UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

SISTEMA DE AGRUPACIÓN DE NOTICIAS PARA EL CONTRASTE DE MEDIOS DE COMUNICACIÓN CHILENOS TRADICIONALES E INDEPENDIENTES

MEMORIA PARA OPTAR AL TÍTULO DE

INGENIERO CIVIL EN COMPUTACIÓN

MAXIMILIANO ENRIQUE VARGAS VARGAS

PROFESORA GUÍA:
BÁRBARA POBLETE LABRA

MIEMBROS DE LA COMISIÓN:
FELIPE BRAVO MÁRQUEZ
JUAN MANUEL BARRIOS NUÑEZ

Este trabajo ha sido parcialmente financiado por FONDECYT 1191604

SANTIAGO DE CHILE
2022

Resumen

La mayoría de los chilenos se informa a través de redes sociales y portales digitales de noticias. El periodismo chileno se encuentra ante múltiples escenarios: presencia de discurso de odio, *fake news*, desconfianza en medios tradicionales, un auge por información en redes sociales y medios independientes, la concentración de los medios tradicionales y también, la presencia de burbujas informativas. A modo de mejorar la calidad de la información, el presente trabajo busca abordar los problemas de sobrecarga de información, el quiebre de las burbujas informativas y también, la generación de métricas respecto de los sesgos periodísticos.

Para ello, se implementa un prototipo de sistema (*Proof of Concept* en inglés) escalable y flexible que recolecte artículos desde medios tradicionales e independientes, los agrupe por similitud formando eventos y los muestre en forma clara y ordenada. También, se utilizan tecnologías del Procesamiento del Lenguaje Natural (PLN) tanto para la medición de subjetividad y polaridad en texto, como la generación de resúmenes de eventos. Para el agrupamiento de noticias, se emplean dos metodologías. La primera usa modelos PLN basados en *transformers* para obtener representaciones semánticas del texto, que luego son agrupadas por algoritmos de *clustering* de una manera no supervisada. La segunda se basa en una heurística de palabras claves para agrupar titulares similares.

La plataforma desarrollada cuenta con dos secciones principales: agrupaciones y buscador. La primera muestra los resultados de las agrupaciones para una ventana de tiempo determinada. La segunda busca ser de utilidad pública para buscar artículos junto a las métricas de subjetividad y polaridad.

La calidad de las agrupaciones se evalúa en forma cualitativa, debido a la falta de una evaluación supervisada. Ambas metodologías muestran resultados prometedores, siendo la heurística la más eficiente en términos de rendimiento. Los modelos PLN utilizados han sido probados en diversos *datasets* y por esto, se tiene el supuesto de que deben funcionar bien en el sistema. Por último, para evaluar la utilidad del sistema completo, se analizan algunos casos de uso, y además, se efectúan veinticuatro entrevistas para recolectar comentarios y apreciaciones.

Finalmente, se demuestra la factibilidad técnica de este concepto, pero evidenciando la necesidad de recursos para el desarrollo a gran escala. Por otro lado, todos los entrevistados entregan comentarios muy positivos de la plataforma, calificándola como novedosa, que usarían con frecuencia y plantean muchos contextos donde puede ser usada: en el ámbito público, académico y empresarial.

A mi abuelita Lucy,
la persona más alegre y sencilla que conozco y que amaré por siempre.

Agradecimientos

Agradezco con mucho cariño a mi familia, a mis padres, mi hermano, mis abuelos y todos quienes han formado parte de mi vida desde que llegué a este mundo, que me han cuidado y querido con cariño.

Especialmente, también agradezco a Valentina, con quien he vivido hermosas experiencias y que me ha apoyado desde que nos conocimos. Y por supuesto al Mati, una luz en mi vida y que ahora nos acompaña en nuestros corazones.

A mis amigas y amigos de colegio, quienes han sido una alegría en mi vida, y en especial al Juancho y al Jimmy quienes me ayudaron a darle un tinte único a este trabajo.

A las y los compas de la u, del Eolian, de Los Pingüinos, del DVI, del Toqui, del DCC, de Imprito, del Team CuarentenaFat y tantos otros que han hecho de mi vida universitaria un recuerdo que llevaré siempre conmigo. A todos y todas, muchas gracias.

Un agradecimiento especial a las personas del equipo *magnet* de INRIA Lille en Francia, que brindaron interesantes ideas para este trabajo en particular.

A mi profesora guía, que me acompañó y ayudó en este proceso, y a quien admiro mucho.

Y por último también, a todos quienes se acercaron desinteresadamente a este trabajo, y que contribuyeron de una u otra forma, muchas gracias.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Índice de Algoritmos

Índice de Códigos Fuente

Capítulo 1

Introducción

Las redes sociales juegan un papel relevante en la sociedad chilena. Medios de comunicación independientes y tradicionales las usan para difundir contenido en forma masiva. Según la encuesta Cadem de septiembre del 2020 [14] existen 15 millones de usuarios chilenos activos en redes sociales. El estudio añade que pese a la pandemia, la mayoría de la gente sigue informándose en la mañana al despertar (46 %). Durante esta hora del día el 71 % se informa por WhatsApp[1], 55 % Televisión abierta, 55 % Facebook[2], y 46 % diarios impresos o portales. Sin embargo, los sistemas de recomendación de las redes sociales provocan que los usuarios reciban información desde medios similares, generando una burbuja informativa que no deja cabida a un contraste de la información entre los medios. Este problema se conoce internacionalmente como *filter bubble* [71] y conlleva una consecuencia evidente: la falta de exposición a otros puntos de vista.

Por otro lado, día a día las personas se enfrentan a una constante sobrecarga de información (*information overload* en inglés) a través de redes sociales, televisión, radio, portales digitales, radios, etc [6]. Esto provoca no sólo problemas pragmáticos, como decidir dónde prestar atención, evitar la redundancia u ordenar la información, sino que también provoca problemas de salud mental [41] [45]. Medios como BioBioChile publican contenido tanto en sus portales de noticias[3] como en sus cuentas de twitter[4] [5]. El informador Chile[6], El Mostrador[7] y PiensaPrensa[8] siguen el mismo patrón.

El sesgo mediático se detecta en los medios de comunicación tanto en la *selección* de eventos que estos informan, como en la *forma* que presentan la información. Este puede ser ideológico, sensacionalista, publicitario, entre otros. Expertos lo identifican claramente a lo largo de la historia [54] y en Chile han identificado sesgos de racismo en la prensa chilena [70]. Casos extremos se traducen en faltas de ética [65] [27]. Es altamente relevante poder contrastar fuentes

[1] https://www.whatsapp.com/about
[2] https://about.fb.com/
[3] https://www.biobiochile.cl/
[4] https://about.twitter.com/
[5] https://twitter.com/biobio
[6] https://twitter.com/informadorchile
[7] https://twitter.com/elmostrador
[8] https://twitter.com/PiensaPrensa

de información para identificar sesgos mediáticos, y elaborar una postura crítica de la información, pero esto puede ser muy costoso en tiempo para una persona, considerando también la sobrecarga de información.

El trabajo *"Power structure in Chilean news media"* Bahamonde et al. [46] analiza la relación entre las entidades propietarias de los medios y su contenido/tópicos abordados. El análisis de redes revela que los medios chilenos están bastante concentrados en estos dos aspectos, provocando similitud de contenido publicado en medios con mismos propietarios. Medios a nivel nacional tienden a agruparse en su propia comunidad, mientras que medios locales se agrupan de acuerdo a zonas geográficas.

La concentración de la propiedad en los medios es un problema grave: acrecienta la falta de diversidad periodística. En cuanto a la legislación chilena sobre este tema, expertos señalan que es inadecuada [3]. Un caso extremo lo protagoniza Sinclair Broadcast Group, el locutor más grande de Estados Unidos con 193 estaciones televisivas, que ha sido acusado de ser históricamente sesgado políticamente, favoreciendo a los republicanos y últimamente de usar influencias para relajar las regulaciones sobre *media consolidation* (concentración de los medios) en la administración Trump. Además, busca por medio de la justicia, la adquisición de Tribune Media, otro gran locutor americano [48]. Hechos como este han provocado que sólo el 20 % de los estadounidenses tenga confianza en los medios televisivos y diarios [56].

Pero Chile no queda atrás y se suma a esta tendencia: junto a Carabineros, los medios de comunicación son la institución que más ha perdido la confianza: en 2009, 6 de cada 10 chilenos entre 18 y 29 años confiaba en los medios. En 2019, solo 1 de cada 10 lo hacía [91], incluso evidenciado en el movimiento "apaga la tv" durante el estallido social de 2019 [34].

Pero existen más antecedentes a considerar, como el auge de medios independientes en redes sociales [49], o el incremento de las *fake news*, con claros ejemplos en tiempos de pandemia [18] [69] [19]. Existe también un aumento en el volumen del discurso de odio [17], y en especial hacia minorías [31], y durante la campaña presidencial en Chile [84]. Además, Chile no queda fuera del fenómeno mundial de la *posverdad*, donde toda información, incluso hechos, son relativizados para cumplir distintos intereses [96][26].

Como último antecedente, buscar contenido histórico de medios nacionales en plataformas como Google o Google News [9] es insuficiente y altamente sesgado porque (1) existe un sesgo económico al mostrar resultados publicitados primero[10], (2) existe un sesgo de relevancia, y medios con menos recursos aparecerán con menos frecuencia[11], y (3) debido a los sistemas de recomendación y personalización de contenido, los resultados de búsqueda muestran medios distintos dependiendo del usuario que use la herramienta [87][71]. Esto incentiva el fenómeno de las burbujas informativas, tal como lo hacen redes sociales con sistemas de recomendación similares. Por otro lado, estas herramientas son poco prácticas para el análisis masivo de datos, debido a sus filtros limitados.

Todos estos antecedentes plantean la necesidad de contar con una herramienta que pueda mostrar las noticias nacionales abordadas por la mayor cantidad de medios posibles, facilitando

[9] https://news.google.com
[10] https://ads.google.com/home/
[11] https://www.google.com/search/howsearchworks/algorithms/

que el usuario pueda contrastar información e identificar los sesgos informáticos presentes. Sin embargo, no es suficiente mostrar todas las noticias de todos los medios, sino que es necesario agrupar la cobertura de cada medio para hechos noticiosos únicos, a modo de evitar la sobrecarga de información y facilitar el contraste de cada hecho.

Pero no solo esto, en los últimos diez años el área de Procesamiento del Lenguaje Natural[12] y los modelos de *machine learning* han tenido un crecimiento e impacto exponencial. La idea es sacar provecho de esta tecnología para generar métricas de los sesgos periodísticos a fin de ayudar a los usuarios a contrastar información y que puedan sacar sus propias conclusiones. Por ejemplo, medir si un texto es subjetivo u objetivo, o que tan probable es de ser discurso de odio, o qué tan positivo o negativo es, son algunas de las aplicaciones inmediatas y notables que pueden ser partícipes de la herramienta.

Esto también aumenta las posibilidades de la búsqueda de información, ya que mediante estas métricas es posible ver los sentimientos predominantes de los artículos en algún periodo, o de algún medio. Evaluar qué pasa con la subjetividad cuando los medios cambian de dueño, o de periodistas. Buscar los medios más neutrales y objetivos ante cierto evento, etc.

La idea es entonces desarrollar un *software* prototipo o *Proof of Concept*, que ayude a mejorar la calidad de la información mostrando todas las perspectivas en forma ordenada, y que junto a las métricas de sesgos periodísticos, se pueda encontrar información objetiva y relevante. El presente trabajo se inspira en una declaración de Eli Pariser, que menciona en una de sus TED Talks "Una democracia sustentable es aquella que cuenta con información de calidad"[72].

[12]https://www.forbes.com/sites/louisaxu/2021/12/01/a-golden-age-for-natural-language/

1.1. Objetivo general

Crear un sistema de recolección y búsqueda de contenido de medios digitales chilenos. La funcionalidad principal se enfoca en mostrar noticias de distintos medios respecto a un mismo hecho o tópico, para que los usuarios puedan contrastar perspectivas y opiniones. Se generarán y mostrarán métricas que permitan al usuario adoptar una postura crítica ante el sesgo periodístico presente. Con el espectro periodístico completo de medios tradicionales e independientes, se busca mitigar el efecto de burbujas informativas provocado por redes sociales.

1.2. Objetivos específicos

1. Crear un recolector de noticias, que con frecuencia diaria, extraiga contenido de los medios digitales nacionales tradicionales e independientes.
2. Diseñar e implementar una metodología que permita agrupar, por temas, las noticias que hablan de un mismo tópico.
3. Diseñar e implementar una metodología que permita generar métricas de sesgo periodístico.
4. Crear una aplicación web que permita realizar una búsqueda de noticias en el sistema, por tópico, fecha y otras métricas.
5. Desarrollar un sistema flexible y escalable, que permita incorporar nuevos medios de comunicación, nuevas formas de agrupar noticias y nuevas formas de medir sesgos periodísticos.
6. Evaluar cualitativamente el sistema, de manera general y específica.

Capítulo 2

Antecedentes

2.1. Agregadores de noticias actuales y sus limitaciones

2.1.1. Agregadores de Noticias

Existen servicios llamados *news aggregators* [95] o agregadores de noticias, que básicamente permiten a un usuario suscribirse a *feeds* de contenido de distintos medios. Un ejemplo de estos es Feedly[1], que busca evitar la sobrecarga de información dejando que el usuario escoja los *feeds* de contenido que quiere ver, y además presentándolos en forma ordenada para su consumo. Servicios similares son Feedbin[2], Inoreader[3] y Google News.

Sin embargo, estos servicios no son tan populares, ya que en plataformas como Twitter o Facebook se logra lo mismo *siguiendo* a los medios que cada usuario desee, generando un *feed* personalizado. Pero esto provoca el vicio de las burbujas informativas donde el usuario ve constantemente contenido similar. Además, es distinto a lo que se pretende hacer, puesto que no sólo se quieren agrupar los *feeds* de contenido de cada medio nacional, sino que juntar las versiones sobre una misma noticia de cada medio para poder contrastar el enfoque de cada cual.

2.1.2. Galean

El sitio Galean[4] detecta eventos a partir de *tweets* en Twitter utilizando una heurística de palabras claves en cada *tweet* y otras técnicas. De este modo, se pueden ver cómo distintos usuarios hablan sobre un mismo tema, y además permite obtener la geolocalización de los mismos [74].

Este sistema actúa sobre el público general de Twitter, y los problemas técnicos que resuelve

[1] https://feedly.com/i/welcome
[2] https://feedbin.com/
[3] https://www.inoreader.com/brand/
[4] http://galean.cl/

son similares a lo que se pretenden resolver con esta memoria. Sin embargo, la presente propuesta busca focalizar su estudio en los medios nacionales de comunicación (no en lo que usuarios comparten en redes sociales), creando una propuesta de valor en la información y el periodismo chileno.

2.1.3. Knowhere News

Knowhere News[5] es una compañía inglesa que busca brindar información sin sesgo informático a nivel global. Para ello se abastece de medios de comunicación globales, de los cuales extrae contenido y mediante técnicas de procesamiento de texto, genera un artículo lo más neutro posible, bajo sus estándares. Algunos de estos son validados por periodistas profesionales.

Esta es una iniciativa ejemplar, ya que demuestra que es posible obtener contenido de diversas fuentes en tiempo real y procesarla rápidamente. Sin embargo, este proyecto no sintoniza con el alcance de la propuesta de memoria, porque se quiere dar enfoque a los medios de comunicación nacionales y por ende al idioma español. Últimamente, ha integrado algunas métricas de análisis de sentimiento que sintonizan con el presente trabajo.

2.1.4. Swipe News

Swipe News[6] es un agregador de noticias del Reino Unido que recolecta distintas fuentes de información respecto de un hecho noticioso. Cuentan con 24 fuentes de información y 86 *feeds* de contenido de temas particulares. Por sus características es un proyecto bastante similar a lo que se pretende hacer.

Sin embargo, el sitio oficial está en blanco[7], su página de Facebook dejó de publicar el 9 de febrero de 2017[8] y el desarrollo hecho entre 2016 y 2017, sólo contaba con cinco personas. Estos antecedentes inducen a concluir que el proyecto está inactivo.

Aún si el proyecto estuviera activo, lo que se pretende hacer difiere de Swipe News por el carácter nacional que se quiere alcanzar, y porque se busca generar valor dejando un motor de búsqueda sobre el contenido histórico recolectado, enriquecido con las métricas de sesgos periodísticos.

2.1.5. Event Registry

Esta empresa fundada en 2017 y situada en Slovenia, es lo más similar al presente trabajo. Este proyecto recolecta artículos de noticias a nivel mundial, agrupa por eventos, y añade información extra a cada hecho noticioso. Destacando análisis de sentimientos, geolocalización y reacciones en redes sociales.

[5] https://knowherenews.com/
[6] https://sloboda-studio.com/work/swipenews/
[7] http://swipenews.com/
[8] https://www.Facebook.com/swipenews/

Ha ganado la mirada de Forbes dos veces, fue finalista para el premio *"Startup of the year"* en Slovenia el año 2018, y en 2019 ganó el premio a *"Top business idea"* dado por la revista *"Časnik Finance"* del mismo país. Su modelo de negocios se basa en la capacidad de analizar grandes volúmenes de datos, generando *insights* para empresas que buscan cómo mejorar sus negocios conociendo las reacciones sus clientes y de la competencia en la web.

Actualmente, soporta 57 idiomas, registra eventos de Chile, y es prueba que una iniciativa cómo se puede tener un gran impacto en la sociedad. Tiene una gran barrera de entrada, ya que muchas funciones son de pago y además es compleja de utilizar. La presente memoria busca transparencia, barrera de entrada bajas y que sea de simple uso. Ver Figuras A.1 y A.2 para una mayor referencia.

2.1.6. Disponibilidad de contenido periodístico

Se presume que cada medio de comunicación nacional almacena su contenido de manera privada, ya que no existe una clara disponibilidad del mismo al público en formato de base de datos. Tampoco se han encontrado entidades que ofrezcan bases de datos de este tema en forma comercial. En forma no comercial se encuentra la Biblioteca Nacional Digital de Chile[9], que bajo la Ley de Prensa [25] busca llevar un registro de los medios escritos, grabaciones sonoras y publicaciones electrónicas del país. Sin embargo, su contenido no es fácilmente accesible públicamente y, en general, no cumple el mismo objetivo de la presente memoria.

Como el acceso al contenido de los medios de comunicación nacionales no está disponible en forma ordenada, se hace importante crear un sistema que permita buscar información, y contrastar diferentes perspectivas mediáticas. De esta manera, la ciudadanía podría realizar análisis de medios, investigación y aplicaciones aún inimaginables.

2.2. Procesamiento del lenguaje natural

Los últimos diez años deslumbran el comienzo de una era dorada para el área del procesamiento del lenguaje natural (PLN por sus siglas en español), definida como el conjunto de metodologías que estudian la interacción entre el lenguaje humano y los computadores. El impacto de esta disciplina se encuentra en el diario vivir: traducción automática, generación de texto, respuesta automática de preguntas y resumidores son algunas de las capacidades del PLN que han permitido a grandes empresas como Apple, Amazon, Netflix y Google el desarrollo de Siri[10] y Alexa[11], *chatbots*, buscadores enriquecidos, sistemas de recomendación, autocompletado de *emails*, entre muchas otras cosas.

Muchas aplicaciones actuales del PLN se apoyan en técnicas de *machine learning* (ML por sus siglas en inglés), que en palabras simples, permiten construir programas complejos a partir de muchos ejemplos. Sin embargo, PLN es una disciplina bastante amplia, relacionada

[9] http://www.bibliotecanacionaldigital.gob.cl
[10] https://www.apple.com/siri/
[11] https://developer.amazon.com/alexa

también a la computación lingüistica, la inteligencia artificial, las ciencias de la computación, el procesamiento de audio, entre otras [32]. Gracias al incremento en el poder computacional y la disponibilidad de datos en internet, esta área ha logrado derribar múltiples barreras que hace treinta años eran impensables.

El año 2017, el área PLN fue revolucionada por *transformers*: una arquitectura ML de redes neuronales que permite a los desarrolladores construir aplicaciones sobre modelos pre-entrenados por otros, en vez de comenzar desde cero cada vez [92]. Además, el paralelismo permite tiempos de entrenamiento más cortos, resultando en modelos de mejor calidad y precisión. De *transformers* derivan las arquitecturas *Generative Pre-Trained Transformer* (GPT) [78] y *Bidirectional Encoder Representations from Transformers* (BERT) [30] que fueron clave para la explosión PLN que se ve hoy en día.

Hugging Face[12] es una compañía R&D que permite usar los últimos y mejores modelos de ML relacionados al PLN en forma gratuita, lo que ha provocado un efecto positivo en los años recientes, a nivel de usuario y empresas. Dado que su librería es *open source*, presenta al mundo una oportunidad de desarrollo abierto y colaborativo.

En el contexto de este trabajo se pretende usar los mejores modelos ML relacionados al PLN para obtener métricas respecto de los sesgos periodísticos. Si bien estos modelos no son perfectos, la idea clave es generar un sistema lo suficientemente flexible para cambiar o añadir modelos a medida que estos mejoran con el tiempo.

Medir sesgos periodísticos entrega un valor agregado que permite que los usuarios tengan una predisposición más crítica respecto a la información que reciben. Los enfoques tradicionales en esta materia miden, por ejemplo, el número de veces que los medios añaden una cuña de un hombre o mujer, para medir el sesgo de género. O las veces que los medios utilizan cuñas de políticos de izquierda o derecha, para medir el sesgo político [40][39][5].

Este trabajo brinda un enfoque distinto, la idea es utilizar técnicas PLN para medir la *subjetividad* de un texto, y la *polaridad* del mismo. De esta forma se pueden separar noticiarios informales de aquellos formales, y por otro lado, contrastar la forma en que los noticiarios abordan una noticia (positiva, neutral o negativa).

Si bien esto no detecta sesgos políticos, religiosos o racistas propiamente tal, representa un primer paso en el proceso de generar métricas respecto a este tema a modo general. La idea es que el sistema a desarrollar sea lo suficientemente flexible y escalable para soportar nuevas técnicas de análisis de texto, modelos de ML y heurísticas.

Además, existen modelos clasificadores de discurso de odio [85][4], de tendencia política [66][93][36], predicción de *fake news* [94][43][13] y de sentimientos (felicidad, rabia, terror, etc.) [59][97] que pueden usarse en el sistema, pero que quedan fuera para acotar el presente trabajo, y porque es necesario analizar en profundidad qué filtro puede ser más o menos útil para los propósitos de crítica y contraste. Esto, sin perjuicio que en un futuro puedan incorporarse.

Finalmente, en miras de ahorrar tiempo a los usuarios, se planea usar modelos PLN resumidores [62][2][58][99][55] (*summarizers* en inglés) que permitan extraer los puntos claves de cada

[12]https://huggingface.co/huggingface

hecho noticioso y utilizarlo para representar un evento. Al igual que en el caso anterior, este resumen de información está sujeto a error y sólo debe emplearse como referencia, sin embargo, es altamente probable que exista un modelo resumidor más fiable en un futuro próximo, porque la información de los noticiarios está altamente estructurada semánticamente.

Con todo esto, en conjunto, se brindan las herramientas necesarias para que los usuarios puedan ordenar la información de acuerdo al criterio que les sea más importante, por ejemplo, de menos a más subjetividad y de más a menos neutralidad. De esta forma pueden encontrar información en forma rápida, pero también les permite contrastar información y sacar conclusiones propias de las predicciones de los modelos en el sistema.

A modo de transparencia e interés, se describe en la siguiente sección el mecanismo mediante el cual distintos modelos *estados del arte* logran tan buenos resultados en las tareas PLN descritas. El eje común de todos estos modelos es la arquitectura *transformers* que se describe a continuación.

2.2.1. *Transformers*

Aspectos generales

En general, toda tarea que busque resolver una tarea PLN mediante técnicas de ML considera tres etapas principales: (1) pre-entrenamiento o generación de representaciones de palabras de un lenguaje, (2) elección de una arquitectura de redes neuronales para entrenamiento, y (3) aplicación y evaluación de la tarea PLN aprendida [98]. La Figura 2.1 ilustra este proceso.

