D. C. W. II, *Clustering*. IEEE Press and Jhon Wiley Inc., New Jersey, 2009.

[39] G. Leban, B. Zupan, G. Vidmar, and I. Bratko, "Vizrank: Data visualization guided by machine learning.," *Data Mining and Knowledge Discovery*, vol. 13, pp. 119–136, 2006.

[40] S. Haykin, *Neural Networks and Learning Machines*. Prentice Hall, 3 ed., Nov. 2008.

[41] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana: University of Illinois Press, 1962.

[42] H. Nyquist, "Certain factors affecting telegraph speed," *Bell System Technical Journal*, vol. 3, pp. 332–333, 1924.

[43] R. V. Hartley, "Transmission of information," *Bell System Technical Journal*, vol. 7, pp. 535–563, 1928.

[44] A. Renyi, "Some fundamental questions of information theory," in *Selected Papers of Alfred Renyi*, vol. 2, (Akademiai Kiado, Budapest), pp. 526–552, 1960.

[45] J. Kapur, *Measures of Information and Their Applications*. John Wiley & Sons, 1994.

[46] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. San Diego: Academic Press, 1999.

[47] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm (with discussion)," *Journal of the Royal Statistical Society B*, vol. 39, pp. 1–38, 1977.

[48] G. J. McLachlan and K. E. Basford, *Mixture Models: Inference and Applications to Clustering*. New York: Marcel Dekker, Inc., 1988.

[49] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*. New York: John Wiley & Sons, Inc., 1996.

[50] E. J. Wegman, "Nonparametric probability density estimation: I. a summary of available methods," *Technometrics*, vol. 14, 1972.

[51] B. Silverman, *Density estimation for statistics and data analysis*. Chapman and Hall, 1986.

[52] M. P. Wand and M. C. Jones, *Kernel Smooting*. London: Chapman and Hall, 1995.

[53] D. W. Scott, *Multivariate Density Estimation*. New York: JohnWiley & Sons, 1992.

[54] G. Deco and D. Obradovic, *An Information-Theoretic Approach to Neural Computing*. New York: Springer, 1996.

[55] S. Haykin, *Neural Networks: A Comprehensive Foundation, Second Edition*. Englewood Cliffs, NJ: Prentice Hall, 1998.

[56] G. H. Hardy, J. E. Littlewood, and G. Polya, *Inequalities*. Cambridge: University Press, 1934.

[57] P. Grassberger and I. Procaccia, "Characterization of strange attractors," *Phys. rev. Letters*, vol. 50, no. 5, pp. 346–349, 1983.

[58] R. Linsker, "An application of the principle of maximum information preservation to linear systems," pp. 186–194, 1989.

[59] D. Xu, *Energy, Entropy and Information Potential for Neural Computation*. PhD thesis, University of Florida, 1999.

[60] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, pp. 386–408, 1958.

[61] F. Rosenblatt, *Principles of Neurodynamics: Perceptron and Theory of Brain Mechanism*. Washington DC: Spartan Books, 1962.

[62] M. L. Minsky and S. A. Papert, *Perceptrons*. Cambridge, MA: MIT Press, 1969.

[63] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[64] A. R. Gallant and H. White, "There exists a neural network that does not make avoidable mistakes," in *Proc. IEEE International Conference on Neural Networks*, pp. 657–664, 24–27 July 1988.

[65] T. Poggio and F. Girosi, "Networks for approximation and learning," vol. 78, pp. 1481–1497, Sept. 1990.

[66] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, pp. 246–257, 1991.

[67] K. J. Lang and G. E. Hinton, "The development of the time-delay neural network architecture for speech recognition," technical report cmu-cs-88-152, Carnegie-Mellon University, Pittsburgh, PA, 1988.

[68] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, pp. 328–339, March 1989.

[69] B. DeVries and J. C. Principe, "The gamma model–a new neural model for temporal processing," *Neural Networks*, vol. 5, pp. 565–576, 1992.

[70] J. C. Principe, B. de Vries, and P. G. de Oliveira, "The gamma-filter-a new class of adaptive iir filters with restricted feedback," vol. 41, pp. 649–656, Feb. 1993.

[71] K. I. Diamantaras and S. Y. Kung, *Principal Component Neural Networks, Theory and Applications*. New York: John Wiley & Sons, Inc, 1996.

[72] E. T. Jaynes, "Information theory and statistical mechanics," *Physical Review*, vol. 106, pp. 620+, May 1957.

[73] J. Kapur and H. Kesavan, *Entropy optimization principles and applications*. Academic Press, 1992.

[74] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[75] J. Atick, "Could information theory provide an ecological theory of sensory processing?," *Network: Computation in Neural Systems*, vol. 3, pp. 213–251, 1992.

[76] H. Barlow, T. Kaushal, and G. Mitchison, "Finding minimum entropy codes," *Neural Computation*, vol. 1, no. 3, pp. 412–423, 1989.

[77] P. Comon, "Independent component analysis: a new concept?," *Signal Proc.*, vol. 36, no. 3, pp. 287–314, 1994.

[78] J. F. Cardoso, "Infomax and maximum likelihood for blind source separation," vol. 4, pp. 112–114, April 1997.

[79] H. H. Yang and S. Amari, "Adaptive on-line learning algorithms for blind separation – maximum entropy and minimum mutual information," *Neural Computation*, vol. 9, no. 7, 1997.

[80] M. Plumbey and F. Fallside, "Sensory adaptation: an information theoretic viewpoint," in *Int. Conf. on Neural Nets*, vol. 2, p. 598, 1989.

[81] M. Verleysen, *Machine learning of high-dimensional data: Local artificial neural networks and the curse of dimensionality.* PhD thesis, Université catholique de Louvain, 2000.

[82] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Netw.*, vol. 13, no. 4-5, pp. 411–430, 2000.

[83] I. Kojadinovic, "Relevance measures for subset variable selection in regression problems based on k-additive mutual information," *Computational Statistics and Data Analysis*, vol. 49, pp. 1205–1227, June 2005.

[84] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, July 2006.