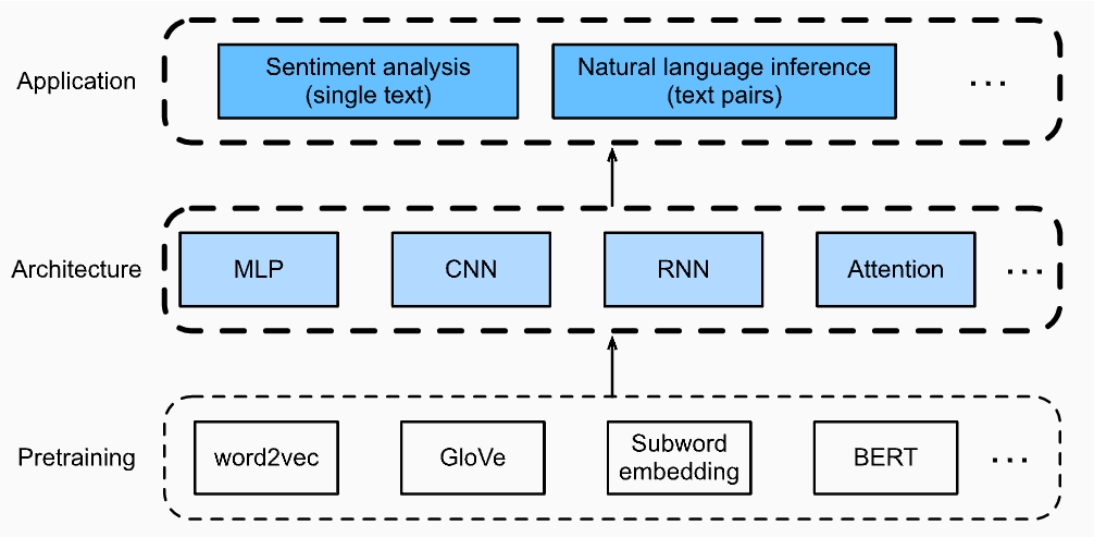

Figura 2.1: Representaciones de texto pre-entradas sirven para varias arquitecturas de *deep learning*, para diferentes tareas PLN [98]

En primer lugar, una representación correcta de las palabras de un lenguaje es fundamental para modelos ML de propósito general. Modelos libres de contexto como Word2Vec [67] o GloVe [75], generan un sólo *embedding* (vector numérico) para cada palabra del vocabulario. Por

ejemplo, "*bank*" tendrá la misma representación que "*bank deposit*" y "*riverbank*". En contraste, los modelos contextuales generan una representación para cada palabra que tiene en consideración todas las demás palabras en una oración. El modelo contextual de redes neuronales densas BERT [30], captura estas relaciones en forma bidireccional.

BERT, mediante una arquitectura tipo *transformer*, un vocabulario de aproximadamente 30 mil palabras (en inglés), y grandes corpus de texto, es pre-entrenado para resolver dos tareas fundamentales que permiten obtener representaciones contextualizadas de las palabras: *Mask Language Model* (MLM) [12] y *Next Sentence Prediction* (NSP) [11]. Las representaciones obtenidas son de propósito general al igual que Word2Vec o GloVe, pero con la gran diferencia de que estas representaciones son contextualizadas [20].

En segundo lugar, es posible combinar distintas arquitecturas de redes neuronales para resolver distintas tareas PLN. Algunas de estas son Multilayer Perceptron (MLP) [38], Convolutional Neural Network (CNN) [1], Recurrent Neural Network (RNN) [68], Attention [35], etc.

En tercer lugar, considerando una representación del lenguaje, y una arquitectura de redes neuronales, es posible entrenar a una red para resolver una tarea PLN determinada. Por ejemplo, se puede afrontar la tarea PLN de clasificación de sentimientos utilizando representaciones GloVe y una arquitectura RNN[13] [98].

Sin embargo, elaborar representaciones y arquitecturas para cada tarea PLN es prácticamente inviable, y en este punto, las representaciones elaboradas por BERT representan una gran ventaja. Primero, porque las representaciones son contextuales y segundo, porque basta añadir una red neuronal al final de esta arquitectura (una red MLP por ejemplo), para poder resolver distintas tareas PLN [98]. Ver Figura 2.2.

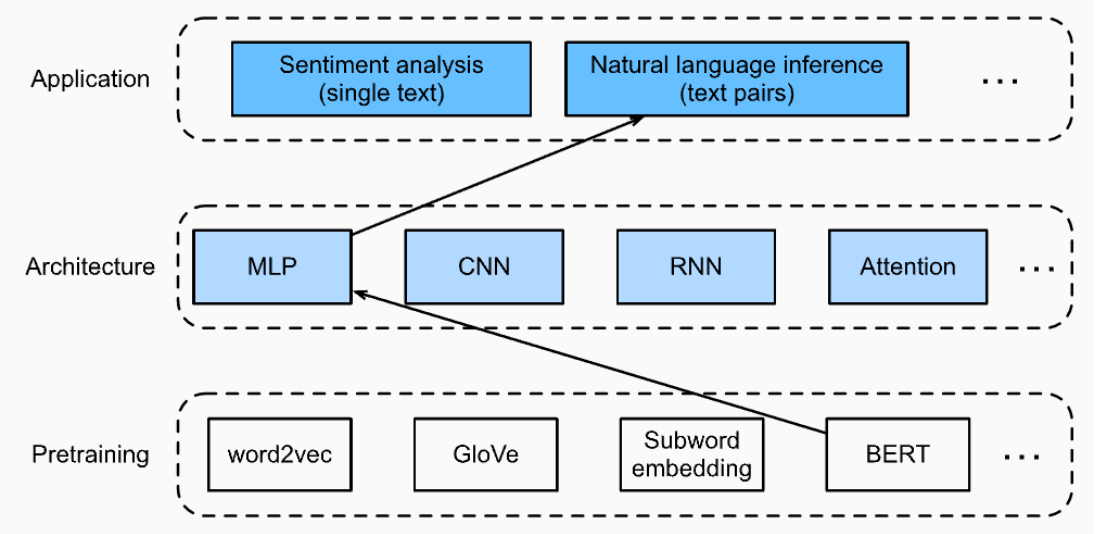

Figura 2.2: Uso de red pre-entrenada BERT junto a una arquitectura MLP para la tarea PLN de inferencia de lenguaje [98]

[13]http://d2l.ai/chapter_natural-language-processing-applications/sentiment-analysis-rnn.html

Este proceso se denomina *transfer learning*, ya que a modo general, se pretende utilizar el conocimiento de un modelo entrenado para las tareas MLM y NSP, y ajustarlo para resolver otras tareas PLN de clasificación. En este caso, al tratarse de modelos de ML, el proceso se denomina *fine-tuning*. Ver Figura 2.3.

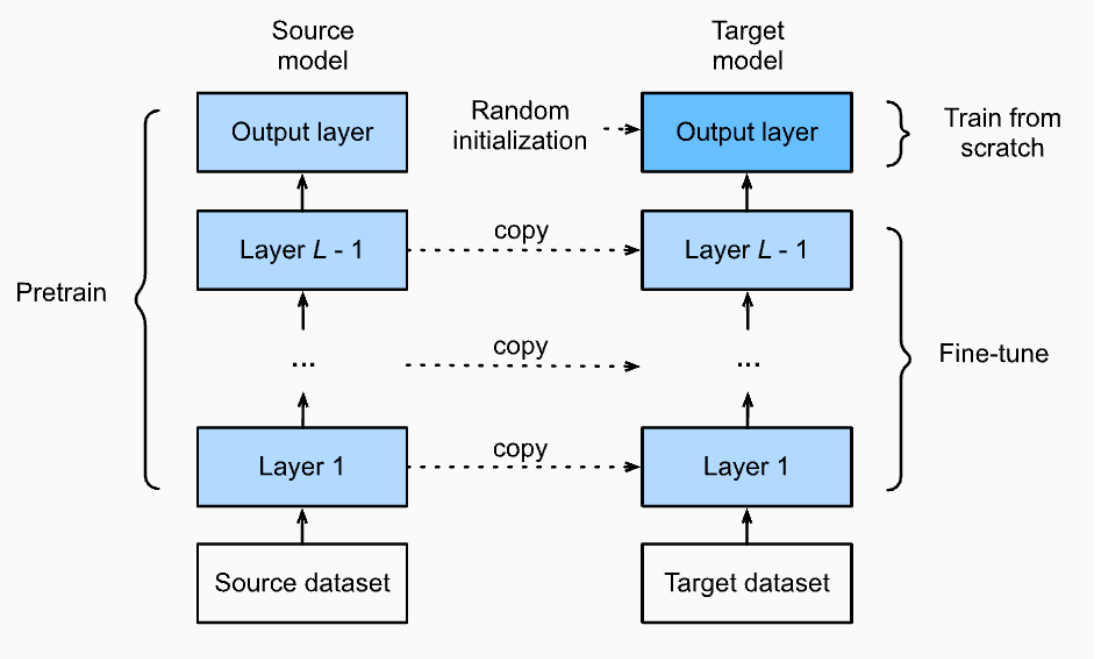

Figura 2.3: Proceso de *fine-tuning* [98]

La gran ventaja de este método es el bajo coste computacional. En menos de 30 minutos, se puede utilizar un modelo pre-entrenado BERT, entrenar una red neuronal de salida, y resolver una tarea PLN determinada. El proceso demoroso y demandador de recursos es el pre-entrenamiento (para MLM y NSP), pero no es necesario hacerlo, ya que existe una disponibilidad pública de modelos pre-entrenados. Es justamente esta accesibilidad la que hizo de BERT (y modelos derivados) una revolución en el contexto PLN.

No tan sólo eso, sino que varias metodologías basadas en el proceso de *fine tuning* de BERT han alcanzado resultados *estado del arte* en distintas tareas PLN [29]. Por esto, los modelos utilizados en este trabajo son principalmente de este tipo. A continuación se explica en detalle la arquitectura de BERT, y de *transformers* en general.

Arquitectura en detalle

La arquitectura *transformer* consta de dos procesos principales: *encoding* y *decoding*, ver Figura 2.4. El primero está encargado de generar *embeddings* contextuales de cada palabra del *input*, capturando la semántica de una frase. El segundo proceso se encarga de tomar estos *embeddings* y producir un *output* (secuencia de texto) para una tarea específica. Por ejemplo, para la tarea de *machine translation* [42] entre inglés y español, el *encoder* se encarga de aprender la semántica del idioma inglés mientras que el *decode* se encarga de aprender la correlación entre las palabras del inglés y del español [77][21].

Existen modelos basados únicamente en *encoders*, únicamente en *decoders* y también algunos que utilizan *encoders* y *decoders*. Modelos del primer tipo se usan generalmente para clasificar texto, reconocer entidades y la tarea *extractive question answering*. Modelos del segundo tipo se usan exclusivamente para tareas generativas de texto. Y por último, modelos del tercer tipo se emplean para tareas más complejas como *summarization*, *machine translation* y *generative question answering* [22].

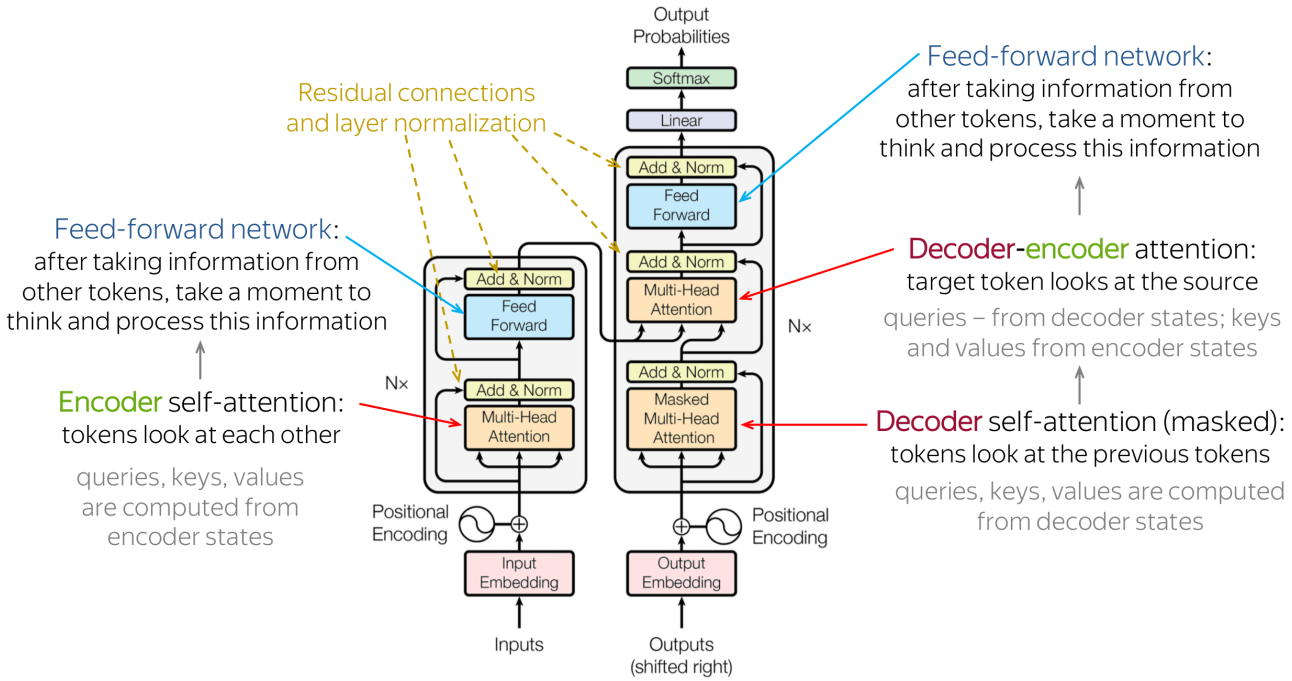

Figura 2.4: Arquitectura *transformer* [92]

Encoders

BERT no es más que un *stack* (conjunto secuencial) de *encoders* tipo *transformers*. Esta arquitectura permite que, en teoría, cada *encoder* aprenda una propiedad distinta del lenguaje y que al funcionar en conjunto, genere *embeddings* que capturen el contexto y la semántica de una frase. Por otro lado, un *stack* de decoders se encuentra en el modelo GPT [78], especializado en la tarea de generación de texto.

En un *encoder*, se comienza con el proceso de *tokenization*, que genera *embeddings* contextuales para cada palabra del texto en dos procesos: *input embedding* y *positional encoding*. En el primero, el texto plano es transformado a números. Se agregan los tokens [CLS] y [SEP] al inicio y fin de la frase. Por ejemplo, "How are you" queda como "[CLS] How are you [SEP]". Estos tokens son necesarios para delimitar el inicio y fin de una frase. Luego se asigna un *input embedding* definido en la fase de pre-entrenamiento a cada palabra del *input*, y por lo tanto, se generan tantos vectores como palabras del *input*. Palabras similares tendrán en consecuencia *input embeddings* similares.

Luego, el proceso *positional encoding* añade a cada *input embedding* información sobre la posición relativa de cada palabra en la frase. Para esto se generan *positional embeddings* para cada palabra, utilizando propiedades matemáticas, que luego se suman a los *input embeddings*.

Posteriormente, se pasa al proceso de *encoding* propiamente tal, donde destacan dos procesos: *multi-head attention* y *feed-forward*.

Multi-head attention es un proceso que repite un proceso denominado *self-attention* que básicamente, rescata la importancia de cada palabra del *input* con respecto a las demás. Por ejemplo, para la frase *"How are you"*, se puede aprender que esta estructura es generalmente una pregunta, o que generalmente va a acompañada de la palabra *"you"*. Múltiples capas de *self-attention* son dispuestas juntas, creando el proceso de *multi-head attention*. El principio teórico es que cada capa de *self-attention* aprende una propiedad distinta del lenguaje, y por lo tanto, el *output* tiene más información. Por su parte, *feed-forward* consta de tres capas lineales de redes neuronales totalmente conectadas que procesan la información nuevamente, a fin de generar *embeddings* más enriquecidos.

Finalmente, se producen los *embeddings* para cada palabra del *input* proporcionado originalmente, incluyendo *embeddings* para los tokens [CLS] y [SEP]. Notar que el proceso contiene mucho más detalles no mencionados, como procesos de normalización, capas de activación, conexiones residuales, entre otros.

BERT entonces, utiliza este proceso de *encoding* múltiples veces para generar *embeddings* aún más enriquecidos. En sus dos variantes, *BERT Base* usa 12 capas de *encoders*, con alrededor de 110 millones de parámetros, mientras que *BERT Large* emplea 24 capas de *encoders*, con cerca de 340 millones de parámetros. En general, *BERT Large* genera mejores *embeddings*, que a la vez mejoran los resultados en distintas tareas PLN, pero su uso consume más recursos computacionales.

Decoders

Los *decoders* tienen la principal tarea de generar texto. En esencia, usan los mismos procesos de *multi-head attetntion* y *feed-fowards* de los *encoders*, pero con ciertas modificaciones.

En cada iteración, un *encoder* toma como *input* el *output* que este mismo produce a modo de tener el contexto que va generando en cada paso. Este *input* pasa por el mismo proceso de *tokenization* (*input embedding* y *positional encoding*) descrito anteriormente.

Luego, estos *embeddings* pasan por dos capas consecutivas de *multi-head attention*, con la diferencia de que la relación contextual de las palabras está limitada a las palabras ya generadas anteriormente. Esto se logra mediante el uso de máscaras.

Luego se pasa por una capa de *feed-forward*, para luego terminar con una capa lineal, junto a una capa *softmax* [37], tan grande como el vocabulario. Estas últimas capas serán las encargadas de decidir qué palabra generar a continuación. Este *output* es incorporado al resultado final, que abastece nuevamente al *decoder* para generar la palabra siguiente. El proceso termina cuando el *decoder* escoge la palabra [END].

2.2.2. Uso práctico

En la práctica, se utilizan modelos de Hugging Face de tipo *transformers* pre-entrenados y ajustados (*fine-tuned* en inglés) por otros para tareas específicas. Tal es el punto, que bastan menos de diez líneas de código para usar cualquier modelo de dominio público, generalmente disponibles en Hugging Face. La calidad de estos modelos es generalmente la mejor posible, ya que compañías como Microsoft, Google y Facebook AI ponen a disposición sus modelos para uso público.

En particular, para las tareas de subjetividad y polaridad se necesita de un clasificador de texto. Como se describe antes, para esto son necesarios modelos basados en *stacks* de *encoders*, y por lo tanto, modelos como BERT [30], RoBERTa [59], ALBERT [53] o DistilBERT [83] son adecuados para esta tarea.

2.3. *Clustering*

2.3.1. *Embeddings* de texto

Clustering es el proceso de agrupar objetos similares en diferentes grupos, o más precisamente, la partición de datos en subconjuntos, de forma tal que cada subconjunto queda definido mediante una medida de distancia o similitud [60].

Para agrupar noticias semánticamente equivalente de manera tradicional, se deben abordar tres ejes principales. El primero de ellos, consiste en representar texto mediante *embeddings* (vectores numéricos). En el segundo eje, se calcula la *distancia* entre las abstracciones de texto mediante una métrica. Y por último, con los cálculos de distancia entre textos, se necesita ejecutar un algoritmo de *clustering* que agrupe los textos similares.

Existen múltiples maneras de generar *embeddings* a partir de texto, pero se destacan dos en el último tiempo. La primera de ellas es mediante el empleo de Word2Vec [67] o GloVe [75] (ya presentados anteriormente). Con estas estrategias, a cada palabra del texto se asigna un *embedding* predefinido. Para representar una oración, generalmente se realiza una *suma normalizada* de los *embeddings* de cada palabra. Para saber qué *embedding* asignar a cada palabra, se entrena un modelo de redes neuronales que aprende asociaciones de palabras y la semántica de un lenguaje mediante grandes volúmenes de texto.

Una segunda alternativa es utilizar *embeddings* de modelos basados en *stacks* de *encoders* tipo *transformers*, como BERT [30] o RoBERTa [59] (presentados anteriormente). El problema con estas arquitecturas es que el *output* corresponde a un *embedding* por palabra (más el de los tokens [CLS] y [SEP]). ¿Cómo se define entonces el *embedding* de una oración completa? Algunas técnicas consideran el *embedding* del token [CLS] como representativo de la oración (técnica CLS), otras usan un promedio de todos los *embeddings* (técnica MEAN), y otras rescatan los máximos de cada dimensión de los *embeddings* (técnica MAX).

Sin embargo, ninguna de estas alternativas ha dado buenos resultados para *datasets* de la tarea PLN de *semantic textual similarity* (STS por sus siglas en inglés) que mide la equivalencia semántica de dos frases [61]. Por otro lado, hay técnicas que entrenan a redes tipo BERT, que reciben dos oraciones como *input*, y retornan la similitud semántica entre ambas oraciones (dígase un valor entre 0 y 1, siendo 1 totalmente similar). Sin embargo, esta última estrategia consume cantidades enormes de recursos computacionales. Por ejemplo, encontrar las oraciones más similares en un conjunto de 10.000 ejemplos, tomaría hasta 65 horas en una GPU V100 moderna [80].

La publicación *"Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks"* Reimers et al. [80], brinda una solución fundamental a este problema. Se basa en las mismas técnicas CLS, MEAN y MAX, para entrenar una capa de salida, denominada *pooling*, que brinda un *embedding* único para cada oración. Mediante el uso de redes siamesas y una función objetivo, es capaz de entrenar esta última capa, generando *sentence embeddings* que bajo la evaluación de distintos *datasets* STS (usando SentEval[14]), logra un *spearman score estado del arte* superior a 88. En contraste a técnicas como MEAN GloVe que alcanzan un *score* de 58, o MEAN BERT que da un *score* de 46.

Lo más destacable de esta metodología, es que el tiempo de procesamiento baja considerablemente. La misma tarea de encontrar las oraciones más similares en un conjunto de 10.000, tardaría ahora 5 segundos con la misma GPU V100 [80]. Por lo tanto, realizar *clustering* sobre *sentence embeddings* se hace factible también. Más aún, el *spearman score* alcanzado evidencia la calidad de los *embeddings*, y por lo mismo la alternativa de utilizar Word2Vec o GloVe deja de ser atractiva.

Por si fuera poco, los autores de esta publicación han desarrollado una librería en Python que permite el uso de esta metodología en forma bastante sencilla[15]. En ella se encuentran modelos basados en BERT, RoBERTa o AlBERTa pre-entrenados y listos para usar.

2.3.2. Métricas y algoritmos

Por otro lado, existen muchísimas métricas de distancia que pueden servir para comparar el grado de separación entre dos *embeddings*. De manera convencional, se usa la distancia euclidiana o coseno, sin embargo, las distancias Minkowski, Manhattan o Mahalanobis pueden ser perfectamente buenas candidatas también[16]. Lo clave, es que emplear una u otra puede cambiar los resultados de *clustering* [52].

También existen muchos algoritmos de *clustering*; pueden ser jerárquicos o particioneros. Los del primer tipo buscan encontrar *clusters* utilizando parámetros predefinidos, mientras que los del segundo tipo determinan los *clusters* al inicio y realizan ajustes. Los algoritmos jerárquicos pueden ser aglomerativos, que comienzan con cada elemento como un *cluster* que mezclan para formar *clusters* más grandes, o divisores, donde todo el conjunto parte como un gran *cluster* que se divide sucesivamente en *clusters* más pequeños [60].

[14]https://github.com/facebookresearch/SentEval
[15]https://www.sbert.net
[16]https://scikit-learn.org/stable/modules/metrics.html

Un algoritmo particioneros es KMeans, mientras que *Agglomerative hierarchical clustering*, DBSCAN, OPTICS, y *Spectral clustering* son algoritmos jerárquicos[17]. Cada uno tiene distintos hiperparámetros ajustables a distintos casos de uso.

A priori, cualquiera de estos algoritmos puede ser útil, al igual que las métricas existentes. El desafío reside en encontrar la mejor combinación de algoritmo, métrica e hiperparámetros, que brinde resultados consistentes. ¿Qué define un resultado como consistente en una agrupación no supervisada?, esto se discute a continuación.

2.3.3. Criterios de evaluación de resultados de *clustering*

Lo más complicado del proceso de agrupación de noticias, es el criterio de evaluación y selección de resultados de *clustering*. Si se ejecutan varias combinaciones de hiperparámetros y algoritmos, no es directo escoger el mejor resultado, debido a la naturaleza inherente del problema, donde no existe *ground-truth*.

El problema también contempla que el *input* cambia cada vez, lo cual representa un gran obstáculo para métodos tradicionales de *clustering*, ya que no son parametrizables ante este escenario: no utilizan los mismos hiperparámetros en cada agrupación.

Existen métricas internas para evaluar resultados de *clustering* no supervisados, como el *Silhouette score* [82] o el *Calinski score* [15]. Si bien, dan indicios de la calidad de los *clusters*, por ejemplo, en base a la densidad de los *clusters* o la distancia de separación entre *clusters*, al final no garantizan que en la práctica los resultados sean buenos o malos. Sin embargo, estas métricas sirven para estimar hiperparámetros. Por ejemplo, sirven para estimar el número de *clusters* para al algoritmo KMeans [90].

De hecho, el *pull request* #6948 de la librería *scikit-learn*[18] plantea el uso de distintas metodologías que permiten escoger el mejor número de *clusters* para el algoritmo KMeans. Para esto utiliza, no sólo los criterios del *Silhouette* o *Calinski score* propuestos inicialmente, sino que tres criterios de elección no supervisada adicionales: *gap*[19], *stability* [10] y *pham* [76].

El autor implementa una clase denominada *OptimalNClusterSearch*, donde se especifican los criterios de selección y los valores del parámetro de búsqueda (*n_clusters*). Luego ejecuta *KMeans* con múltiples valores de *n_clusters*, y mediante el criterio especificado, elige cuál de todas las agrupaciones realizadas es la mejor. Por ejemplo, para el criterio *Silhouette*, elige la agrupación con un mayor valor del *Silhouette score*.

Esto representa una base a tener en consideración, para elaborar criterios de selección de resultados de *clustering*, en el contexto del problema de agrupación de noticias.

[17] https://scikit-learn.org/stable/modules/clustering.html
[18] https://github.com/scikit-learn/scikit-learn/pull/6948
[19] https://towardsdatascience.com/k-means-clustering-and-the-gap-statistics-4c5d414acd29

2.4. Base de datos de texto

2.4.1. Tecnologías existentes

El presente trabajo enfrenta un problema denominado *information retrival* [86], que plantea el desafío de obtener información en forma eficiente a partir de grandes volúmenes de datos. Como se aspira a ser una plataforma es masiva, es necesario contar con las mejores herramientas disponibles para almacenar y consultar información, en términos de flexibilidad, escalabilidad y eficiencia. Sobre todo cuando el problema que busca afrontar este sistema apunta justamente a la sobrecarga de información (*information overload* en inglés) [6].

En particular, una base de datos de texto es lo indicado para este sistema. Actualmente existen bases de datos con indexación inversa [33], que en vez de guardar la información en tablas, la guardan en estructuras complejas. Un índice invertido lista cada palabra que aparece en algún documento e identifica todos los documentos donde la palabra aparece. Esta propiedad permitirá que la plataforma cuente con un motor de búsqueda de palabras claves eficiente.

Dentro de las opciones más populares se encuentra Elasticsearch[20], herramienta distribuida, gratuita y abierta que permite la búsqueda de datos de texto, numéricos, geo-espaciales, estructurados y no estructurados. Está construido sobre Apache Lucene[21], utilizando los índices inversos de este *software*, pero que además implementa una REST API distribuida, rápida, escalable y simple de usar. Por otro lado, Elasticsearch ofrece una gama de herramientas para almacenar, analizar y visualizar datos. Logstash[22] para realizar *logging* y Kibana[23] como principal interfaz visual para buscar y visualizar datos.

Otra alternativa es Apache Solr[24], también construida sobre Apache Lucene, es una herramienta de búsqueda que ofrece casi las mismas funcionalidades que Elasticsearch. Sin embargo, Elasticsearch cuanta con una documentación más clara y simple de usar[25], ya que en el caso de Apache Lucene, esta no es amigable de leer y es difícil encontrar casos de uso sencillos[26]. Gracias a Kibana, Elasticsearch ofrece una forma sencilla de visualizar los documentos en la base de datos más allá de la consola.

Otra opción es ArangoDB[27], herramienta que almacena documentos en estructuras de grafos, es de uso libre y tiene un lenguaje de consulta propio. También se encuentra Vespa[28] desarrollada por Yahoo, menos popular pero totalmente suficiente. Finalmente, Algolia[29] se presenta como una base de datos en la nube, con servicios de búsqueda y recomendación de datos, pero de pago.

[20] https://www.elastic.co/what-is/elasticsearch
[21] https://lucene.apache.org/
[22] https://www.elastic.co/logstash/
[23] https://www.elastic.co/kibana/
[24] https://solr.apache.org
[25] https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html
[26] https://solr.apache.org/guide/8_11/
[27] https://www.arangodb.com
[28] https://vespa.ai
[29] https://www.algolia.com

Todas estas herramientas son suficientes para abordar el problema, pero al ser menos populares, se plantea el supuesto de que es más difícil solucionar problemas en el futuro, al tener una comunidad más reducida y menos activa en comparación a Elasticsearch o Solr. De esta manera, Elasticsearch se posiciona como una alternativa bastante balanceada en términos de calidad y comunidad.

Desde el punto de vista técnico, Elasticsearch funciona mediante el almacenamiento de documentos JSON, tal como lo haría una base de datos noSQL como MongoDB[30]. El modelo de datos de cada documento se denomina *mapping*, el cual define los atributos del documento. Estos atributos pueden ser de texto, numéricos, fechas, entre otros. Los documentos son almacenados en *index* que son configurados para tener el *mapping* adecuado. Las acciones CRUD de consulta, inserción, modificación y eliminación se realizan a través de la REST API de Elastichsearch, mediante consultas tipo GET, POST, PUT y DELETE en HTTP.

Se pueden hacer búsquedas sobre varios *index* a la vez, mediante el uso de expresiones regulares. Por ejemplo, se puede realizar una búsqueda sobre todos los *index* que empiecen con algún prefijo determinado. Lo ideal es no saturar los *index* con miles de documentos, ya que para la inserción de cada documento, se deben actualizar los índices inversos presentes en cada *index*. Esta forma de búsqueda permitirá tener múltiples *index*, datos más distribuidos, sin perder rendimiento en la búsqueda.

La búsqueda también permite agregar configuraciones avanzadas más allá de preguntar si una palabra está incluida en un documento. *Fuzzy query, Prefix query, Range query, Regex query, Terms query* y *Wildcard query* son algunas de las consultas que pueden hacer para buscar texto considerando palabras inexactas, que tengan cierto prefijo, que estén dentro de un rango numérico o de fecha, que satisfagan alguna expresión regular, que contengan algún término o que sigan un patrón definido, respectivamente[31].

Para utilizar la API de Elasticsearch desde Python se puede utilizar el cliente de Elasticsearch para Python[32], que permite, mediante el uso de funciones, abstraer las llamadas HTTP y simplificar el proceso.

2.4.2. Escalabilidad

Lo que se espera de una base de datos es que pueda escalar horizontalmente mediante la redundancia, pero que también lo haga mediante la distribución de datos. Si una base de datos está distribuida ente varios servidores, entonces es difícil que un solo servidor sea sobrecargado en algún minuto. Esto se denomina *sharding* y es súper importante en el contexto de base de datos escalables[33].

Elasticsearch permite distribuir el contenido de cada *index* en distintos nodos (servidores) mediante el uso de *shards*: fragmentos del *index* que unidos conforman la totalidad del mismo. De esta forma, las solicitudes son balanceadas naturalmente debido a la distribución de datos.

[30]https://www.mongodb.com/es
[31]https://www.elastic.co/guide/en/elasticsearch/reference/current/term-level-queries.html
[32]https://elasticsearch-py.readthedocs.io/en/v8.1.0/
[33]https://opster.com/blogs/elasticsearch-shards-and-replicas-getting-started-guide/

También existen las denominadas *replicas*, que son copias de los *shards* originales para generar redundancia y evitar pérdida de información. Cada *replica* esta alojada en un nodo distinto al *shard* original, ya que ante un fallo, esta copia pueda suplir la demanda.

El número de *shards* se debe especificar al momento de crear un *index*, pero el número de *replicas* puede ser modificado en cualquier momento. Esto permite tener tanta redundancia como sea necesario para evitar eventualidades colapsos de los *index*. Por otro lado, Elasticsearch se encarga de la comunicación entre *shards* y *replicas* para mantener consistencia, realizar búsquedas y todas las operaciones necesarias, lo cual es un beneficio al momento de desarrollar.

Un conjunto de nodos se denomina un *cluster*[34], lo cual se entiende como la totalidad de la base de datos. Cada *cluster* puede clonarse a sí mismo para satisfacer demanda en distintas regiones físicas, pero a la vez genera una carga en términos de mantención de la consistencia entre los *clusters* existentes.

Con todos estos mecanismos, Elasticsearch se convierte en un candidato perfecto para una aplicación a gran escala. Además, mediante el sistema de administración Kibana, se pueden ejecutar funciones de mantenimiento general y monitorear los nodos, *shards* y *replicas*.

Las otras alternativas cuentan con un proceso similar de escalabilidad, y por lo tanto, también son suficientes para afrontar este problema.

2.5. Arquitecturas de software

Para diseñar un sistema escalable y flexible, primero se deben conocer los requisitos técnicos asociados a la interacción del mismo. La cantidad de usuarios, la disponibilidad de servidores, el ancho de banda disponible, la latencia esperada, entre otras cosas.

La presente memoria busca ser en un futuro una aplicación masiva, con miles de usuarios activos al día que puedan informarse en forma enriquecida. Los tiempos de respuesta deben ser mínimos, tal como lo hacen las redes sociales hoy en día. Y para ello, eventualmente se debe contar con decenas de servidores a disposición: escenario que no es viable de momento.

Bajo este contexto, se necesita de una arquitectura escalable a miles de usuarios, pero que será acotada al contexto actual de trabajo: prototipo o *Proof of Concept*. La idea es que de ser necesario escalar y de contar con los recursos en un futuro, se pueda hacer sin problemas y con un lineamiento ya predefinido.

Utilizar una arquitectura monolítica, donde toda la aplicación está contenida en un sólo proceso, tiene la ventaja de poder escalar horizontalmente muy fácilmente. Sin embargo, en el largo plazo representa grandes obstáculos al momento de añadir nuevas funcionalidades, manejar grandes volúmenes de datos y una serie de motivos más[35].

Por otro lado, aplicaciones masivas suelen emplear arquitecturas basadas en microservicios, bajo el principio de que cada servicio tiene una sola responsabilidad de negocios. Esto hace que

[34]https://www.elastic.co/blog/multiple-elasticsearch-clusters
[35]https://microservices.io/patterns/monolithic.html

la aplicación sea modular y que pueda escalar en términos de funcionalidad y también horizontalmente. Además, al ser una aplicación modular, el código es altamente mantenible y fácil de testear. Cada microservicio permite hacer *deploy* en forma independiente y ser desarrollado por equipos de trabajo pequeños[36].

Dentro de las desventajas de esta arquitectura se encuentra la complejidad de la misma, ya que pueden haber solicitudes que abarquen varios microservicios, la comunicación entre microservicios es más difícil de testear, entre otras cosas.

Desde un punto de vista más general, se encuentran varios patrones de diseño de arquitecturas de *software*. Por ejemplo, el patrón basado en capas[37] (*multitier architecture* en inglés), generalmente divide la aplicación en capa de presentación, negocio, persistencia y base de datos. Si bien, permite separar las responsabilidades de negocios, no es escalable y una falla generalmente provoca fallas en cascada.

Un patrón cliente-servidor no es suficiente por los requisitos técnicos de la plataforma, debido a la complejidad de los procesos. Por otro lado, el patrón Modelo Vista Controlador, similar al patrón basado en capas, busca separar las responsabilidades del *software*. Conveniente de utilizar en servicios de *backend*, por ejemplo, Django[38] se basa en esta patrón.

Es importante también, tener en consideración la cohesión de componentes [64], para evitar desarrollar un software poco mantenible, propenso a fallas en cascada y altamente complejo.

Se busca entonces, un equilibrio que permita desarrollar una aplicación a gran escala desde un inicio, considerando los desafíos de escalabilidad presentes, pero ajustando estos parámetros al contexto actual del *software* como prototipo.

2.6. Oportunidad de desarrollo

Desde los negocios, a excepción de Event Registry, este nicho no ha sido explorado en el plano nacional, y por lo tanto, representa una oportunidad de desarrollo que vale la pena explorar. Esto justifica la motivación de seguir con un emprendimiento en un futuro.

Múltiples casos de uso se pueden encontrar en las áreas del periodismo, ciencias sociales, educación, humanidades, y también desde el punto de vista de los negocios, y todos convergen en lo mismo: el análisis masivos de datos. La idea es que esta herramienta ahorre tiempo y facilite la búsqueda de información tanto en forma diaria, como histórica.

Se plantea el desafío de encontrar una manera de desarrollo sustentable que junte a todas estas partes, sin olvidar el objetivo subyacente de este trabajo: facilitar el contraste de la información en Chile.

[36] https://microservices.io/patterns/microservices.html
[37] https://www.ibm.com/cloud/learn/three-tier-architecture
[38] https://www.djangoproject.com

Capítulo 3

Diseño de solución

3.1. Selección de medios de comunicación

En el escenario ideal, todos los medios nacionales son incluidos en el sistema, pero esto es imposible por un tema de recursos. Por lo tanto, la integración de medios al sistema debe ser gradual y bajo criterios de priorización.

Para elaborar una lista de priorización, se propone utilizar los siguientes criterios: popularidad e impacto, diversidad periodística y factibilidad técnica.

La popularidad e impacto de un medio es directa de obtener por el número de seguidores en redes sociales como Twitter, Instagram o Facebook. Por acotamiento de tiempo, se limitará considerar a Twitter como fuente confiable de esta información y se asumirá que para el resto de redes sociales el número de seguidores es proporcional a Twitter.

Existen también otras métricas para obtener el impacto de medios de comunicación, tales como *rating* televisivo, número de visitas en portales digitales, venta de diarios, entre otros. Sin embargo, no es sencillo obtener esta información y en consecuencia el desafío reside en encontrar las cuentas de Twitter de cada medio, para obtener únicamente el número de seguidores.

Para medir la diversidad periodística, se debe determinar si un medio es considerado tradicional o independiente, y tener en consideración si informa por otros canales de comunicación: TV, radio, diarios, redes sociales, etc.

La factibilidad técnica sólo se demostrará a sí misma, al momento de implementación. De no ser posible integrar un medio al sistema, simplemente se avanzará en la lista de priorización.

Finalmente, de existir, también se debe contar con los dominios de los portales de noticias de cada medio, para poder extraer información de esta fuente.

La definición de los medios a incluir, y la metodología final utilizada se detalla en la Sección 4.2.

3.2. Recopilación de noticias

3.2.1. Aspecto técnico

Aquellos medios que tienen portales de noticias en sitios web propios, generalmente tienen información estructurada: se emplea la misma plantilla de publicación una y otra vez para difundir noticias. En contraste a esto, hay medios que publican imágenes de sus diarios en sus portales (como Hoyxhoy[1] o La Segunda[2]), o bien, informan por Instagram (además que en sus portales) generando publicaciones con una imagen referencial junto a un titular, mientras que el cuerpo va en la descripción de la publicación (como Radio Usach[3] o Cooperativa CL[4]). Otros medios alternativos mezclan noticias con comentarios en cuentas de Twitter (como Chileokulto[5]) y algunos tienen portales de noticias, pero semi-abandonados (como Radio Villa Francia[6]).

Ante este escenario tan diverso se hace necesario acotar los medios según la forma en que difunden información. Claro está, que aquellos medios con información más estructurada son más fácil de extraer, por lo tanto, se priorizan medios que tengan su información estructurada en portales de noticias, bajo el argumento de factibilidad técnica.

Esto deja fuera a medios que informan por Instagram, o a aquellos que publican su diario como imagen en sus portales, ya que para procesar esa información es necesario, entre otras cosas, detectar texto en imágenes, tarea que no se ve fácil de abordar en poco tiempo. También el acceso a la API de Instagram[7], que es más complicada de obtener que la de Twitter[8].

De esta manera, se acota el presente trabajo a considerar medios que tengan portales de noticias actualizadas en forma periódica. Para esta tarea, el uso de *scrappers* es adecuado [63].

Los *scrappers* cumplen una tarea sencilla y repetitiva: se les provee de *urls* de sitios determinados, las visitan, extraen todas las *urls* presentes en el sitio visitado y luego hacen *parsing* la información recolectada en cada sitio.

De este modo, extraen información recursivamente de todas las *urls* que van encontrando en un sitio, emulando el comportamiento de un usuario que navega por un sitio en particular. La idea entonces, es desarrollar un *scrapper* por medio de comunicación que extraiga como mínimo la fecha, titular, bajada, cuerpo y autor de cada artículo que encuentra.

Herramientas de *scraping* existen varias y se discuten en la Sección 4.3. Sin embargo, queda un problema de diseño por resolver: definir las *urls* base (o semilla) a entregar a los *scrappers*.

Es directo utilizar el dominio base de cada medio para esta labor, pero se puede hacer algo mejor. Los medios generalmente publican en Twitter las *urls* hacia sus portales de noticias, y

[1] www.hoyxhoy.cl
[2] https://digital.lasegunda.com
[3] https://www.instagram.com/radiousach/
[4] https://www.instagram.com/cooperativa/
[5] www.twitter.com/Chileokulto
[6] www.radiovillafrancia.cl
[7] https://www.instagram.com/developer/
[8] https://developer.twitter.com

en consecuencia, se pueden tomar todas estas *urls* y usarlas de semilla para los *scrappers*. De esta manera, se garantiza que al menos todo el contenido que los medios publican en Twitter es integrado al sistema mediante los *scrappers*. Además pueden obtenerse métricas directas del impacto de una noticia con el número de *retweets* y *likes*, lo cual servirá como filtro posteriormente.

Para esto se debe descargar el muro de Twitter de cada medio, extraer las *urls* semilla y luego utilizarlas en los *scrappers*: con esto queda definida la solución para extraer información de los medios.

3.2.2. Aspecto legal

Una última observación es el aspecto legal de esta práctica. La ley 17.336 de Propiedad Intelectual, establece en el Título III: Limitaciones y Excepciones al Derecho de Autor y a los Derechos Conexos, lo siguiente:

Artículo 71 B. Es lícita la inclusión en una obra, sin remunerar ni obtener autorización del titular, de fragmentos breves de obra protegida, que haya sido lícitamente divulgada, y su inclusión se realice a título de cita o con fines de crítica, ilustración, enseñanza e investigación, siempre que se mencione su fuente, título y autor [44].

Artículo 71 O. Es lícita la reproducción provisional de una obra, sin que se requiera remunerar al titular ni obtener su autorización. Esta reproducción provisional deberá ser transitoria o accesoria; formar parte integrante y esencial de un proceso tecnológico, y tener como única finalidad la transmisión lícita en una red entre terceros por parte de un intermediario, o el uso lícito de una obra u otra materia protegida, que no tenga una significación económica independiente [44].

Artículo 71 Q. Es lícito el uso incidental y excepcional de una obra protegida con el propósito de crítica, comentario, caricatura, enseñanza, interés académico o de investigación, siempre que dicha utilización no constituya una explotación encubierta de la obra protegida. La excepción establecida en este artículo no es aplicable a obras audiovisuales de carácter documental [44].

En primer lugar, este trabajo está absolutamente alineado con el Artículo 71 B, ya que nunca se muestra la obra completa de un medio, sólo el titular, autor (medio) y la fuente (hipervínculo). Sumado a esto, el uso de esta información en la aplicación, es precisamente a modo de cita y con la finalidad de crítica, ilustración, enseñanza e investigación.

En segundo lugar, la aplicación es capaz de alinearse con el Artículo 71 O, porque la exposición de los fragmentos de las obras (título, medio e hipervínculo) puede ser transitoria de ser necesario en un futuro.

En tercer lugar, las finalidades de crítica, comentario, caricatura, enseñanza, interés académico o de investigación del Artículo 71 Q, coinciden también con las de la aplicación.

Teniendo todo esto en consideración, la indexación de contenido, y el uso exclusivo del título, medio e hipervínculo de artículos de noticias, parecen ser los límites legales presentes. El

propósito de la aplicación no es por ninguna circunstancia, la exposición completa y explícita de artículos. Por otro lado, la recopilación de bajada, cuerpo, fecha, etc, se pretende usar de modo acotado y momentáneo para poder abastecer el buscador, la agrupación de noticias y la generación de resúmenes.

De ser necesario en un futuro, por un tema legal, se debe idear una forma de mantener la funcionalidad del sistema, a costa de la eliminación de contenido potencialmente protegido en las bases de datos. Por ejemplo, el uso de técnicas de *topic modeling* [73] puede ser útil para extraer conceptos claves de los artículos, sin almacenar los artículos propiamente tal.

Finalmente, es prudente señalar que Google News, desde su lanzamiento en 2006, ha sido demandado en otros países[9][10] e incluso ha tenido que cerrar sus operaciones en España por temas económicos[11]. Pero en Chile, con 17 años en funcionamiento, jamás ha sido demandado. Esto podría deberse a que Google News representa más una ventaja, que una amenaza para los medios nacionales (por la exposición), y porque efectivamente el contenido no es expuesto de manera completa y explícita, ya que solo expone titulares, junto a su medio e hipervínculo. Sin embargo, de crecer el proyecto a una *startup*, se debe contar con la asesoría legal pertinente.

3.3. Sesgos periodísticos y otras funcionalidades

Una de las principales funcionalidades del sistema es la generación de métricas de sesgos periodísticos. Como se adelanta en la Sección 2.2, la presente propuesta plantea la utilización de modelos PLN basados en ML, y específicamente en *transformers*, para la predicción de subjetividad y polaridad en texto.

El objetivo es que mediante estas métricas, el usuario de la herramienta pueda ordenar la información de más a menos objetiva, y contrastar a los medios de acuerdo a cómo abordan una noticia. Por ejemplo, un mismo evento puede ser abordado de manera positiva por un medio, pero de forma negativa por otro.

Además de esto, en la Sección 2.2 se propone también utilizar modelos PLN resumidores de texto, que permitan identificar información relevante de un evento y mostrarlo en forma ordenada a los usuarios. Estos modelos tienen como *input* los cuerpos de las noticias de un evento, y el *output* que brindan es el resumen de la noticia.

A continuación se describe en detalle el diseño y propuesta final de cada una de estas tareas, junto a las limitaciones y consideraciones existentes.

3.3.1. Polaridad

Muchas veces denominado como análisis de sentimientos (*sentiment analysis* en inglés), la clasificación de polaridad consiste en predecir si un texto es positivo, neutral o negativo.

[9] https://tinyurl.com/3xyn6ej4
[10] https://www.reuters.com/article/us-google-afp-idUSN0728115420070407
[11] https://www.reuters.com/article/us-australia-media-google-idUSKBN26N0HJ

Oraciones que contienen palabras como "celebración", "éxito" o "bueno" tendrán tendencia a clasificarse como positivas, mientras que oraciones con palabras como "muerte", "horrible" o "decepción" tenderán a ser clasificadas como negativas.

Las técnicas de análisis de sentimiento se clasifican en aquellas basadas en *lexicons*, las basadas en ML e incluso algunas híbridas. Las primeras se basan en el uso de diccionarios, donde cada palabra tiene un valor de polaridad asignado, mientras que las segundas se basan en redes neuronales.

Recientemente, técnicas de *deep learning* como RoBERTa [59] y T5 [79] son usadas para entrenar clasificadores de sentimiento de alto rendimiento. Estos son evaluados mediante métricas como F1, *precision* y *recall*, sobre *datasets* como SST, GLUE e IMDB reviews[12].

Al igual que en el caso anterior, existen muy buenos modelos que generalmente superan el 97 % de precisión sobre varios *datasets*. Disponibles en forma de abierta en librerías como Hugging Face.

Dentro los clasificadores basados en *lexicons*, se encuentra SentiStrenght[13], gratuito para fines académicos y pagado para fines comerciales, y también VadeSentiment[14], de uso libre. Por otro lado, en Hugging Face hay cientos de modelos disponibles para esta tarea[15], y por lo tanto, serán considerados al momento de implementación.

3.3.2. Subjetividad

Los modelos de clasificación PLN de subjetividad tienen una tarea específica: clasificar texto en objetivo o subjetivo, y para esto tienen en especial consideración palabras que presuman estados de ánimo, emociones y sentimientos. Por ejemplo, la presencia de adjetivos calificativos indica la presencia de interpretación, como también lo hacen las oraciones escritas en primera persona.

En la práctica, los modelos de ML entrenados para esta tarea de clasificación se basan principalmente en el uso del *dataset Movie Review* (más conocido como SUBJ) desarrollado por académicos de la Cornell University, que contiene 5000 oraciones subjetivas y 5000 oraciones objetivas[16]. Para generarlo extrajeron *reviews* de películas desde Rotten tomatoes[17] consideradas como subjetivas, y resúmenes de películas desde IMDB[18], consideradas como objetivas.

El modelo *estado del arte* para esta tarea sobre este dataset alcanza un 97.34 % de precisión, demostrando la capacidad de modelos de ML de resolver esta tarea PLN con éxito (fuente Paperswithcode [19]). Además, éste y otros modelos suelen ser de código abierto, lo cual permite utilizarlos libremente.

[12]https://paperswithcode.com/task/sentiment-analysis
[13]http://sentistrength.wlv.ac.uk
[14]https://github.com/cjhutto/vaderSentiment
[15]https://huggingface.co/models?pipeline_tag=textclassification
[16]https://www.cs.cornell.edu/people/pabo/movie-review-data/
[17]https://www.rottentomatoes.com
[18]https://www.imdb.com
[19]https://paperswithcode.com/sota/subjectivity-analysis-on-subj

Hugging Face no tiene modelos disponibles para esta tarea. La alternativa es aprovechar los modelos encontrados en Paperswitcode, o bien, emplear modelos basados en *lexicons* ofrecidos por librerías como TextBlob[20]. También se puede tomar BERT-base[21] y hacer un proceso de ajuste (*fine-tuning*) para la tarea de clasificación de subjetividad, sobre el *dataset* SUBJ de uso público. Detalles de la implementación final se encuentran en la Sección 4.4.4.

3.3.3. Resumidor

Los modelos que generan resúmenes (*summaries* en inglés) tienen por objetivo producir una versión corta del documento entregado preservando la información importante de las oraciones más relevantes y representativas.

Los modelos *extractivos* son aquellos que identifican las partes fundamentales de un texto y la extraen sin modificación, mientras que los modelos *abstractivos* generan nuevo texto mediante una comprensión semántica del mismo. Una de las métricas para evaluar el rendimiento de un modelo resumidor se denomina ROUGE score (abreviación de *RecallOriented Uderstudy for Gisting Evaluation* en inglés). La idea base de esta métrica es comparar el resumen generado con una serie de resúmenes generados por humanos, por ejemplo, comparando el número de palabras solapadas. Como regla general, mientras mayor sea el ROUGE score de un modelo, mejor será el modelo [57].

Si el 2019 los mejores modelos lograban un ROUGE *score* alrededor de 35, hoy en día la mayoría de modelos *estado del arte* alcanzan un *score* sobre 45, sobre diversos *datasets*. Lo cual demuestra el gran avance en términos de investigación sobre estos modelos[22].

Hugging Face da la posibilidad de usar modelos del *estado del arte* para esta tarea, y por lo tanto, se pretende usar modelos que se puedan encontrar en esta librería[23].

La idea es generar resúmenes de eventos noticiosos a partir de la concatenación de bajada y cuerpo de todos los artículos que componen un evento noticioso, o en su defecto, si esto no es factible técnicamente, seleccionar aleatoriamente tres o cuatro artículos y producir el resumen a partir de ellos.

Por otro lado, nace una idea interesante: si para cada evento se generase mediante el modelo resumidor, un titular a partir de los titulares de los artículos que componen al evento ¿Es este nuevo titular más representativo del evento? ¿Logra captar todos los matices de los artículos? ¿Es más neutro que los artículos, o más objetivo? O también, si los demás titulares son sesgados ¿Se traspasa este sesgo al titular generado?.

Se establece la hipótesis de que, dado que los títulos de por sí engloban la idea central de una noticia, y que además son similares, el nuevo título será bastante representativo, o incluso puede ser un parafraseo de los titulares utilizados como *input*.

[20]https://textblob.readthedocs.io
[21]https://huggingface.co/bert-base-uncased
[22]https://paperswithcode.com/task/text-summarization
[23]https://huggingface.co/models?pipeline_tag=summarization

En cuanto al sesgo, se cree que si una noticia es negativa, como la muerte de alguien, entonces el titular generado seguirá siendo negativo, lógicamente, al componerse muy probablemente de las mismas palabras. Por lo tanto, no es esperable que cambie mucho en términos de neutralidad, a menos que omita cuñas o declaraciones de personas. Por otro lado, quizás es probable que la subjetividad baje en algunos casos, porque los modelos resumidores generalmente son entrenados con cuerpos de noticias o resúmenes de películas, que no hablan en primera persona y contienen menos adjetivos calificativos. De esta manera, la segunda hipótesis platea que es posible que los titulares generados sean menos subjetivos que los titulares usados como *input*.

3.3.4. Limitaciones y consideraciones

Hay que tener claro que todos los modelos con altos índices de precisión mencionados están implementados para procesar texto en inglés. Esto representa una gran limitación y es en general una limitación en el contexto PLN para diversos investigadores en todo el mundo.

Existen al menos tres formas de afrontar este problema: la primera consiste realizar el procedimiento de entrenamiento completo, desde la generación de *datasets* en español[24], hasta la búsqueda adecuada de hiperparámetros para obtener mejores rendimientos en tareas específicas. Esto considera contar con grandes corpus de texto en español y de recursos físicos para entrenar modelos, lo cual es prácticamente inviable debido al tiempo que demora efectuar todo este proceso tanto para subjetividad, como para polaridad, porque prácticamente representa un proceso completo de investigación.

La segunda consiste en buscar modelos de ML de subjetividad y polaridad ya entrenados para el español. Para polaridad existen varias alternativas en Hugging Face, con modelos multilinguales[25], mientras que para subjetividad no es posible encontrar siquiera una alternativa.

La tercera alternativa es traducir texto del español al inglés y luego hacer uso de los modelos mencionados. Esto conlleva la gran desventaja de que la calidad de los resultados de polaridad y subjetividad, está ahora sujeta además a la calidad de la traducción del texto. Por lo tanto, el procedimiento para traducir debe ser lo más preciso posible.

Por ejemplo, el titular "Doble chileno de Daddy Yankee echó al agua a Junior Fernandes y Ronnie Fernández: se fueron de carrete tras empate con Palestino"[26], es traducido por Google a "Daddy Yankee's Chilean double threw Junior Fernandes and Ronnie Fernández into the water: they went off the rails after drawing with Palestino". Notar que se pierde la semántica de las expresiones localistas *echó al agua* (delatar) y *carrete* (fiesta). Si esta frase en inglés, es traducida al español, queda "El doble chileno de Daddy Yankee tiró al agua a Junior Fernandes y Ronnie Fernández: se descarrilaron tras empatar con Palestino" demostrando que por traducir se pierde semántica en ambas direcciones (*carrete* termina siendo *descarrilar*).

Existen distintos servicios externos para llevar a cabo una traducción que incluyen a grandes compañías como Google y Microsoft[27], pero que tienen la limitación de ser de pago. Sin duda

[24]https://www.cloudfactory.com/data-annotation-tool-guide
[25]https://huggingface.co/models?language=es&pipeline_tag=text-classification
[26]https://www.theclinic.cl/2022/04/17/junior-fernandes-ronnie-fernandez-carrete-doble-daddy-yankee/
[27]https://www.microsoft.com/es-es/translator/

representan una buena alternativa a considerar por la calidad de los resultados que pueden tener, pero en el contexto actual es una solución inviable.

Otra alternativa es utilizar *machine translation*: modelos de ML especializados para traducir de un idioma a otro. En Hugging Face se encuentran disponibles tres modelos para traducir del español al inglés[28]. El más popular, proviene de un desafío hecho por la Universidad de Helsinki[29], y es el que tiene más un mayor *BLEU score*, métrica que se emplea para evaluar traducción de texto (más es mejor). La licencia del modelo es CC BY SA NC, por lo tanto, puede emplearse para investigación sin fines comerciales, y en especial para esta etapa.

Se considera entonces la opción de usar modelos multilinguales disponibles, o traducir texto al idioma inglés mediante modelos *estado del arte*. Sin embargo, para acotar el alcance de esta memoria, se decide emplear la segunda opción únicamente. Pero sin perjuicio de haber descartado la primera opción, porque el sistema debe ser lo suficientemente flexible para cambiar cualquier modelo de ML que lo compone.

Considerar que, aunque la traducción sea sensible a los modismos y expresiones locales, cualquier modelo basado en BERT también lo es. Esto es así, porque BERT consta de un proceso llamado *tokenization* (detallado en la Sección 2.2.1), donde asigna a cada palabra del *input*, un *embedding* (vector numérico) determinado. Si no encuentra la palabra, asigna un *embedding* equivalente a "desconocido". El conjunto de palabras que identifica, está directamente relacionado con el *dataset* que entrena el modelo, por ello, si el modelo no es entrenado con un *corpus* de texto que contenga modismos locales, la mayoría de palabras de estos modismos serán identificadas como "desconocido".

Como los modelos se basan un inglés neutro (generalmente extraído de enciclopedias como Wikipedia[30]) muchos modismos anglosajones serán omitidos. Incluso los modelos multilinguales de BERT se entrenan con *corpus* de español neutro y caen en el mismo escenario. Para evitar esto, se debe entrenar a un modelo con *corpus* de texto en español latinoamericano, lo cual es difícil de encontrar. Sin embargo, existen iniciativas en este camino por parte de la Sociedad Española de Procesamiento de Lenguaje Natural (SEPLN), lo cual puede ser un puntapié para la generación de modelos basados en BERT, en español para distintas tareas PLN[31].

Pero nuevamente, este es un proceso que requiere de muchos recursos y no es el enfoque de la presente memoria. Este trabajo debe ser capaz de aceptar fácilmente cualquier modelo de evaluación PLN que se desarrolle en el futuro, ese es el objetivo.

Finalmente, destacan los esfuerzos realizados por académicos y estudiantes de la Universidad de Chile en este tema, quienes entrenaron un modelo de propósito general como BERT-base, denominado BETO [16]. El *corpus* de texto usado contiene enciclopedias, subtítulos, libros, noticias, etc. Sin embargo, este modelo debe ser ajustado (*fine-tuned*) para cada tarea que se pretenda evaluar, en este caso, subjetividad y polaridad. Para lo segundo, podría entrenarse con el *dataset* de la SEPLN, pero para lo primero no existen *datasets*, ya que el único disponible es SUBJ para inglés. Se descarta su uso por temas de recursos y de enfoque del trabajo.

[28] https://huggingface.co/models?language=es,en&pipeline_tag=translation
[29] https://github.com/Helsinki-NLP/Tatoeba-Challenge/blob/master/models/spa-eng/README.md
[30] https://es.wikipedia.org
[31] http://tass.sepln.org

3.4. *Clustering*

3.4.1. Problema de fondo y limitaciones

Ahora bien, los artículos llegan en forma de *streaming* al sistema, por lo tanto, es necesario seleccionar una ventana de tiempo para efectuar un agrupamiento. Por ejemplo, se pueden agrupar todos los artículos en una ventana de tiempo de los últimos dos días, cada día, y mostrar esos resultados en el sistema. Esto no es óptimo, ya que seguramente más de un *cluster* se repetirá en días distintos, integrando nuevos artículos, pero a la vez dejando artículos atrás que no entran en las ventanas de tiempo más recientes.

Lo ideal es abordar esto como un problema de *clustering online*, donde antes de cada agrupación ya existan *clusters* en el tiempo anterior, y se asignen nuevos artículos a *clusters* existentes. Sin embargo, esto no deja cabida a la creación de nuevos *clusters*. Lo ideal es entonces que cada vez que se tomen agrupaciones de dos ventanas de tiempo, juntar los *clusters* que sean similares y evitar la redundancia de *clusters*. Algo así como un *clustering* sobre los resultados de *clustering*, que permita crear *clusters* sobre temas o hechos únicos, en modo cronológico e incremental.

Sin embargo, se limita el desarrollo a la primera propuesta: agrupar artículos en ventanas de tiempo de dos días, y mostrar estos resultados en el sistema. Esto, a modo de encontrar un algoritmo de *clustering* de calidad en primer lugar, ya que el proceso para evitar redundancia toma de base que se tienen buenos resultados de *clustering*.

Con el problema acotado, se debe encontrar la forma de agrupar un número fijo de artículos, siendo el titular y el cuerpo de la noticia el principal *input*. Sin embargo, también se dispone de *metadata* disponible, como la fecha de publicación, *tags*, categorías y las palabras más repetidas de cada artículo. Estos datos pueden servir para idear nuevos procedimientos de *clustering*.

Entonces, la manera de evaluar resultados debe ser mediante el análisis de casos, en forma cualitativa y donde sea posible analizar las métricas internas. Otra opción es generar pequeños *datasets* supervisados, hechos manualmente, pero se descarta esta opción por falta de tiempo.

3.4.2. Método con *embeddings*

Como se plantea en la Sección 2.3, la idea es utilizar Sentence-BERT [80] para elaborar *embeddings* a partir de las noticias disponibles en el sistema.

Sin embargo, se encuentra el mismo problema mencionado en la Sección 3.3.4: estos modelos están diseñados para el inglés. Nuevamente, se tienen las mismas alternativas descritas en la sección anterior, pero además se considera que en este caso, tanto Word2Vec como GloVe también tienen su versión en español de propósito general[32].

[32]https://github.com/dccuchile/spanish-word-embeddings

Generar un modelo desde cero se descarta por temas de recursos, mientras que emplear modelos como BETO o Word2Vec en español, se descarta porque no están ajustados (*fine-tuned* en inglés) para la tarea STS. Se tiene la hipótesis de que traducir texto al inglés y usar *embeddings* ajustados para STS pueden dar mejores resultados. Realizar un ajuste para STS de BETO o Word2Vec en español es inviable porque no existen *datasets* en español para ese propósito. La alternativa de traducir *datasets* en inglés de STS es redundante.

Por otro lado, usar MEAN GloVe genera *embeddings* en menos tiempo que aquellos producidos por modelos basados en BERT, pero que son de peor calidad (evidenciado por *spearman score*). En contraste, producir *embeddings* basados en BERT tiene un costo en tiempo de ejecución y memoria creciente en forma cuadrática con respecto al tamaño del *input*, pero el resultado es de mejor calidad [80]. Como en este caso se quiere priorizar la calidad de los resultados de *clustering*, entonces se escoge usar los *embeddings* basados en BERT, teniendo en cuenta que en un futuro se pueden utilizar otro tipo de *embeddings* en el sistema.

Se decide entonces usar *machine translation* junto a los *sentence embeddings* basados en BERT, y ajustados para STS, para luego usarlos como *input* para algoritmos de *clustering*. Se limita el uso de un sólo tipo de *embedding* basado en BERT, para enfocar el proceso de *clustering* en los algoritmos de agrupamiento, y no únicamente en el *input*. Además, por limitaciones de tiempo es inviable probar más de un tipo de *embedding*.

Queda un pequeño problema por resolver: modelos tipo BERT no pueden generar *sentence embeddings* para textos largos, porque tienen una limitación en el tamaño del *input* que generalmente es de 100 palabras aproximadamente. Por lo tanto, emplear el cuerpo de las noticias no es una opción válida. Además, modelos basados en BERT fueron entrenados con textos cortos, y representaciones de texto largas pueden no ser buenas.

Por esto, se decide utilizar únicamente titulares para generar *sentence embeddings*, ya que de hacer uso del cuerpo, o concatenar titular y cuerpo, el *input* se tiene que truncar y pueden quedar oraciones no resueltas.

Ahora bien, se pueden generar *sentence embeddings* para todos los textos, pero existe otro problema abierto: ¿Cuál es la mejor métrica?. No es directo que la métrica euclidiana o la similitud coseno sean las óptimas para calcular *distancias* entre *embeddings*. Como el espacio vectorial de los *embeddings* tiene una dimensionalidad tan alta (128 dimensiones en el caso de SMART-RoBERTa large [47]) existen problemas con la distancia euclidiana [89]. Por otro lado, la distancia coseno asigna el mismo peso a todas las dimensiones del *embedding*, lo cual teóricamente, la hace una métrica balanceada.

Esta elección se puede abordar de dos formas. Se puede dejar como un parámetro más a optimizar en el proceso de *clustering*, experimentando con una u otra métrica. O bien, se puede abordar como un problema en sí mismo. Librerías como *metric-learning* [28] pueden ayudar a adaptar una métrica Mahalanobis, transformando el espacio vectorial a uno nuevo, donde la distancia euclidiana sea efectiva. Pero para esto, se necesita información adicional, por ejemplo, de pares de *embeddings* que debieran estar más cerca o lejos. Modelos semi-supervisados utilizan esta información para encontrar una transformación del espacio que mejore resultados de tareas como *clustering*. Sin embargo, al ser un problema más grande, se deja planteada esta propuesta para el futuro, y se considera la métrica como un parámetro más de los algoritmos de *clustering*.

Finalmente, queda escoger una metodología para realizar *clustering* no supervisado. En general, no existe una limitación sobre qué algoritmo escoger. La librería *scikit-learn*[33] tiene 9 algoritmos implementados de libre uso. El problema nace en qué hiperparámetros emplear para cada uno y cómo evaluar los resultados si no existe *ground-truth*. Por ejemplo, especificar el número de *clusters* para *KMeans*[34], o el *epsilon* para algoritmos como *DBSCAN*[35].

Aún más relevante, si bien los hiperparámetros pueden ser optimizados para un *input* fijo, si este cambia, entonces los hiperparámetros encontrados para el *input* anterior no pueden reutilizarse. Lo cual es el presente caso, ya que los grupos de titulares cambiarán en cada agrupación. Por lo tanto, es imposible generalizar hiperparámetros para los algoritmos de *clustering* y deben ser optimizados cada vez, lo cual implica un costo de procesamiento, e inviabilidad al no existir *ground-truth*.

Como se menciona en la Sección 2.3.3, existen métricas para evaluar la calidad de un *cluster*, dentro de las más conocidas se encuentra el *Silhouette score* [82] y el *Calinski Score* [15]. Cada uno tiene un criterio de evaluación, por ejemplo, un *Silouette score* cercano a uno, indica que los *clusters* están lejos los unos de los otros y que son claramente distinguibles. Por lo tanto, basta el resultado de *clustering* con mayor *score*.

La idea es entonces, para cada algoritmo de *clustering*, probar cada criterio que evalúe la calidad de los *clusters* encontrados, y contrastar los resultados de cada *pipeline* (secuencia de acciones) cualitativamente. Sin embargo, este proceso es costoso, sobre todo para algoritmos como *OPTICS*[36] que realizan cientos de cálculos en cada iteración del algoritmo, y que se acrecientan en costo por la dimensionalidad de los *embeddings*.

Como la búsqueda de hiperparámetros (como el número de *clusters* para *KMeans*) es costosa, se plantea reemplazar el uso de una búsqueda exhaustiva por una búsqueda que utilice una optimización baysiana[37]. Esta trabaja construyendo una distribución posterior de funciones (proceso gausiano) que mejor describe la función que se quiere optimizar. Al aumentar el número de observaciones, la distribución posterior mejora, y el algoritmo se vuelve más certero en regiones donde vale la pena probar parámetros y en donde no. Lo cual mejora significativamente los tiempos de ejecución en comparación a una búsqueda de fuerza bruta.

Con todo esto presente, el proceso completo queda planteado:

1. Traducir los titulares al inglés mediante *machine translation*.
2. Calcular *sentence embeddings* usando SRoBERTa o SBERT (modelos tipo BERT).
3. Para cada algoritmo de *clustering*, escoger el resultado con mejor *Calinski o Silhouette score*.
4. Evaluar cualitativamente cual *pipeline* fue la mejor.

[33]https://scikit-learn.org
[34]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[35]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
[36]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.OPTICS.html
[37]https://github.com/fmfn/BayesianOptimization

3.4.3. Método con heurística

Procedimiento

Se propone utilizar un método propuesto en la publicación *"Prediction and Characterization of High-Activity Events in Social Media Triggered by Real-World News"* Kalyanam J et al. [50] que permite detectar eventos a partir de *tweets*. Esta metodología, inspirada originalmente, en otra publicación [8], detecta palabras claves (*keywords* en inglés) en un conjunto de *tweets*, generando como *output* conjuntos de *keywords*, que finalmente, representan eventos distintos.

El proceso completo contempla una fase previa de limpieza denominada *tokenization* de los *tweets*. En esta etapa se eliminan las *stopwords* de cada *tweet* para descartar palabras que no aportan información. También se identifican entidades por tipo (persona, localización, empresa, etc.) que pueden ser filtradas. Notar que las entidades pueden contener más de una palabra, como por ejemplo, "Convención Constitucional". Una vez hecha la *tokenization*, se aplica el Algoritmo 1 que detecta eventos propiamente tal.

Algoritmo 1 Heurística de detección de eventos

Input: A set of M sets of words, $S = \{H_1, H_2, ..., H_M\}$, positive integers k,η
Output: k sets of keywords, $G = \{I_1, I_2, ..., I_k\}$

$I_i \leftarrow \emptyset$ for $i = 1, 2, ..., k$
$score \leftarrow$ empty dictionary for $i = 1, 2, ..., k$
$i \leftarrow 1$
for every pair of headlines $\{H_a, H_b\} \in S$ such that $|H_a \cap H_b| \geq \eta$ **do**
    $G \leftarrow H_a \cap H_b$
    $j \leftarrow argmax_j |I_j \cap G|$
    if $|I_j \cap G| \geq \eta$ **then**
        $I_j \leftarrow I_j \cap G$
        $score_j[w] \leftarrow score_j[w] + 1$ for all $w \in I_j$
    else
        $I_i \leftarrow G$
        $score_j[w] \leftarrow 1$ for all $w \in I_i$
        $i \leftarrow i + 1$
    end if
end for
$total\_score_i \leftarrow \sum_{w \in I_i} score_i[w]$ for $i = 1, 2, ..., k$
return $G \leftarrow (I_i$ sorted by $total\_score_i)$

Notar que se asigna un *score* de relevancia a cada evento. Este procedimiento funciona de forma tal que aquellos eventos con menos *keywords*, tendrán más relevancia al final. Por otro lado, la relevancia está directamente relacionada con número de *tweets* que utilizan las *keywords* de estos eventos. De este modo, se retorna una lista de eventos definidos como un conjunto de elementos del estilo (*keywords*, relevancia).

El algoritmo en palabras simples, obtiene un *stream* de *tweets*, que luego bajo la simple lógica de palabras en común entre *tweets* y *kewords* de eventos, obtiene relaciones entre *tweets* y eventos. Se define un parámetro *threshold* η (o simplemente *threshold*), para determinar el

número mínimo de palabras en común para pertenecer a un evento. Si un *tweet* puede pertenecer a dos eventos, se asigna a alguno arbitrariamente. Y *tweets* que tienen menos que *threshold* palabras en común con todos los eventos, son descartados.

Adaptación

La idea es usar este mecanismo identificando dos procesos principales: (1) identificación de eventos, y (2) asignación de elementos a los eventos. Para lo primero basta reemplazar los *tweets* por los titulares de noticias en el Algoritmo 1, pero esta vez no filtrando entidades en el proceso de *tokenization*. Para lo segundo se propone utilizar una heurística propia, expuesta en el Algoritmo 2.

Esta heurística de asignación sirve para determinar una *label* (que representa un evento) para cada titular y además identifica *outliers* para aquellos titulares que no tienen palabras en común con ningún evento.

Este método genera tanto un conjunto de *keywords* para cada evento (mediante las *label*), como un *score* de relavancia. Se usan las *keywords* para identificar a un evento, y mostrarlas en las visualizaciones del *frontend*. Mientras que el *score* de relevancia será ignorado de momento, porque si bien sirve para ordenar eventos por relevancia, no se estima que este *score* denote la relevancia real de un evento, al relacionarse directamente con la cantidad de elementos que dan cobertura, ya que pueden existir eventos relevantes pero con menor cobertura. Por simpleza, se asume que usarlo es análogo a ordenar los eventos por número de artículos en cada evento.

Este método es mucho más sencillo que la metodología con *embeddings*, y al mismo tiempo, requiere de muchos menos cálculos, por lo tanto, se espera que tenga un mayor rendimiento. Para comparar esta metodología con aquella basada en *embeddings*, se deben analizar casos de uso.

3.5. Base de datos

Como se adelanta en la Sección 2.4, Elasticsearch representa una alternativa atractiva y balanceada para ser la base de datos de texto del sistema. De esta forma, se escoge su uso sobre las otras herramientas, por tener una comunidad más grande y activa, contar con una mejor documentación, realizar actualizaciones recurrentemente, ser escalable y por su uso sencillo mediante una API desde Python.

A continuación, se detalla el modelo de datos elaborado para esta herramienta, que tiene en consideración la flexibilidad de los datos, debido a que el sistema debe permitir la incorporación de nuevos modelos PLN, sin mayor fricción o inconvenientes.

Algoritmo 2 Heurística de asignación a eventos

Input: A set of N sets of words, $A = \{A_1, A_2, ..., A_N\}$,
a set of k sets of keywords, $E = \{E_1, E_2, ..., E_k\}$,
and a positive integer α acting as threshold

Output: N labels

$outlier_{index} \leftarrow k$
$l_i \leftarrow \emptyset$ empty list for $j = 1, 2, ..., N$
$j \leftarrow 0$
for $j \in \{1, 2, ..., N\}$ **do** ▷ For each article
  $comun_i \leftarrow \emptyset$ empty list for $i = 1, 2, ..., k$
  $i \leftarrow 0$
  for $i \in \{1, 2, ..., k\}$ **do** ▷ For each event
    $counter \leftarrow 0$
    for $word \in A_j$ **do** ▷ Count common words
      for $keyword \in E_i$ **do**
        if $word == keyword$ **then**
          $counter \leftarrow counter + 1$
        else if $|word \cap keyword.split(" ")| > 0$ **then** ▷ Separate words in *keyword*
          $counter \leftarrow counter + |keyword.split(" ")|$
        else if $|word.split(" ") \cap keyword| > 0$ **then** ▷ Separate words in *word*
          $counter \leftarrow counter + |word.split(" ")|$
        end if
      end for
    end for
    if $counter \geq \alpha$ **then** ▷ If common words are above α, asssign *counter*
      $comun_i \leftarrow counter$
    else ▷ If common words are below α, assign 0
      $comun_i \leftarrow 0$
    end if
    $i \leftarrow i + 1$
  end for
  if $comun_w = 0 \forall w = 1, 2, ..., k$ **then** ▷ If article is an outlier, assign an outlier label
    $l_j \leftarrow outlier_{index}$
    $outlier_{index} \leftarrow outlier_{index} + 1$
  else ▷ If not, assign the label of the event with more affinity
    $l_j \leftarrow argmax(comun)$
  end if
  $j \leftarrow j + 1$
end for
return l

3.5.1. Modelo de datos

Dentro de las decisiones de diseño se encuentra la definición del modelo de datos que se emplea para almacenar artículos, eventos y agrupaciones. Se exponen los *mappings* propuestos y se justifica la decisión de diseño de cada uno. Como principal criterio de diseño, se utiliza la flexibilidad de los modelos, porque se deben poder incorporar nuevas formas de calcular polaridad, subjetividad, nuevas formas de agrupamiento, entre otras cosas.

Se usa el idioma inglés para el *mapping* de modelos, por estandarización. Por otro lado, a modo general, todos los parámetros de fecha adoptan el formato estándar "DD-MM-YYYY'T'HH:MM:SS.ZZZ" reconocido por Elasticsearch[38].

Artículos

Dentro de la información básica de cada artículo, se propone incluir: *title*, *drop*, *body*, *media_id*, *date_scrapping*, *date_publication*, *date_modification*, *authors*, *sources*, *categories*, *tags*, y *url_article*. La mayoría de estos parámetros se explica por sí mismo, a modo de aclaración, *date_modification* es fecha que potencialmente traen algunos artículos que informan sobre ediciones del mismo, mientras que *sources* son las fuentes de información de cada artículo, que algunos medios proveen. Tanto *authors* como *sources* y *categories* son listas de texto, ya que cada una puede contener más de un elemento. Ver Código Fuente B.1 para ver el resultado final del *mapping*.

También, se incluye en cada artículo un objeto llamado *image*, que tiene *url, autorship y description*, que sirve como imagen destacada. Para artículos que provengan de un *tweet*, se incluye un parámetro *tweet_id*, un *flag has_tweet* (que permite distinguir artículos provenientes de *tweets*), y una lista de objetos denominada *tweet_metrics*, donde cada objeto tiene los parámetros *retweets*, *likes* y *timestamp*. La idea es poder en un futuro actualizar las métricas de cada artículo y ver su evolución en el tiempo. Basta ordenar la lista por *timestamp* y graficar los *retweets* y *likes*. Ver Código Fuente B.2 para ver el resultado final del *mapping*.

Dentro de los parámetros PLN se encuentran seis objetos principales: *tokenization_es, en, embeddings, knn, polarity y subjectivity.*

El primero se usa para preprocesar el texto que recibe la agrupación por heurística, es una lista de objetos con los parámetros *timestamp, text_source, model, words* y *word_count*. En *text_source* se especifica qué texto fue procesado (*title, drop, body,* o *drop+body*), en *model* se especifica el algoritmo utilizado, en *words* quedan las palabras finales después del proceso de *tokenization* y en *word_count* queda una lista de pares *(word, count)* que contiene el recuento de palabras únicas provenientes de *text_source*. La idea es guardar en *words* el *input* para el agrupamiento por heurística, mientras que en *word_count*, el *input* necesario para elaboración de visualizaciones relevantes en el *frontend* que se detallan en la sección subsiguiente. Este parámetro PLN es una lista de objetos para soportar distintos modelos de *tokenization*, y de *text_source*. Ver Código Fuente B.3 para ver el resultado final del *mapping*.

[38]https://www.elastic.co/guide/en/elasticsearch/reference/current/date.html

El segundo parámetro PLN corresponde a una lista de objetos que representan las traducciones realizadas al texto original. Contiene los parámetros *timestamp, text_source, model, text* y un objeto *tokenization_en*. Este último cumple la misma función que el parámetro anterior, pero esta vez procesado con la traducción hecha del texto al inglés. Por su parte, *text_source* especifica qué texto se tradujo (*title, drop, body* o *drop+body*), *model* el modelo de ML utilizado para traducir, y *text* el texto final en inglés. Este parámetro PLN es una lista para soportar la traducción de todos los *text_source* posibles, y múltiples formas de realizar la *tokenization* para cada traducción. Ver Código Fuente B.4 para ver el resultado final del *mapping*.

El tercer parámetro denominado *embeddings*, es una lista de objetos con los atributos *timestamp, text_source, lang, model, embedding*. El campo *text_source* cumple la misma función anterior, *lang* especifica el lenguaje del modelo que genera el *embedding*, *model* el modelo en sí y *embedding* el vector numérico calculado por el modelo. Este parámetro es una lista de objetos para soportar distintos modelos, lenguajes y *text_source*. Ver Código Fuente B.5 para ver el resultado final del *mapping*.

El cuarto parámetro sirve para guardar los k vecinos de un artículo en una ventana de tiempo determinada. Corresponde a una lista de objetivos para variar el modelo utilizado, el texto utilizado como *input* y la ventana de tiempo. Cada objeto tiene los parámetros *article_ids, text_source, lang, model, timestamp, date_since y date_last*. Estos dos últimos definen la ventana de tiempo utilizada, mientras que el resultado queda contenido en *article_ids* como una lista de *ids* de artículos del vecindario. Ver Código Fuente B.6 para ver el *mapping* final.

El quinto parámetro sirve para guardar los resultados de polaridad, y corresponde a una lista de objetos con los parámetros *timestamp, text_source, lang, model, positive, negative, neutral y compound*. Los últimos cuatro campos corresponden al *output* que brindan los modelos de polaridad. En *model* se especifica qué modelo se usó, *lang* el lenguaje, y *text_source* el tipo de texto utilizado. El parámetro es una lista de objetos para poder variar modelo, lenguaje y *text_source*. Ver Código Fuente B.7 para ver el resultado final del *mapping*.

Finalmente, el sexto parámetro se usa en forma similar al anterior, para guardar los valores de subjetividad. Por ello, es una lista de objetos con los parámetros *timestamp, text_source, lang, model y score*. Este último representa el valor de subjetividad del texto especificado en el campo *text_source*, evaluado por el modelo especificado en *model*. Es un valor que va de 0 a 1, donde 1 es totalmente subjetivo y 0 es objetivo. Ver Código Fuente B.8 para ver el resultado final del *mapping*.

Además, se incluyen los *flags* tipo *bool*: *has_knn, preprocessed* y *event_id*. La primera sirve para conocer directamente si el artículo ha tenido un cálculo *knn* (más eficiente que preguntar si la lista de objetos del parámetro PLN *knn* está vacía). El segundo, sirve para determinar si el artículo tiene los parámetros *embeddings*, *tokenization_es, en, polarity* y *subjectivity* fueron calculados. Y por último, el parámetro *event_id* sirve para ver todos los eventos a los cuales un artículo está asociado, esto se detallará en el *mapping* de Eventos. Ver Código Fuente B.9 para ver el resultado final del *mapping*.

Como se puede ver, los cinco parámetros PLN son listas de objetos que permiten incorporar nuevos modelos que realicen las distintas tareas sin mayores complicaciones. Además, la idea de guardar detalles de qué modelo se utilizó, con qué texto específicamente, brinda una trans-

parencia al sistema posible de mostrar en el *frontend*, y también para citar a los autores de los modelos en sí. Por su parte, los parámetros de *timestamp* brindan una referencia temporal de cuándo se ejecutaron los modelos cada vez.

El *mapping* descrito se encuentra en su totalidad en Anexo B.1.

Eventos

En simple, los eventos son agrupaciones de artículos. A modo más general, un evento denota un hecho particular para el cual se han hecho diversas coberturas. Un evento comprende artículos de días distintos y su duración total es relativa a la naturaleza del evento. Por ejemplo, un *robo a un banco* es un evento que puede ser noticia por un par de días, pero una *elección presidencial* puede durar más de una semana.

Los eventos en este trabajo corresponden al *output* generado por los algoritmos de *clustering*. Se considera en este *mapping* la idea de que un evento puede incorporar nuevos artículos día a día en forma *online*. Sin embargo, para esto es necesario el desarrollo de heurísticas que realicen agrupamientos sobre resultados de *clustering* hechos sobre distintas ventanas de tiempo consecutivas. Tal como se describe en la Sección 3.4.1, esto queda propuesto para trabajo futuro, pero se tiene la voluntad de dejar el *mapping* actual listo para esta labor.

Un evento, como objeto en el *mapping*, es entonces una lista de objetos denominados *snapshots*. Cada *snapshot* contiene los campos *timestamp, date_since, date_last, clustering_model, new_keywords* y *new_articles_id*. Estos sirven para denotar la ventana de tiempo en la que hizo una agrupación, el modelo utilizado, las *keywords* (palarbas claves) que identifican al evento y la lista de *ids* de artículos que comprenden este evento. La idea es que si un evento necesita incorporar nuevos artículos, se agrega un nuevo objeto *snapshot* a la lista, con los nuevos *new_articles_id*, y donde se actualicen todos los demás parámetros del evento. Ver Código Fuente B.10 para ver el resultado final del *mapping* de *sanpshots*.

A cada *snapshot*, se suma un objeto tipo *image* con los campos *url, media, autorship* y *description* (al igual que el *mapping* de los artículos, ver Código Fuente B.2 como referencia). La idea es seleccionar aleatoriamente la imagen de algún artículo del evento, y emplearlo como representativo del evento.

Se guardan también métricas acumuladas de los artículos, que comprenden el evento y para ello existen los objetos *accum_twitter, accum_avg_polarity, accum_avg_subjectivity* y *accum_top20_word_count_drop_and_body*. La primera métrica almacena el número de *retweets* y *likes*, la segunda los campos *text_source, lang, model, positive, negative, neutral* y *compound*, que contienen los promedios de polaridad de los artículos. La tercera métrica sigue un camino similar con los parámetros *text_source, lang, model* y *score*, pero para subjetividad. Finalmente, la última métrica, es en realidad una lista veinte de pares *(word, count)* que contiene las palabras más repetidas en la concatenación de bajada y cuerpo. Ver Código Fuente B.11 para ver el resultado final del *mapping*.

Por último, se incorporan también al modelo los objetos de *title* y *summary* que tendrán los mismos parámetros: *model, lang, text*, un objeto tipo *polarity* y un objeto tipo *subjectivity*, con

los mismos campos que en casos anteriores. Acá se limita *title* y *summary* a un sólo objeto, y no una lista de objetos por simplicidad. Esto impide que, por ejemplo, se puedan probar distintos modelos *summarizers* para un mismo *snapshot*. Lo mismo ocurre para *polarity* y *subjectivity*. Se toma esta decisión porque se cree que en vez de flexibilizar el modelo de esta forma, esto ya ocurre a través del objeto *snapshot*. Por ejemplo, para probar nuevos modelos basta agregar un nuevo objeto a la lista de *snapshots*. Al hacer esto, se permite no solo cambiar un modelo, sino que todos los parámetros de un evento. Para tomar el último *snapshot*, basta ordenar la lista por el parámetro *timestamp*. Ver Código Fuente B.12 y B.13 para ver el resultado final del *mapping* para los objetos de *title* y *summary*.

Y para simplicidad de consultas para ordenar eventos por orden cronológico, se agrega el parámetro *date_first_article* a cada evento, fuera del objeto *snapshot*. Esto, porque este parámetro nunca cambia, y porque es más sencillo consultar este campo, que buscar el primer *snapshot*, traer todos los artículos, ordenarlos por fecha y obtener la fecha del primer artículo. Ver Código Fuente B.14 como referencia.

Finalmente, un evento es un conjunto de *snapshots* que contienen los artículos del evento en forma incremental. Sin embargo, como esto es en realidad un modelo pensado para el trabajo futuro, en la práctica sólo se usa el primer *snapshot* de cada evento.

El *mapping* descrito se encuentra en su totalidad en Anexo B.2.

Agrupaciones

Finalmente, el modelo llamado *clusters* guarda la información de un conjunto de eventos, es decir, de agrupaciones. El campo más importante se denomina *events_ids*, que contiene la lista de *ids* de eventos. A esto se le suman los campos del modelo usado para agrupar eventos: *date_execution, date_since, date_last, n_articles, execution_time_in_sec, algorithm* y *algorithm_simple*. Los dos últimos describen el algoritmo utilizado en palabras, el campo *algorithm* identifica todos los algoritmos utilizados para agrupar, mientras que *algorithm_simple* más bien la técnica utilizada (para mostrar en el *frontend*). Por ejemplo, si se emplean titulares en inglés, junto a la heurística de agrupación, se debe identificar el modelo utilizado para traducir, que se tradujo el título, la forma de *tokenization* en inglés y el algoritmo específico de heurística utilizado. Al mismo tiempo *algorithm_simple* sólo sería "Heurística". Ver Código Fuente B.15 como referencia.

A esto se suman dos listas de objetos: *algorithm_params* y *algorithm_metrics*. Ambos son simples listas con pares *(name, value)* que permiten guardar la información de los parámetros utilizados por el algoritmo, y de las métricas internas del *cluster* de eventos formados, por ejemplo, guardar el *Silouhette* o *Calinski score*. Ver Código Fuente B.15 como referencia.

El *mapping* descrito se encuentra en su totalidad en Anexo B.3.

3.6. Buscador avanzado

Con las restricciones impuestas por los *mapping* especificados anteriormente, es posible ahora definir el funcionamiento del buscador. La idea central es que el sistema provea de un mecanismo para buscar artículos en forma histórica mediante filtros de palabras y parámetros de un artículo. La búsqueda de eventos se deja como trabajo futuro.

Como base, se plantea implementar más filtros que Google News. Estos son: búsqueda por frase exacta, incluir palabras, excluir palabras, sitio web (medio) y fecha (del presente hacia atrás). Ver Figura 4.28 como referencia.

La REST API de Elasticsearch permite buscar texto en los distintos campos de un *mapping*. Como la búsqueda es sobre los artículos, se puede buscar texto directamente en los campos *media_ id, authors, soucres, categories, tags, title, drop* y *body*. El buscador debe permitir buscar texto sobre cualquiera de estos campos.

Para satisfacer la búsqueda de palabras exactas, basta tomar el *input* de búsqueda y concatenar la consulta mediante conectores lógicos AND. Para la búsqueda con palabras a incluir se procede de forma similar pero con conectores OR, y por último, para excluir palabras se especifican las palabras con conectores NOT.

Para filtrar por medio, basta especificar el *media_ id* y juntar todos los medios que se quiera mediante conectores tipo OR. Para buscar artículos en un rango de fecha basta utilizar una *range query* en elastic para el campo *date_ publication* de los artículos.

Hasta este punto se tienen todos los filtros de Google News y filtros adicionales también, como lo es la búsqueda por autor, fuente y categorías. También el grado de fineza es mayor al poder buscar palabras en distintos parámetros de cada artículo. Pero además de esto, se quiere filtrar artículos por parámetros PLN y métricas de Twitter. Como esta información está dentro de una lista de objetos en cada artículo, es necesario primero realizar el filtro por palabras y luego procesar aparte los demás filtros.

La idea es incorporar filtros de comparación numérica, por ejemplo, que el campo subjetividad sea: menor, menor o igual, igual, mayor o igual o mayor a cierto valor dado por quien hace la búsqueda. Esto para los campos de *positive, negative, neutral y compound* en el caso de polaridad. Para el campo *subjectivity* en el caso de subjetividad. Y para *retweets* y *likes* en el caso de métricas de Twitter. Se espera que el sistema permita añadir n filtros numéricos.

Los resultados de búsqueda deben mostrar todos los artículos coincidentes con filtros especificados. El usuario puede entonces visitar el sitio del medio que aloja cada artículo y obtener desde ahí la información que necesite.

Con todo esto, el buscador queda con la capacidad de buscar con múltiples filtros, a fin de poder realizar búsquedas históricas de artículos, y con contenido enriquecido. Se espera que esto permita a cualquier usuario la búsqueda objetiva de información histórica, favorezca la investigación periodística y sirva como repositorio para cualquier fin que los usuarios estimen convenientes.

Notar que esta disponibilidad de la información queda sujeta a la mantención de los sitios de cada medio, ya que el sistema en sí no es un repositorio de información, sino un repositorio de indexación de información.

Finalmente, a modo de ejemplo, se exponen las siguientes consultas como caso de uso (donde las letras mayúsculas son reemplazables):

1. Buscar los artículos del medio X, entre la fecha Y e Z, donde la subjetividad haya sido mayor a W.
2. Buscar los artículos que contengan la palabra X, y no la palabra Y, cuya positividad sea mayor a Z.
3. Buscar los artículos que contengan la frase exacta X en su titular, de los medios Y, Z y W de la última semana.
4. Buscar cualquier artículo cuya subjetividad sea mayor a X y negatividad menor a Y.
5. Buscar los artículos cuya cantidad de *retweets* sea mayor a 100.

3.7. Arquitectura del sistema

Teniendo en cuenta los antecedentes de la Sección 2.5, se decide utilizar una arquitectura basada en microservicios, por su modularidad, escalabilidad y por ser altamente mantenible.

En el caso particular de esta memoria, existen procesos que serán abordados por modelos de ML que son altamente consumidores de recursos. Si se optara por la arquitectura monolítica, cada servidor debiera ser capaz de cargar en memoria todos los modelos de ML a usar, *frontend*, *backend* y la base de datos, lo cual es altamente inviable.

Si, por el contrario, se tienen servicios especializados para cada función mayor de la aplicación, entonces cada servidor es capaz de albergar más de una instancia de microservicio y escalar horizontalmente.

3.7.1. Componentes y microservicios

Se propone el uso de los siguientes componentes en aplicación:

1. **Elasticsearch:** Base de datos de texto, altamente escalable y que permite la redundancia de información para evitar pérdidas. Encargada de guardar artículos de noticias, eventos y agrupaciones de eventos.
2. ***Scrappers*:** Procesos encargados de hacer llamados a la API de Twitter para extraer las *urls* base de cada medio, y luego extraer la información desde portales de noticias simulando la interacción de un usuario común. Sólo extraen información públicamente visible y necesitan conexión con la base de datos para evitar duplicados.

3. ***Preprocessing service*:** Servicio que procesa la información extraída por los *scrappers* añadiendo *sentence embeddings*, traducciones y cálculos de subjetividad y polaridad necesarios para cada artículo.
4. ***Query service*:** Servicio que atiende solicitudes generales de lectura, para los módulos de artículo, evento y agrupaciones.
5. ***Seacrh service*:** Servicio que atiende las búsquedas avanzadas entregadas por los usuarios.
6. ***KNN service*:** Servicio que atiende las búsquedas por similitud de *embeddings* realizadas por los usuarios.
7. ***Clustering service*:** Servicio que efectúa agrupaciones de artículos y genera eventos para un rango de fechas dado. Realiza llamadas al *Summary Service* para realizar resúmenes de noticias de cada evento.
8. ***Summary service*:** Servicio que genera resúmenes de eventos particulares.
9. ***Frontend*:** Plataforma que, a nivel de usuario, permite interactuar con el sistema. Consta de dos vistas principales: agrupaciones y buscador avanzado.
10. ***Control service*:** Servicio que vela por la consistencia de los datos almacenados al terminar procesos pendientes como la generación de resúmenes y el preprocesamiento de artículos.
11. **Kibana *service*:** Servicio que observa el estado de la base de datos, incluyendo los *shards*, *replicas* y que permite navegar por la misma en forma independiente.

Cada uno de ellos representa un componente a desarrollar en el sistema, autocontenido e independiente, lo cual permite hacer *deployment* de manera modular y también generar varias instancias del mismo para escalar forma horizontal.

Ahora bien escalar en forma horizontal implica el uso de balanceadores de carga (Load balancers (LB) en inglés), que en términos generales reciben múltiples solicitudes y deciden a qué instancia del servicio dirigir cada solicitud mediante un criterio establecido. Pueden ser implementados a nivel de hardware o *software* y dentro de estos criterios de elección se encuentran: *round-robin*, *least load*, IP *hash*, etc.

A modo de ejemplo, nginx[39] es un *web server* que puede ser usado como balanceador de carga que mediante *software* redirige las solicitudes a los distintos procesos de un microservicio. De ser necesario, el sistema puede escalar utilizando este *software* o cualquier otro.

Con todo esto presente, el flujo de datos esperado entre los distintos microservicios y componentes del sistema, se aprecia en la Figura 3.1.

En primer lugar, se observa como los *scrappers* usan la API de Twitter para obtener las *urls* base, y luego realizar la lectura de noticias en los distintos portales de los medios digitales. En segundo lugar, estos datos son enviados al *Preprocess service* quien genera los cálculos necesarios para enriquecer los artículos y almacena los resultados en los *clusters* de Elasticsearch. En tercer

[39]https://www.nginx.com

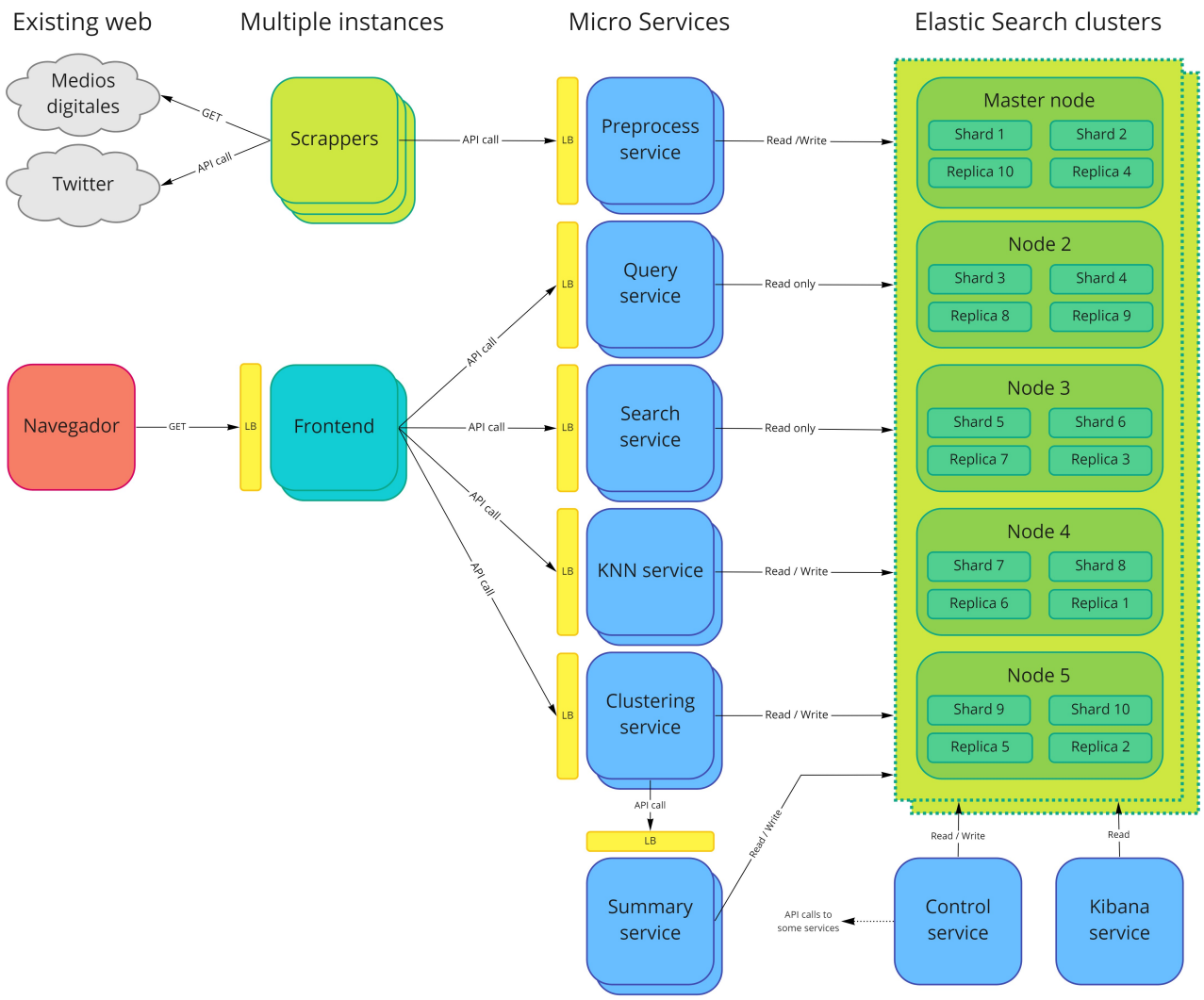

Figura 3.1: Arquitectura del sistema

lugar, se encuentran los distintos servicios que interactúan con el *frontend*: *Query, Search, KNN y Clustering service*. Notar que el *Summary service* queda en forma independiente del *Clustering service*, porque su tarea específica es resumir noticias, no agrupar, a pesar de que ambos servicios interactúan con eventos.

En cuarto lugar, se encuentra el *Control service*, que de manera regular hace llamadas a Elasticsearch en búsqueda de resúmenes no elaborados o artículos no preprocesados. Finalmente, se encuentra Kibana como administrador interactivo de la base datos.

Dentro de los posibles cuellos de botella, se encuentra la base de datos misma, ya que un número de *shards* inadecuado puede provocar un *overload* en términos de solicitudes o de tamaño de paquetes. Encontrar un número balanceado de *shards* es importante, al igual que el número de *replicas*, porque muchas *replicas* pueden saturar la memoria de cada servidor. Lo bueno es que existen varias guías prácticas para abordar este problema[40].

[40]https://www.elastic.co/guide/en/elasticsearch/reference/current/size-your-shards.html

También se encuentra un posible cuello de botella en el *overload* generado por el *Search service* entrega muchos resultados, pero que es fácilmente manejable mediante paginamiento.

3.7.2. Acotamientos

La arquitectura planteada anteriormente busca abastecer la demanda de miles de usuarios, escenario que no es necesario de abordar en la etapa actual de la aplicación, pero que es necesario considerar desde un comienzo. Al tratarse de un prototipo o *Proof of Concept*, el uso de múltiples servidores no es necesario, ni tampoco el uso de más de una instancia por microservicio.

Para este trabajo, se utiliza una arquitectura basada en el diseño escalable, donde cada componente del sistema representa un solo proceso dentro de una máquina local, y no un servidor completo. Elasticsearch cuenta con dos nodos, compuestos de al menos dos *shards*, cada uno con una *replica*, lo cual garantiza abordar eventuales fallas. También se hacen respaldos periódicos para evitar pérdida de información.

En específico, el servidor a utilizar para ejecutar cada componente cuenta con las siguientes características:

1. **Sistema Operativo:** Windows 10
2. **Procesador:** Intel Core i5-9300H CPU @ 2.40GHz, 4 núcleos y 8 procesadores lógicos
3. **Almacenamiento:** 1 Tb SSD PCI Express 3.0
4. **RAM:** 24 Gb dual channel @ 1666 MHz
5. **Tarjeta gráfica:** NVIDIA GeForce GTX 1650 4Gb

Se espera que sea suficientemente capaz de cargar los modelos de ML necesarios para calcular *embeddings*. También de resumir y traducir, procesos que pueden consumir más recursos en el sistema.

3.8. Visualización

3.8.1. Metodología

El sistema consta de cinco vistas principales descritas a continuación:

1. **Lista de Agrupaciones:** En ella se listan todas las agrupaciones realizadas, junto a la metodología usada, la cantidad de eventos y artículos de cada agrupación. También permite realizar nuevas agrupaciones mediante un formulario.

2. **Agrupación:** Se muestra una agrupación en particular, y se despliegan verticalmente todos los eventos que pertenecen a esta agrupación. Se despliega información relevante de cada evento.

3. **Detalle de evento:** Se muestra un evento particular con todos sus campos, la lista de artículos pertenecientes a él junto a las métricas PLN y de Twitter. Además, se muestran gráficos y visualizaciones relevantes.

4. **Buscador:** Despliegue de un buscador con todas las funcionalidades ya mencionadas y un botón "buscar" que al accionarse liste todos los artículos relacionados con la búsqueda. Se adjuntan visualizaciones adicionales también.

5. **Informaciones:** Vista explicativa del sistema, que explique las funcionalidades más relevantes del mismo, y que incluya explicaciones sobre polaridad y subjetividad.

Desarrollar una visualización adecuada es fundamental para incrementar la usabilidad de un sistema. Pero esto no es preocupante por la naturaleza de prototipo o *Proof of Concept* de este trabajo. Un prototipo se define como un módulo que sirve para probar la factibilidad de un sistema, muchas veces incompleto. El principal enfoque de este trabajo es crear un prototipo funcional, y no desarrollar una interfaz para usuarios finales. Es muy importante recalcar esta distinción.

El principal foco de este trabajo es la funcionalidad del sistema, y por ello se procederá con una metodología mucho más sencilla: (1) Se realizarán *mockups* con algún *software* de diseño de interfaces, de las vistas mencionadas anteriormente, y (2) Se implementarán estas vistas a nivel de *frontend*, que luego se conectarán al *backend*. La idea, es inspirarse en soluciones que ya están en producción como Google News o Event Resgistry. Ver Anexo A.

El problema con esta metodología es que en ningún momento se obtiene *feedback* de los usuarios y se corre el riesgo de desarrollar una interfaz que tenga baja usabilidad.

Se elaboran entonces, *mockups* para cada una de las vistas descritas anteriormente, en forma iterativa, mediante la herramienta Adobe XD[41].

[41]https://www.adobe.com/cl/products/xd.html

3.8.2. *Mockups*

A continuación se describen las vistas que deben ser implementadas en el sistema. Sólo dos tienen *mockups* hechos en Adobe XD, mientras que las demás cuentan con la descripción de la vista únicamente.

Vista lista de agrupaciones

Esta vista consta de dos componentes principales: el formulario para realizar nuevas agrupaciones, y la lista de agrupaciones realizadas anteriormente.

En el formulario, se puede especificar el rango de fechas de artículos a considerar, con dos campos tipo fecha. Luego, mediante un selector se debe indicar qué metodología utilizar: *embeddings* o heurística. De escoger heurística, deben aparecer los parámetros modificables de esta metodología en forma de selectores y campos numéricos: *join_similar* y *min_size*. De seleccionar *embeddings*, también deben aparecer campos personalizables, pero en este caso son: algoritmo de *clustering*, criterio de *clustering* no supervisado, porcentaje de ruido estimado y mínimo número de *clusters*. Mediante un botón "agrupar" se debe enviar una solicitud al microservicio de *clustering*, que retorne los resultados de la agrupación y se redirija a la misma.

Es interesante también, configurar los parámetros específicos de los algoritmos de *clustering* que se usan en casa caso, pero esto no se incluye por dos razones. La primera es para mantener la simplicidad de la interfaz, y la segunda es porque su implementación es un poco compleja a nivel de *frontend* y de *backend*, ya que los parámetros de cada algoritmo no son comunes para todos.

Por otro lado, la lista de agrupaciones mostrará rango de fechas, número de eventos, número de artículos y metodología utilizada de cada agrupación. La idea es siempre mostrar las agrupaciones en el orden cronológico según la fecha desde de cada agrupación. Esta lista estará paginada para no sobrecargar la vista. Estas agrupaciones deben ser las realizadas por el *backend*, y que consideran una ventana de tres días.

Esta vista no cuenta con *mockups*.

Vista agrupación

Esta vista muestra verticalmente todos los eventos pertenecientes a una agrupación determinada. La idea es mostrar los eventos en forma de *feed*, que es bastante estándar en redes sociales. Se mostrarán los datos de la agrupación al inicio, y abajo se mostrará el *feed* de eventos. Se toman como referencia las vistas de Event Registry y de Google News adjuntas en Anexo A.

La información relevante de cada evento es su título generado, una imagen representativa, el resumen, las métricas acumuladas, las palabras claves que lo identifican y sus términos más frecuentes. Lo último, se visualiza en forma de nube de palabras, a modo de destacar los términos con más importancia pertenecientes al evento. Para armar esta nube, se utilizan los 20 términos más repetidos presentes en las bajadas y cuerpos de todos los artículos que componen el evento,

y que es obtenible directamente desde el *mapping* de evento. Los campos de *tags*, categorías se muestran sólo en la Vista detalle de evento, mientras que la sobre imagen representativa se muestra la información de autoría (desde qué medio se sacó).

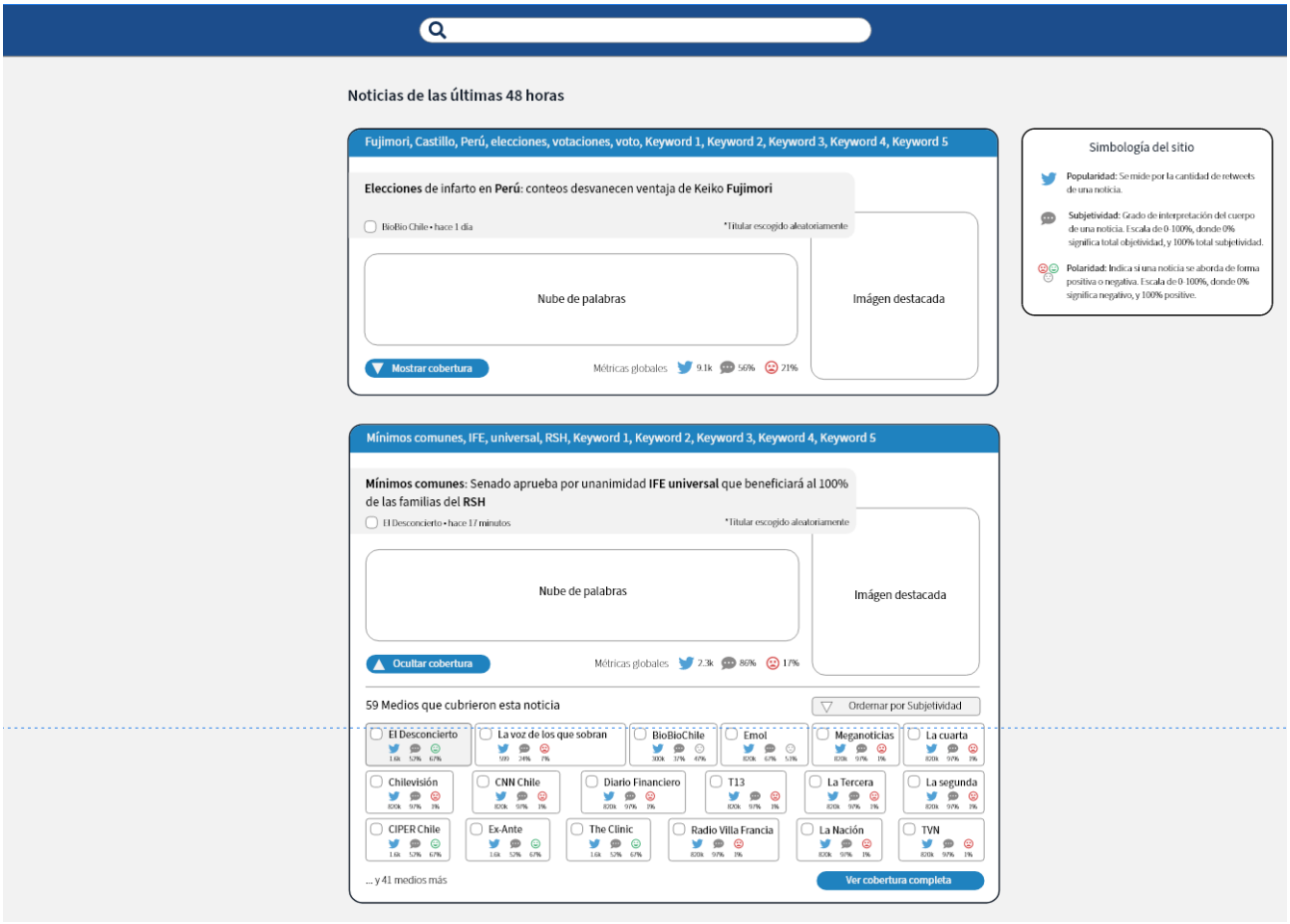

Figura 3.2: *Mockup* de Vista agrupación

La organización visual de esta información intenta tener el formato de *post*, y dejar espacios en blanco para no sobrecargar la vista (Ver Figura 3.2). Lo más relevante está al inicio: palabras claves junto al titular generado, mientras que el resto de la información rodea estos parámetros. Bajo el evento hay dos botones: *Vistazo* y *Detalle*.

El botón *Vistazo* permite ver todos los artículos asociados al evento. Cada artículo esta representado por su medio, título, fecha de publicación, métricas de polaridad (positividad, neutralidad y negatividad), métrica de subjetividad, y métricas de Twitter (*retweets* y *likes*). Las métricas son representadas por un icono, y van acompañas del valor en cada caso. Para visualizar mejor, cada artículo se encuentra agrupado por medio. Se utiliza un artículo representativo arbitrario para el medio y se deja un botón con una flecha vertical para ver los demás artículos de ese medio. Al hacer clic sobre el titular de un artículo, se redirige al sitio externo original.

Sobre la lista de artículos se encuentra un buscador para filtrar por medio, y dos selectores. El primero define un criterio de orden (de menos a más, de más a menos) y el segundo el criterio (*retweets* y *likes*, subjetividad, positividad, neutralidad y negatividad). Con estos pequeños

filtros, se puede ordenar rápidamente la información de un evento de menos a más subjetivo, de más a menos negatividad, etc. Esto satisface el objetivo de mostrar información potencialmente lo más objetiva posible primero.

Finalmente, el botón *Detalle* permite al usuario ir a la Vista detalle de evento que contiene más información.

Vista detalle de evento

Esta vista muestra a un evento con toda la información disponible del mismo y visualizaciones extra (Ver Figura 3.3). Se sigue el mismo formato de evento anterior, y se agregan los parámetros *tags* y categoría bajo el resumen. Ahora, bajo este enunciado hay cuatro secciones: lista de artículos, gráficos de volumen, gráfico de términos más utilizados y gráficos de análisis de sentimiento. Se toma como referencia de la versión de Event Registry (Ver Figura A.2).

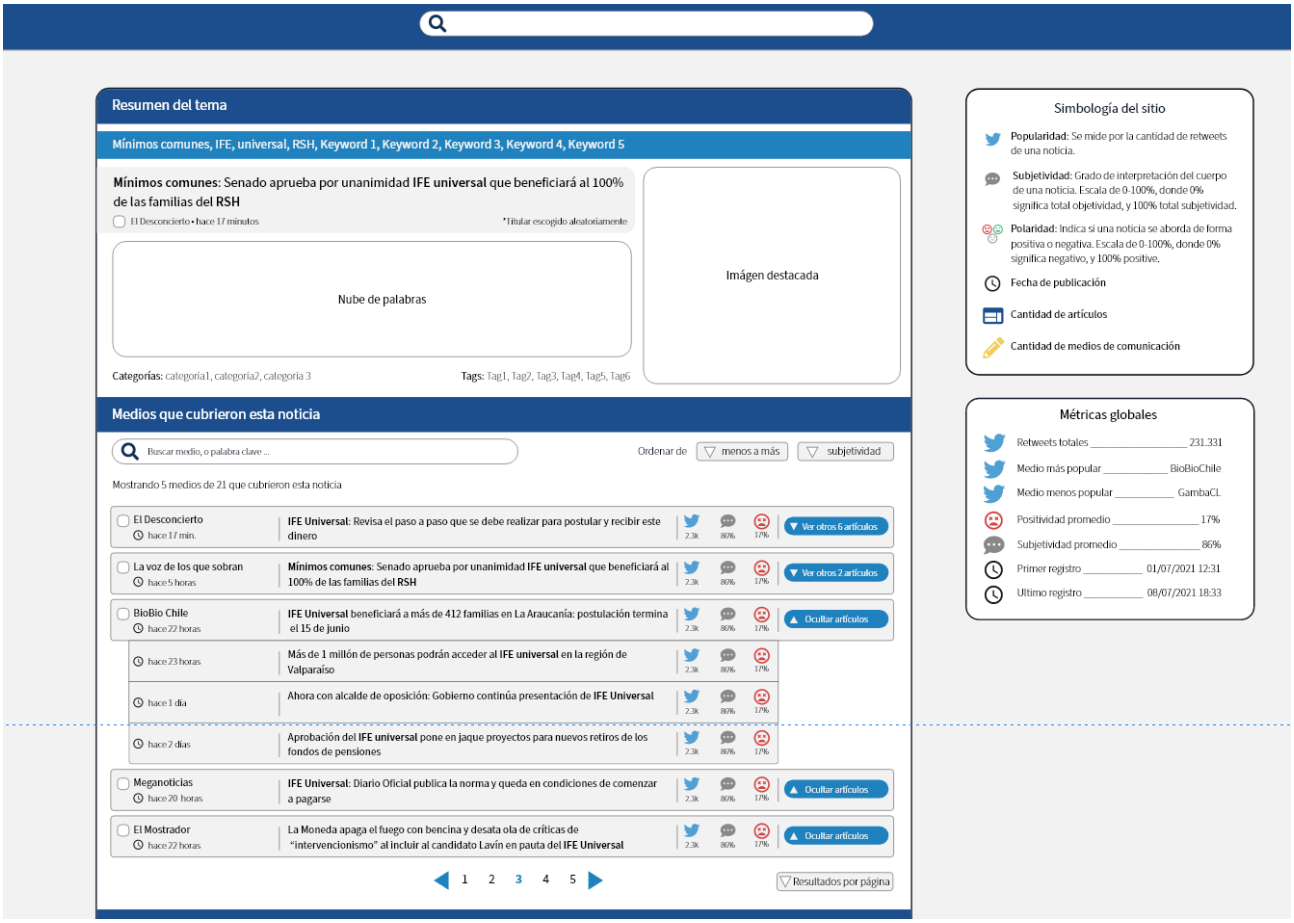

Figura 3.3: *Mockup* de Vista detalle de evento: elementos generales y artículos

La lista de artículos es la misma presente en la vista anterior: se muestran los artículos, con sus métricas, agrupados por medio y con los filtros para ordenar la información.

El gráfico de volumen ofrece dos visualizaciones (Ver Figura 3.4). La primera muestra el número de artículos por día, agrupando los artículos por medio de comunicación, permitiendo ver la evolución de la cobertura en el tiempo, en forma global y para cada medio. La segunda muestra la cantidad total de artículos publicados por cada medio, lo cual permite visualizar qué medio da mayor y menor cobertura a ese evento. También se expone el primer y último artículo de este evento, a modo de mostrar la información más reciente, y más antigua del evento.

Figura 3.4: *Mockup* de Vista detalle de evento: menciones en Twitter y gráficos de volumen

El gráfico de términos más utilizados no es más que un gráfico de barras horizontal que contiene los términos más utilizados en las bajadas y cuerpos de las noticias del evento. Los resultados están ordenados para mostrar directamente los términos más significativos del evento. Este gráfico usa el mismo *input* que la nube de palabras del evento.

El gráfico de análisis de sentimiento aplica un algoritmo sencillo, para catalogar a cada artículo en muy negativo, negativo, neutral, positivo o muy positivo. Para cada artículo, se toma el máximo entre positividad, negatividad y neutralidad. Si el máximo es neutralidad, el artículo queda como neutral. Si es positividad, queda como positivo, y si la positividad es mayor a 0.5, entonces queda como muy positivo. Análogo para la negatividad.

Luego, se grafica esta clasificación en un gráfico tipo torta (o *pie* en inglés) y polar, intercambiable por el usuario mediante un botón superior. La idea central es visualizar directamente la polaridad de un evento en forma global. Esto lo facilita el uso de los colores verde, amarillo y rojo en el gráfico de torta, y en el polar, además del área del gráfico, el radio de cada categoría sirve para dimensionar la magnitud de cada categoría.

Finalmente, se desarrolla un gráfico opcional (Ver Figura 3.5) que junta los histogramas de distribución para los parámetros de subjetividad y polaridad. No se pretende que este gráfico se

use a nivel de usuario, pero sí a nivel de investigación. Conocer las distribuciones, permite evitar búsquedas fuera de rango (al conocer los valores máximos y mínimos), y también tener una percepción aproximada sobre dónde se encuentran concentrados los valores para cada criterio.

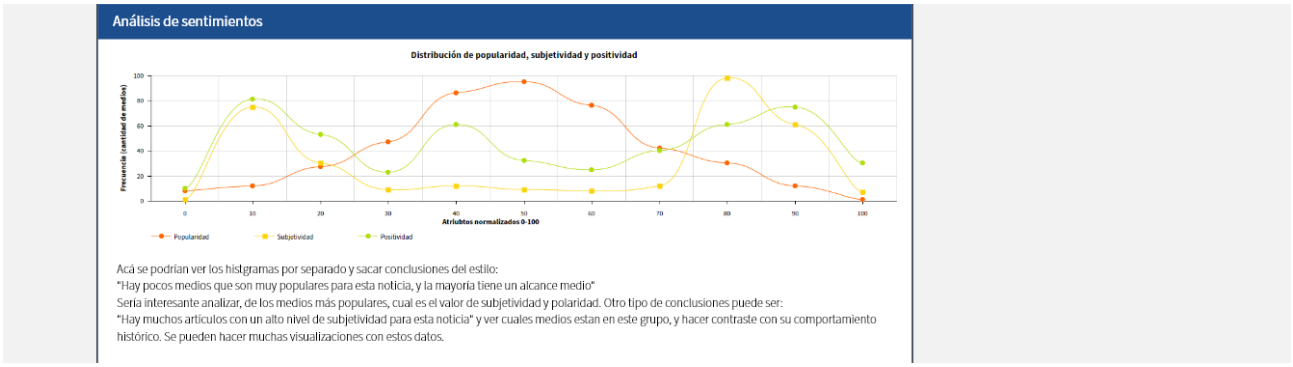

Figura 3.5: *Mockup* de Vista detalle de evento: agrupaciones similares y gráfico de polaridad

Por ejemplo, se puede ver fácilmente que para un evento dado, los valores de negatividad se concentran entre 0.7 y 0.9, y por lo tanto, concluir que la mayoría de los artículos son negativos. O bien, ver que la subjetividad se concentra en los intervalos 0 y 0.2, pero también en 0.8 y 1, por lo tanto, hay medios que abordan el evento en forma tanto objetiva como subjetiva.

Esta vista es la más importante del sistema, pues representa en sí el propósito del *software*. Primero se muestra la información más relevante para un evento mediante el resumen generado. Se visualizan los conceptos clave de los evento. Luego, se muestra la cobertura del evento, donde las noticias se pueden ordenar bajo los criterios de subjetividad y polaridad. Y finalmente, se muestran visualizaciones que analizan al evento en su totalidad.

Vista buscador

Esta vista consta de dos componentes principales: filtros y resultados. La primera parte se compone de filtros de texto, selectores, y campos numéricos que brindan la gama de filtros mencionados en la Sección 3.6. Al final de los filtros se muestra un botón *Buscar*. Se toma como referencia el buscador de Google News (Ver Figura 4.28).

Los resultados se muestran mediante la misma componente de lista de artículos utilizada para las vistas de eventos. Los artículos se agrupan por medio y tienen los filtros simples para ordenar información. Además, se incluyen todos los gráficos desarrollados para la Vista detalle de evento, ya que permiten obtener una información global de los resultados de la búsqueda.

La idea es que al cambiar los filtros y apretar nuevamente el botón *Buscar* se actualicen los resultados. No se considera que al cambiar un filtro la búsqueda se realice automáticamente, para evitar cargas innecesarias a los microservicios.

Esta vista, es la segunda más importante de la plataforma, ya que permite buscar contenido periodístico histórico, y filtrar la información mediante los parámetros enriquecidos de subjetividad y polaridad. Esta vista no cuenta con *mockups*.

Vista informaciones

Esta vista es bastante sencilla, consta de diversos párrafos con explicaciones referentes a: el objetivo de la plataforma, sus secciones y funcionalidades principales, la subjetividad, la polaridad y los parámetros de Twitter.

En un futuro se pretende que esta vista sea mucho más amigable con los nuevos usuarios, que mediante la creación de animaciones o videos, muestre como utilizar el sistema en forma más intuitiva. Esta vista no cuenta con *mockups*.

3.8.3. Comentarios y observaciones

Los *mockups* expuestos anteriormente fueron presentados al equipo Telar del Instituto Milenio Fundamento de los Datos (IMFD)[42], en una reunión que tuvo por objetivo presentar este trabajo, e intentar incorporar datos de noticias obtenidos por el instituto, en especial referentes al proceso constitucional del país.

Se propuso añadir dos secciones más a la Vista detalle de evento: Intervenciones del senado y Debates legislativos históricos. La idea era detectar eventos relacionados con la política, y más específicamente, respecto de la Convención Constitucional, y agregar información histórica respecto a un evento en particular. Por ejemplo, un evento referente a la legislación de derechos de agua, podría además incluir información histórica de debates legislativos anteriores, o videos de intervenciones del senado respecto a este tema. Esta información puede dar más contexto a un evento particular, permitiendo al usuario tener más herramientas para contrastar información.

Sin embargo, esta idea se deja planteada para el futuro, ya que implica resolver varios problemas, para los cuales se carece de tiempo en esta etapa.

Lo positivo de este encuentro fueron los comentarios del equipo Telar, catalogando el presente trabajo de interesante e innovador, pero al mismo tiempo, identificando en el sistema bastante trabajo a realizar, y desafíos que superar.

[42]https://imfd.cl/imfd-presenta-plataforma-telar-investigacion-basada-en-datos-al-servicio-de-la-ciudadania/

Capítulo 4

Implementación

4.1. Metodologías de desarrollo

Se utiliza programación orientada a objetos [23] (*Object Oriented Programming* (OOP) en inglés) en todos los componentes del sistema. El objetivo central es brindar de una mayor flexibilidad y escalabilidad, por ello, esta metodología es la indicada, debido a que es más sencillo modularizar partes del *software*, por ejemplo, creando clases según las responsabilidades de negocios.

También se desarrollan *tests* unitarios para la mayoría de los métodos implementados a modo de garantizar el correcto funcionamiento de cada uno. Esto inspirado en la metodología de desarrollo *Test Driven Development* (TDD) [7], donde se elaboran *tests* unitarios antes de la implementación de los métodos en sí.

4.2. Selección de medios

La primera tarea de la implementación es simple y concreta: definir cuáles son los medios nacionales, y para cada medio encontrar su cuenta de Twitter, dominios de portal web y definir si es tradicional o independiente. La publicación *Power structure in Chilean news media* Bahamonde et al. [46], que analiza la concentración de 411 medios chilenos al año 2018, da información sobre propiedad y nombre de usuarios de cuentas de Twitter en formato CSV[1].

Para obtener el número de seguidores de Twitter basta utilizar la API de Twitter[2] y revisar la cuenta de cada medio. De forma similar, cada medio publica en Twitter el dominio de su portal web, por lo tanto, también es directo obtener esta información. Para usar esta API, es suficiente declarar a Twitter que las solicitudes serán con fines académicos.

Para realizar la distinción entre tradicional e independiente se definen dos criterios: en primer

[1] https://github.com/eelejalde/Chilean_Media_Power_Structure/
[2] https://developer.twitter.com/en

lugar, se emplean fuentes externas que realicen esta clasificación, y en segundo lugar, se usa el principio de que si un medio tiene sólo una entidad propietaria, entonces es independiente.

En marzo de 2021, CIPER Chile publicó un artículo académico titulado *"El ruidoso silencio de los medios tradicionales"* [49], donde se describe el auge por medios alternativos y se expone una tabla que hace la distinción de medios independientes o alternativos. Ver Tabla 4.1.

Medios alternativos	**Medios tradicionales**
Chileokulto	24Horas.cl
El Ciudadano	ADN
El Desconcierto	CHV Noticias
El Dínamo	CNN Chile
El Líbero	Cooperativa
El Obervatodo	La Cuarta
El Periscopio Chile	El Mostrador
Gamba.cl	Emol
Interferencia	Diario La Hora
La Izquierda Diario Chile	La Tercera
Mapuexpress	Meganoticias
OPAL Prensa	Publimetro Chile
Puranoticia.cl	Radio Agricultura
VerdadAhora.cl	Radio Bío Bio
	T13
	The Clinic

Tabla 4.1: Medios alternativos y tradicionales de Chile, consideración de CIPER [49]

Como observación, hay medios que no están incluidos en el archivo CSV y son agregados manualmente, buscando sus cuentas de Twitter también en forma manual. Tal es el caso de Chileokulto[3], InterferenciaCL[4], Puranoticia[5], Piensaprensa[6], DefrenteCL[7], entre otros.

Considerando lo anterior, se desarrolla un breve *script* para guardar toda esta información en un archivo CSV actualizado. Mediante este CSV se ordenan los medios de más a menos seguidores, para aplicar el criterio de selección de impacto.

En la Tabla 4.2 puede apreciarse el orden en el cual se pretenden incorporar los medios de comunicación, ordenando los resultados por número de seguidores en Twitter. La idea es luego aplicar el criterio de diversidad periodística y finalmente el de factibilidad técnica, descritos en la Sección 3.1. La misma tabla adelanta cuáles medios fueron incorporados al sistema. La justificación de esta elección se aborda en la Sección 4.3.

[3]https://twitter.com/Chileokulto
[4]https://twitter.com/InterferenciaCL
[5]https://twitter.com/puranoticia
[6]https://twitter.com/PiensaPrensa
[7]https://twitter.com/defrente_cl

Twitter id	Nombre en Twitter	Dueño(s)	Seguidores	Incluído
24horastvn	24 Horas	Estado de Chile	3.944.433	sí
cnnchile	CNN Chile	CNN Chile	3.638.505	sí
t13	T13	Inversiones Canal 13	3.606.671	no
biobio	BioBioChile	Grupo Mosciatti	3.574.548	sí
cooperativa	Cooperativa	Fundación para las ...	3.186.011	sí
adnradiochile	Radio ADN	Grupo Prisa	2.378.239	no
emol	Emol.com	El Mercurio	2.064.633	sí
latercera	La Tercera	Copesa	2.047.993	no
thecliniccl	The Clinic	Ediciones Bobby	2.037.373	sí
meganoticiascl	Meganoticias	Grupo Bethia	2.025.972	sí
elmostrador	El Mostrador	La Plaza	1.879.771	sí
onemichile	onemichile	Estado de Chile	1.845.713	no
chilevision	Chilevisión	CNN Chile	1.718.318	sí
publimetrochile	Publimetro	Grupo Metro Int...	1.236.029	no
ciper	CIPER Chile	Copesa	1.157.096	sí
lacuarta	La Cuarta	Copesa	880.673	no
laredtv	LaRed	Grupo Albavision	780.142	sí
la_segunda	La Segunda	El Mercurio	632.926	no
el_ciudadano	El Ciudadano	Sociedad Periodi...	526.912	sí
radiocarolina	Radio Carolina	Copesa	487.701	no
eldesconcierto	El Desconcierto	Medios de Edicion...	434.907	sí
nacioncl	La Nación Chile	Com. LANET	343.342	sí
PiensaPrensa	PiensaPrensa	Unknown	306.049	sí
diariolahora	La Hora	Copesa	303.626	no
futurofm	Radio Futuro	Grupo Prisa	273.980	no
DFinanciero	Diario Financiero	Grupo Claro	243.965	no
lun	Las Últimas Noticias	El Mercurio	234.428	no
chileokulto	Chileokulto	Unknown	220.290	sí
gamba_cl	Gamba	Unknown	201.399	sí
prensaopal	Prensa OPAL	Unknown	180.821	sí
interferenciacl	Interferencia	Ed. Interferencia SpA	160.568	sí
uchileradio	Diario-Radio UChile	Universidad de Chile	153.501	no
elquintopoder	El Quinto Poder	Fundacion Democ...	133.836	sí
rvfradiopopular	Radio Villa Francia	Unknown	104.372	sí
grupoperiscopio	El Periscopio.cl	Grupo Periscopio	93.349	sí
elliberocl	El Líbero	Soc. Periodistica El Libero	93.260	no
delosquesobran	La voz de los que ...	Unknown	86.426	sí
radiopalomafm	Radio Paloma Talca	Radiodifusora Paloma	81.774	sí
elrancaguino	Diario El Rancaguino	Soc.Informativa Regional	54.367	sí
ladiscusioncl	La Discusión	U.de Concepción	51.973	sí
conciertoradio	Radio Concierto	Grupo Prisa	51.411	sí
enlineamaule	En Línea Maule	Luis Verdejo Vega	33.994	sí

Tabla 4.2: Medios de comunicación seleccionados

4.3. *Scrappers*

4.3.1. Extracción de *urls* desde Twitter

Se deben obtener las *urls* semilla del muro de Twitter de cada medio. Para esto se hace uso de la API de Twitter, para extraer en formato JSON los últimos 3.200 *tweets* de cada cuenta (200 páginas de 16 *tweets* cada una). Se filtran los *tweets* que corresponden a un *retweet* de otro usuario, para no considerarlos. Luego, para extraer las *urls* basta consultar el parámetro *entities* y *urls*, que contiene todas las *urls* contenidas en un *tweet*.

Estas *urls* vienen generalmente acortadas y se deben resolver. Para esto se desarrolla una función llamada *unshort* que mediante el uso de la librería *requests* de Python, realiza las resoluciones necesarias para revelar la *url* original. Luego, se compara si esta *url* pertenece al dominio del medio mediante el uso de la librería *re* (de expresiones regulares), y de ser así, se incluye en la lista de *urls* semilla. Si una *url* no logra ser resulta en un determinado tiempo, se descarta.

Adicionalmente, se extrae el número de *retweets* y *likes* de cada *tweet* consultando el parámetro *retweet_count* y *favourite_count* (de *likes*) para incluir como métrica de cada artículo. Para actualizar la variación de estas métricas a medida que pasa el tiempo, también se extrae la *id* del *tweet* (con el parámetro *id_str*) para asociar a cada artículo con su *tweet*.

De esta forma, las *urls* semilla (junto al *id* del *tweet* y sus métricas) son extraídas para cualquier medio. No hay mayores inconvenientes en esta parte del sistema.

4.3.2. Portales digitales

Para extraer información desde los portales de noticias se consideran las siguientes herramientas de *scrapping*: Apache Nutch[8], Scrapy[9] y algunas avanzadas como Octoparse[10] y Parsehub[11].

Las últimas dos son bastante profesionales e ideales para hacer *scrapping* a gran escala, ya que tienen diversas interfaces de usuario para hacer este proceso de manera directa y sencilla, incluso ofreciendo servidores en la nube para este proceso. Lo negativo, es que para escalar horizontalmente, se debe pagar por el uso del servicio, comenzando en 75 USD y 149 USD mensuales para Octoparse y Parsehub respectivamente (a fines de 2020). Otras herramientas similares siguen este mismo patrón y es inviable usarlas por un tema de recursos.

Apache Nutch es altamente escalable y gratuito, pero su utilización es un poco compleja. Scrapy es una herramienta gratuita y sencilla de usar en Python, también escalable horizontalmente. Se prueban ambas y se decide utilizar Scrapy, ya que demuestra ser suficiente para los propósitos de este trabajo.

[8] https://nutch.apache.org
[9] https://scrapy.org
[10] https://www.octoparse.es
[11] https://www.parsehub.com

Scrapy es una herramienta orientaba a objetos. Cada *scrapper* es un objeto de la clase *Spider* que hereda de *CrawlSider* y que implementa el método *parse* que recibe la respuesta de un sitio como parámetro[12]. Este método está encargado de extraer y estandarizar la información de cada sitio que visita por medio de *CSS selectors* y *XPaths*. Al mismo tiempo, cada *Spider* tiene un conjunto de reglas (*rules* en inglés) respecto de qué *urls* considerar y cuáles no al momento de visitar un sitio. Con estas reglas se pueden filtrar *urls* fuera de dominio, y especificar qué *urls* seguir, como aquellas contenidas en los paginamientos de noticias.

Se crea la clase *BaseSpider* para colocar entre las clases *CrawlSpider* y *Spider*, que realiza procesos comunes entre todas las *Spider*. En primer lugar, esta clase implementa el método *get_seed_urls* para obtener las *urls* semilla desde Twitter con el procedimiento descrito anteriormente. En segundo lugar, el método *url_belongs_to_domain* para filtrar *urls* que pertenecen al dominio del medio (para *urls* encontradas en forma recursiva y no por Twitter). En tercer lugar, el método *remove_get_params* para remover parámetros tipo GET en las *urls* encontradas, y finalmente, el método *save_article* que guarda el artículo *parseado* en la base de datos.)

La mejor *id* que se puede asignar a un artículo en la base de datos es justamente la *url* del artículo, ya que en la web no pueden existir dos *urls* con contenido distinto. El único problema, es que pueden haber parámetros GET presentes en una *url*, donde dos *urls* muestran el mismo contenido. Para solucionar esto, está el método *remove_get_params*.

Para evitar duplicados en la base de datos y trabajo innecesario, antes de procesar una *url*, se consulta a la base de datos si el artículo con dicha *url* existe y de existir, se ignora. En la Sección 4.6 se detalla este mecanismo de llamadas a Elasticsearch.

Existen medios que no se implementan por problemas técnicos, o bien, por priorizar medios menos masivos, pero de carácter independiente, regionalista o alternativo. Dentro de los motivos técnicos para descartar un sitio se encuentra: que el contenido no sea paginado sino con carga continua (como T13[13]), que el contenido tenga formato de imagen (como Hoyxhoy[14] o La Segunda[15]) y que exista una barrera de subscripción de pago que impide ver artículos (como El Líbero[16] y Diario Financiero[17]).

Una dificultad fue la extracción directa de texto: se fuerza la creación de una función llamada *clean_text* por la presencia de textos repletos de *tabs*, saltos de línea y espacios en blanco al inicio, al final y entre medio de un texto. Esta función remueve todo esto y elabora un texto limpio. Esto se aplica a varios medios como CHV[18], Meganoticias[19] y CNN Chile[20].

[12]https://docs.scrapy.org/en/latest/
[13]https://www.t13.cl
[14]www.hoyxhoy.cl
[15]https://digital.lasegunda.com
[16]https://ellibero.cl
[17]https://www.df.cl
[18]https://www.chvnoticias.cl
[19]https://www.meganoticias.cl
[20]https://www.cnnchile.com

Otra dificultad presente, se relaciona con la estructura de la información, porque medios como CHV o El Desconcierto[21], pueden desplegar el título o cuerpo de la noticia repartido en varios *tags* HTML *<p>*, donde además algunas palabras se escriben en negrita con el *tag <s>* o ** y otras en cursiva con el *tag <i>*. Quizás es interesante preservar estos *tags*, pero no es amigable para la generación de *embeddings* que idealmente recibe solamente palabras. Por esto, se hace uso de expresiones regulares y de la librería Beautiful Soup[22] para eliminar estos *tags*.

Por otro lado, la inclusión de *tweets* en cuerpos de noticias representa otra dificultad. Algunos medios como Meganoticias, suelen poner reacciones de usuarios, o citas de otras fuentes de información en sus artículos. Realizar la distinción entre qué es *tweet* o no, es imposible, ya que todo párrafo está bajo el *tag <p>*. Tal es el problema, que finalmente se considera dejar toda esta información como noticia.

La publicidad es un problema,en medios como El Mercurio[23] es imposible utilizar Scrapy. Por ello, se emplean solicitudes directas a su base de datos mediante el uso de la librería *requests*. Misma suerte la consigue el medio La voz de los que sobran[24], donde es sencillo realizar este tipo de solicitudes.

Se observa que la extracción de artículos es más efectiva para artículos más recientes, que para los más antiguos. Al mismo tiempo, la adquisición de *urls* semilla desde Twitter ayuda a mitigar este efecto.

Otra observación es que no se llega al punto de prohibición de la *ip* de *scrapping*, ya que el proceso es lento para emular lo más fielmente el comportamiento de un usuario. Sin embargo, se hace necesario en un futuro considerar este escenario, debido a que los *firewall* de los diversos sitios pueden identificar la *ip* de *scrapping* como maligna (por ejemplo, considerarla ataque DDoS[25]). Para esto se deben rotar parámetros como el *User-Agent* e *ip*. Lo primero configurable en Scrapy y lo segundo sólo lograble a través el uso de *proxies* y servidores externos, que generalmente son de pago.

A pesar de esta serie de dificultades, se incorporan 27 medios nacionales (Ver Tabla 4.2), de los cuales 24 Horas TVN[26], CNN Chile, BioBioChile[27], Cooperativahttps://www.cooperativa.cl, El Mercurio, Meganoticias y CHV pueden considerarse como tradicionales. El resto de los medios son alternativos, tienen un dueño que no tiene más medios, se autodenominan como independientes, o son regionales.

Es ideal contar con más medios, pero al mismo tiempo se cree que 27 medios son suficientes para demostrar la factibilidad y utilidad de este trabajo.

[21]https://www.eldesconcierto.cl
[22]https://www.crummy.com/software/BeautifulSoup/bs4/doc/
[23]https://www.emol.com
[24]https://lavozdelosquesobran.cl
[25]https://www.cloudflare.com/es-es/learning/ddos/what-is-a-ddos-attack/
[26]https://www.24horas.cl
[27]https://www.biobiochile.cl

4.4. Procesamiento del lenguaje natural

4.4.1. Clases y componentes

El eje principal de desarrollo se enfoca en la flexibilidad, escalabilidad y simpleza del sistema. Incorporar y quitar modelos debe ser simple e intuitivo. Para cumplir con esto se implementan dos clases abstractas principales: *NLPTask* y *NLPModel*. La primera denota una tarea particular, medir polaridad, subjetividad, traducir texto o generar resúmenes. Mientras que *NLPModel* actuará de interfaz para los distintos modelos que ejecutan estas tareas.

La clase *NLPModel* cuenta con dos métodos abstractos: *process_ text* y *process_ text_ batch*. El primero recibe un texto plano y entrega los resultados de tarea PLN específica. El segundo realiza el mismo procedimiento, pero con varios textos simultneamente, en miras de aprovechar el paralelismo que ofrecen algunos modelos. En este caso, de *NLPModel* heredan cuatro clases: *PolarityModel, SubjectivityModel, TranslationModel* y *SummaryModel*. Estas clases, abstractas y de tipo *Model*, sirven para establecer las plantillas estándar de *output* para cada tarea.

Con estas guías, para implementar un modelo basta extender de la clase abstracta tipo *Model* de la tarea que se quiere abordar, y generar un *output* estándar ya definido. De esta forma, la implementación específica de cada modelo queda encapsulada en los métodos *process_ text* y *process_ text_ batch*.

Por otro lado, la clase *NLPTask* recibe en su constructor todos los objetos tipo *Model* que satisfacen una tarea PLN. Cuenta con los métodos implementados *compute* y *compute_ batch*, y dos métodos abstractos *compute_for_ mapping* y *compute_for_ mapping_ batch*. El primero itera sobre todos los objetos tipo *Model* que satisfacen la tarea PLN y genera tantos *outputs* como modelos se tengan, considerando el lenguaje (inglés, español, etc.) de cada modelo. Para ello, el método *compute* recibe tanto el texto a procesar como el lenguaje del texto en cuestión. Por su parte, *compute_ batch* busca procesar una lista de textos en paralelo de manera similar.

Los métodos abstractos deben ser implementados por las clases *Subjectivity*, *Polarity*, *TranslationESEN*, *TranslationENES* y *Summary*, que heredan de *NLPTask*. La idea de estos métodos es generar un *output* específico para cada *task*, que coincida con el *mapping* de artículos y eventos. Un diagrama visual que explica estas relaciones entre clases se observa en la Figura 4.1.

Finalmente, se encuentran la clases *TextProcessing* y *Summarizer*. La primera contiene implementa los métodos *process_ article* y *process_ article_ batch*, que mediante el uso de las clases *Subjectivity*, *Polarity* y *TranslationESEN*, calcula el contenido enriquecido de cada artículo. Por su parte, la clase *Summarizer* utiliza las clases *Summary* y *TranslationENES* para generar el título y resumen de un evento, en los métodos *process_ event* y *process_ event_ batch*.

Este sistema de clases es flexible, ya que permite al usuario incorporar nuevos modelos de cualquier tarea PLN, prácticamente sólo estandarizando el *output* generado. Por otro lado, las clases que heredan de *NLPTask* tienen el control sobre qué modelos utilizar en el sistema, pudiendo eliminar y añadir modelos en forma sencilla. Esto se apoya en el *mapping* realizado tanto para artículos y eventos, donde los resultados son generalmente listas de objetos, dan la flexibilidad suficiente para agregar y quitar modelos.

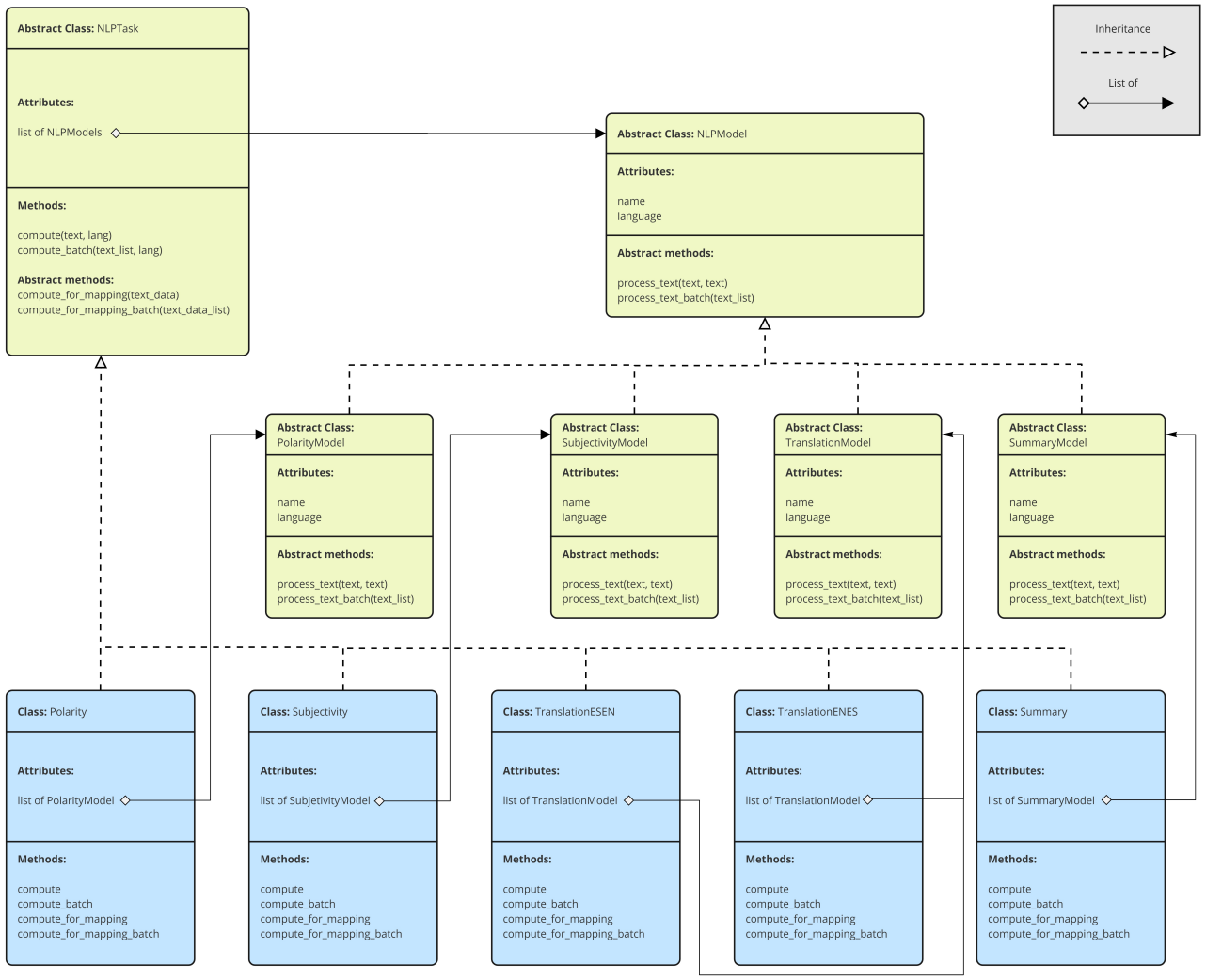

Figura 4.1: Diagrama UML de clases asociadas a tareas PLN

El sistema es también eficiente, debido a que busca aprovechar el procesamiento en *batches* que brindan algunos modelos, y porque algunos modelos permiten su ejecución en GPU. Por último, las clases permiten al sistema escalar horizontalmente, porque basta con generar múltiples instancias de *TextProcessing* y *Summarizer* para procesar más artículos y eventos.

4.4.2. *Machine translation*

Se pretende traducir texto del español al inglés, para usar modelos de polaridad, subjetividad y resumidores en ese lenguaje, y viceversa también, para traducir el *output* del resumidor a español. Como se adelanta en la Sección 3.3.4, se utilizan los modelos *"opus-mt-es-en"*[28] y *"opus-mt-en-es"*[29] de la Helsinki NLP alojados en Hugging Face.

[28]https://huggingface.co/Helsinki-NLP/opus-mt-es-en
[29]https://huggingface.co/Helsinki-NLP/opus-mt-en-es

Cada modelo tiene un BLEU *score* sobre 35 para múltiples *datasets*, y sobre 54 para el *dataset Tatoeba-test.spa.eng*[30] y *Tatoeba-test.eng.spa*[31] en casa caso.

Su uso es bastante directo, mediante las clases *AutoTokenizer* y *AutoModelForSeq2SeqLM*, se pueden realizar los dos procesos principales de cualquier modelo basado en *trasformers*: *tokenization* y el procesamiento. La primera clase permite transformar el texto plano a *word embeddings* junto a los *positional encodings*, mientras que la segunda emplea estos *embeddings* para obtener el *output* deseado. En este caso, los modelos usan tanto el proceso de *encoding* como de *decoding*.

Un gran detalle es la traducción de grandes textos, ya que los modelos aceptan *inputs* de hasta 128 palabras aproximadamente. Para abordar esto, cada oración de un texto largo es diseccionada, y luego concatenada para formar partes de no más de 128 palabras cada una, teniendo en consideración no trucar ninguna oración. Luego, este conjunto de oraciones queda listo para su procesamiento en *batches* o en forma secuencial.

Lo destacable de este modelo es que permite su uso en la GPU, que está optimizada para el cálculo de matrices. Así, el proceso completo es siguiente: el modelo se carga en la GPU, se generan *embeddings* en CPU, se traspasan de la CPU a la GPU, se realizan los cálculos, y finalmente, estos son movidos a la CPU. Para que esto sea posible, la GPU debe ser compatible con CUDA (afortunadamente lo es) y se debe instalar tanto la librería pytorch con CUDA[32] como el paquete de herramientas CUDA de NVIDIA[33]. Luego, el proceso de implementación es estándar y se sigue sin mayores problemas.

Se procede entonces a encapsular el funcionamiento logrado dentro de las clases *HelsinkiESEN* y *HelsinkiENES*, implementando los métodos *process_text* y *process_text_batch*. Para la implementación del segundo método, basta pasar una lista de *inputs* a los *Tokenizers* y modelos, ya que estos manejan el procesamiento de varios *inputs* en sus clases. Luego, las clases *TranslationESEN* y *TranslationENES* incorporan las clases antes mencionadas para ejecutar la tarea de traducción con todos los modelos disponibles, en este caso uno para cada tarea.

Para verificar el funcionamiento de estas clases y a modo de preevaluar los resultados se realizan dos experimentos. Ambos consisten en traducir una noticia al inglés, y luego nuevamente al español. La idea es contrastar si este procedimiento es capaz de recrear el texto original en español. El resultado de estos experimentos se encuentra en Anexo C.2. El primer texto consta de un conjunto de oraciones extraídas de artículos en torno a la inauguración de la Convención Constitucional. El segundo una noticia textual de CNN este mismo hecho[34]. Se adjuntan para cada caso el texto original, el texto intermedio en inglés, y el texto final en español.

De ambos experimentos es posible evidenciar al menos cuatro cosas: (1) el texto intermedio es una buena traducción al inglés, (2) las entidades no pierden su nombre original, y (3) el texto final es un parafraseo de las oraciones originales.

Se puede decir que en estos casos es una buena traducción al inglés, porque al leer el texto en

[30]https://github.com/Helsinki-NLP/Tatoeba-Challenge/blob/master/models/spa-eng/README.md
[31]https://github.com/Helsinki-NLP/Tatoeba-Challenge/blob/master/models/eng-spa/README.md
[32]https://pytorch.org/get-started/locally/
[33]https://developer.nvidia.com/cuda-zone
[34]https://www.cnnchile.com/pais/rojas-vade-detenidos-18-o-apoyo_20210705/

este idioma el mensaje entregado es prácticamente el mismo, y esta observación la respalda el BLEU *score*. Por otro lado, es interesante evidenciar cómo las entidades "Convención Constitucional", "Congreso Nacional"o "Lista del Pueblo" son traducidas a *"Constitutional Convention"*, *"National Congress"* y *"the People's List"*, y después de vueltas al mismo texto original. En contraste con "Plaza de la Constitución" que nunca es traducida, o por ejemplo, la entidad "Comisaría Virtual" que es traducida a *"Virtual Police Station"* y que luego es traducida nuevamente a "Estación Virtual de Policía". Notar también que "Lista del Pueblo" es traducido a "Lista Popular" en el segundo párrafo, creando una ambigüedad.

Es interesante, porque si se traducen estas entidades sin contexto, se obtienen resultados muy distintos. Por ejemplo, el caso de la "Lista del Pueblo" ya mencionado, o el caso de "Plaza de armas" que es traducido a *"Arms square"* y de vuelta a "Brazos cuadrados". Incluso una leve variación en la entidad original, da resultados adversos, por ejemplo, "Lista del pueblo", usando "p" en minúscula, es traducido a *"Town list"* y luego a "Lista de ciudades". Por ello, se tiene la hipótesis de que el modelo usa el contexto de una frase al momento de traducir, lo cual es esperable de un modelo tipo *transformers*, y que además identifica que un conjunto de dos a tres palabras escritas con mayúscula, son probablemente entidades, pero al mismo tiempo, no es clara la forma en que decide usar una traducción u otra como en el caso de "Lista del Pueblo".

Por último, el parafraseo es esperable, debido a la naturaleza de la tarea que se está llevando a cabo, ya que es imposible que, incluso traductores profesionales, traduzcan textos de la misma forma. Lo importante es que en los experimentos hechos, el sentido del mensaje no cambió. De esta forma, estos modelos de *machine translation* se instalan como una buena alternativa para el sistema. Queda entonces, la tarea de evaluar su rendimiento frente a distintos casos de uso y no solamente ante un par de casos aislados.

4.4.3. Polaridad

Se utilizan tres modelos para abordar esta tarea: VadeSentiment[35], SentiStrenght[36] y *RoBERTa polarity*[37] [59]. Los primeros dos modelos se basan en *lexicons*, mientras que el tercero corresponde a un modelo tipo *trnasformers*.

Para implementar cada uno, se crean las clases *VaderSentimentPolarity*, *SentiStrenghtPolarity* y *RoBERTaPolarity*, que heredan de *NLPModel* y por lo tanto, implementan los métodos *process_ text* y *process_ text_ batch*.

VaderSentiment es directo de usar, porque basta con importar la librería, generar un objeto denominado *SentimentIntensityAnalyzer* y luego llamar al método *polarity_ scores* para obtener los resultados deseados. Esto se pone dentro del método *process_ text* y se ejecuta en forma iterativa en *process_ text_ batch* ya que no existe la forma de procesar *inputs* en *batches* para este modelo. Notar que este modelo retorna un resultado denominado *compound* que corresponde a la suma de los *score* de positivdad, negatividad y neutralidad, normalizada entre 0 y 1.

[35]https://github.com/cjhutto/vaderSentiment
[36]http://sentistrength.wlv.ac.uk
[37]https://huggingface.co/cardiffnlp/twitter-RoBERTa-base-sentiment

SentiStrenght se encuentra empaquetado en un archivo *.jar* (de Java) y se interactúa con él mediante comandos en consola. Para esto, se implementa la función en Python *rate_sentiment* que mediante las librerías *subprocess* y *shlex* simplifica este proceso. Un punto positivo de este modelo es que soporta un vocabulario personalizado. En este caso, se emplea un vocabulario en español obtenido del *software* de uso libre Weka[38]. Este modelo tampoco permite el procesamiento en *batches*, pero cabe destacar que es un modelo altamente eficiente, ya que permite procesar cerca de 17 líneas por segundo.

Por último, *RoBERTaPolarity* corresponde a un modelo alojado en Hugging Face. Su uso consta de dos etapas principales: la *tokenization* y la predicción del modelo, para lo cual hay dos clases: *AutoTokenizer* y *AutoModelForSequenceClassification*. La primera transforma el texto plano al *word embedding* junto al *positional encoding*, mientras que la segunda usa los *embeddings* para realizar la predicción a través del *stack* de *encoders* de RoBERTa.

Además, como el cálculo es matricial, se pueden procesar varios *inputs* al mismo tiempo, para lo cual no se necesitan mayores modificaciones. Un detalle, es que todos los *inputs* deben tener el mismo largo. Para esto se toma el *input* más largo como referencia, y a los más pequeños se les agrega un token [PAD] (por *padding*), para completar el largo del *input* más extenso. Este *padding* no afecta el *output*, ya que se considera espacio vacío[39].

Se hacen experimentos procesando 4.000 *inputs*, con repeticiones de las siguientes frases *dummy*: "Hello", "Bye", "Good morning" y "Bad guy", procesadas en forma secuencial y en *batches* de a 20, para el modelo cargado en CPU y GPU. El tamaño del *batch* no puede ser mayor a 20 porque no es posible cargarlo en la memoria de la GPU.

Los resultados expuestos en la Tabla 4.3 evidencian que es mejor utilizar el modelo GPU que en CPU. Sin embargo, no es directo que el procesamiento en *batches* sea más eficiente. Se tiene la hipótesis de que esto sucede porque el proceso de cargar los *embeddings* en la GPU y descargar los resultados a la CPU es un proceso que no compensa el procesamiento en *batches* de a 20. Quizás para un tamaño de *batch* mayor, este costo se compense, pero para probar eso se necesitaría de una máquina más poderosa.

	Modelo en CPU tiempo total [s]	**Modelo en GPU tiempo total [s]**
Secuencial	185.8	55.9
Batches de 20	1319	97.9

Tabla 4.3: Rendimiento de *RoBERTa polarity* en CPU vs GPU para 4.000 *inputs*

La clase *Polarity* es la encargada de cargar los tres modelos de polaridad, y mediante el método *compute_for_mapping* procesar el texto en cada modelo. Se descarta el uso de *compute_batch_for_mapping* dado que para la actual máquina se empeora el rendimiento. Notar que *Polarity* y su clase padre *NLPTask* manejan el lenguaje del texto ingresado y evitan que por ejemplo, modelos de polaridad en español reciban un *input* de texto en inglés.

[38]https://www.cs.waikato.ac.nz/ml/weka/
[39]https://huggingface.co/docs/transformers/preprocessing

Finalmente, se hace una experimentación breve con estos tres modelos, y es posible observar cómo frases con palabras como "éxito", "bueno" y "cariño" son catalogadas como positivas, y aquellas con palabras cómo "pérdida", "malísimo" y "horrible" son catalogadas como negativas. No se realiza una experimentación extensa en esta parte, sólo se verifica que los tres modelos funcionen en el sistema. En la Sección 5 se efectúa una evaluación por casos de uso en detalle.

4.4.4. Subjetividad

Como se menciona en la Sección 3.3.2 de diseño, no hay modelos en Hugging Face para esta tarea. Sin embargo, la implementación de alternativas como TextBlob[40] (modelo basado en *lexicons*) es bastante directa y se decide utilizar como base. La idea es tener un modelo base y luego poder evaluar el uso de más modelos de acuerdo a su complejidad de implementación. Sin embargo, no se implementan otras alternativas debido al escaso tiempo disponible.

Usar TextBlob es bastante sencillo: se importa la librería, se crea un objeto *TextBlob* al cual se le pasa un texto a analizar, y luego se llama al atributo *sentiment* del objeto para obtener la subjetividad y polaridad del texto ingresado.

Esto se encapsula dentro la clase *TextBlobSubjectivity*, siguiendo el patrón de casos anteriores. Luego, la clase *Subjectivity* se encarga de incorporar este modelo en el cálculo de resultados.

Se efectúa un análisis cualitativo sobre algunos titulares aislados. Se observa que el comportamiento no es ideal, pero que sirve como base. Por ejemplo, titulares como "30 años de ignorantización en esta elección presidencial" se califican con una subjetividad de 0, aún cuando presenta adjetivos calificativos. Quizás la palabra "ignorantización" no está en el *lexicon* del modelo. O también, "30 casos confirmados en Chile: Estos son los síntomas de la variante Ómicron" que es calificado con subjetividad de 1, y donde no hay trazas de opinión ni interpretatividad.

Sin embargo, titulares con cuñas como "Otro importante club se suma al interés por Fernando Zampedri" o "No voy a votar por el que va a votar usted. . . ": La ácida respuesta de Julio César que sorprendió a Roberto Cox" sí son capturados como subjetivos con un *score* de 1.0, al usar las palabras "importante", "interés", "ácida" y "sorprendió". También el titular "Proyecto minero en Argentina desata violentas protestas por riesgo de contaminación con cianuro" que contiene la palabra "violentas" da un *score* de 1.0.

En contraste, titulares como "Gendarmería despide a funcionario por fuga de reo que mató a su expareja", "Autoridades recomiendan comprar y utilizar guirnaldas navideñas certificadas" y "Dua Lipa en Chile 2022: Día, hora y cómo comprar entradas para el concierto" son calificados con un *score* de 0.

Esta breve experimentación previa muestra que el modelo funciona en algunos casos, pero que también detecta falsos positivos y falsos negativos. Para evaluar de mejor forma el modelo, se debe medir la precisión del mismo frente al *dataset* SUBJ, midiendo los errores cuadrados en las predicciones. Sin embargo, esto no se hace por falta de tiempo y por no ser el enfoque del trabajo. El objetivo no es reevaluar los modelos incorporados en el sistema, sino más bien, que

[40]https://textblob.readthedocs.io

el sistema sea capaz de incorporar cualquier modelo. En la Sección 5 se efectúa una evaluación cualitativa con más casos de uso.

Finalmente, es interesante contrastar este modelo de *lexicons* con uno basado en *transformers*. Para esto se puede, por ejemplo, realizar un ajuste (*fine-tuning*) de SBERT para el *dataset* de SUBJ, tal como lo hacen los autores al evaluar SBERT en SentEval [80]. No debe ser un proceso complejo, pero no se concreta al no ser el foco de este trabajo.

4.4.5. Resumidor

Se prueban tres modelos de Hugging Face para el español: *bert2bert_ shared-spanish-finetuned-summarization*[41], *bert2bert_ shared-spanish-finetuned-muchocine-review-summarization*[42] y *mt5-small-spanish-summarization*[43].

Y dos modelos para el inglés, también de Hugging Face. Por un lado *google/pegasus-xsum*[44][99] y por el otro *facebook/bart-large-cnn*[45][55]. Notar que todos estos modelos son del tipo abstractivo y no extractivo.

El uso directo de cada uno es relativamente sencillo, ya que todos utilizan clases de Hugging Face similares a las utilizadas anteriormente para otras tareas.

Dado que los modelos tienen un tamaño límite de texto que pueden procesar (cerca de 210 palabras por vez). Para procesar textos largos se implementa el siguiente procedimiento: el texto original se divide en oraciones, y se forman grupos de oraciones que no excedan las 200 palabras en conjunto. Se resumen estos conjuntos (*batch*) de oraciones en forma secuencial y luego el *output* final se concatena. Se repite el proceso hasta lograr un *output* de tamaño deseado. En este caso se generan resúmenes de hasta 200 palabras únicamente.

Para probar los modelos de manera preliminar, se efectúa un breve experimento. Se crea un texto respecto al evento de inauguración de la Convención Constitucional, a partir de diferentes oraciones tomadas de distintos medios. Luego, este texto es tomado como *input* para los tres modelos al español. Para los últimos dos modelos en inglés, el texto al original se traduce al inglés utilizando la clase *TranslateESEN*, se genera el resumen, y luego se traduce de vuelta al español apoyándose en la clase *TranslateENES*. Finalmente, estos modelos deben realizar un resumen de no más de 250 palabras únicas, mediante el procedimiento descrito. La idea es evaluar de manera cualitativa el *output* de cada procedimiento.

Los resultados de este experimento se encuentran adjuntos en Anexo C.3. De ellos es posible concluir que los tres modelos en español dan resultados totalmente deficientes. Por ejemplo, el *output* del segundo modelo es la frase "Una declaración de principios, pero más bien, una declaración de intenciones...", mientras que el *output* del tercero es "los convencionales del pueblo mapuche, el pp y el gobierno llama". Lo cual no rescata en ningún sentido la idea central del

[41]https://huggingface.co/mrm8488/bert2bert_shared-spanish-finetuned-summarization
[42]https://huggingface.co/mrm8488/bert2bert_shared-spanish-finetuned-muchocine-review-summarization
[43]https://huggingface.co/josmunpen/mt5-small-spanish-summarization
[44]https://huggingface.co/google/pegasus-xsum
[45]https://huggingface.co/facebook/bart-large-cnn

texto original. Por su parte, *output* del primer modelo es "El ministro del Interior dice que a los constituyentes "los pueden acompañar algunas personas, siempre y cuando tengan su pase de Movilidad o su permiso en Comisaría Virtual ", lo cual tiene un poco más de sentido, pero que, de igual forma, está mal escrito.

En contraste, el *output* del primer modelo en inglés es: "Cuatro partidos políticos en Chile han llamado a sus seguidores a participar en una serie de marchas en Santiago el domingo 4 de julio", que incluso podría servir como titular.

Por su parte, el resultado del segundo modelo en inglés es: "Los convencionales del pueblo mapuche, la Lista del Pueblo, el PS y la FA llamaron a sus adherentes a caminar con ellos desde diferentes puntos de la capital. El Ministro del Interior indicó que los constituyentes "pueden estar acompañados por algunas personas, siempre que tengan su Pase de Movilidad o permiso en la Estación Virtual de Policía", que justamente rescata dos puntos importantes del texto original.

En vista de este breve experimento, ambos modelos basados en inglés dan resultados prometedores, mientras que los basados en español dan malos resultados. El tiempo de ejecución de este experimento es superior en comparación a otras tareas PLN (3 min aprox.). Esto se debe a la cantidad de texto que se debe procesar. Por este motivo, se decide escoger un sólo modelo resumidor.

Y basándose únicamente en este caso, se decide utilizar el modelo *facebook/bart-large-cnn*, dado que el resumen generado para este experimento en particular contiene un poco más de información que el generado por *google/pegasus-xsum*. Mencionar que el modelo *facebook/bart-large-cnn* fue entrenado con un *dataset CNN Daily* de noticias de CNN[46], y que *a priori* es más adecuado para el presente trabajo.

Si bien, esta decisión se toma basándose en un caso aislado y sesgado, se tiene la hipótesis de que si los modelos no tienen un buen comportamiento para un caso sencillo como este, entonces no lo tendrán para casos más complejos donde se necesite procesar aún más texto. Esto no limita que en un futuro el modelo resumidor no se pueda cambiar, ya que el sistema de clases permite incorporar nuevos modelos en cualquier minuto. La idea es usar este modelo como *baseline* en el sistema.

Se encapsula el procedimiento en la clase *FacebookSumarizer* que hereda de *SummaryModel*. Después, la clase *Summary* se encarga de incluir a esta clase para el cálculo de resúmenes.

Una observación importante es que el procedimiento implementado para procesar textos largos es ineficiente e inadecuado. En primer lugar, esta forma de abordar el problema hace que tanto la memoria como los recursos CPU crezcan en forma cuadrática respecto al largo del *input*. En segundo lugar, al crear conjuntos disjuntos de textos, puede perderse información o contexto al procesar un *batch*. En otras palabras, se pierden las *residual conections* propias de la arquitectura *transformers*. Este problema se conoce como *context fragmentation*.

Sin embargo, existen diversos acercamientos para afrontar este problema, por un lado *Reformer* [51] logra bajar la complejidad de la operación de $O(n)$ a $O(nlog(n))$, por otro lado

[46]https://huggingface.co/datasets/cnn_dailymail

Transformer-XL [24] propone una manera de afrontar el *context fragmentation* sin sacrificar tiempo de ejecución y logrando buenos resultados. Finalmente, *Longformer* [9] propone otra forma de procesar grandes *inputs* pero escalando linealmente. La idea es que en un futuro se utilicen estos mecanismos para obtener los mejores resultados posibles al momento de resumir.

4.4.6. *Preprocess service*

Las clases antes implementadas permiten ejecutar las tareas PLN de manera simple y sirven de base para este servicio. El *Preprocess service* está encargado de recibir un artículo con información base (titular, cuerpo, etc.) y generar todas las métricas PLN y procesos de *machine translation* asociados.

Los servicios se comunican a través de solicitudes HTTP. Se usa Flask[47]: *framework* liviano y sencillo, para generar procesos que puedan enviar y recibir solicitudes HTTP a determinados *endpoints*. Para comunicarse con Elasticsearch, se crea un objeto *QueryMaker* (detallado en la Sección 4.6.1) que permite abstraer las llamadas HTTP y realizar operaciones CRUD sobre artículos, eventos y agrupaciones. Para el procesamiento PLN y las traducciones se crea el objeto *TextProcessing*, que mediante el método *process_ text* efectúa todos los cálculos necesarios para un artículo. Finalmente, cambia el *flag preprocessed* del artículo a verdadero.

En particular, este servicio implementa un solo *endpoint* que en la ruta */preprocess* recibe un *article_ id*. Se trae al artículo mediante el *QueryMaker*, se calcula el contenido PLN mediante *TextProcessing*, y luego se guarda nuevamente utilizando el *QueryMaker*.

Este *endpoint* tiene la limitación de procesar un artículo a la vez. La idea es tener un segundo *enpoint* que pueda procesar una lista de *article_ ids* y aprovechar los métodos que procesan texto en *batches* (aunque de momento no sean óptimos). Pero esto no se implementa por falta de tiempo.

El *endopint* desarrollado muestra ser suficiente para las funciones del sistema. Sin embargo, es capaz de procesar un artículo en 20 segundos aproximadamente, lo cual no es para nada ideal dado el volumen de datos que se pretende procesar. A pesar de esto, en el escenario ideal, una escalabilidad horizontal es capaz de abordar este problema (más procesos/servidores).

4.4.7. *Summary service*

Como la generación de resúmenes es un proceso costoso y demoroso, se crea el *Summary service*. Este servicio implementa mediante Flask un *endpoint* denominado */summary* que recibe un *event_ id*, mediante un *QueryMaker* trae el evento y todos los artículos asociados al mismo. Posteriormente, mediante la clase *Summarizer* y el método *summarize_for_ mapping*, genera un resumen del evento tomando como *input* la bajada y cuerpo concatenada de todos los artículos pertenecientes al evento. También se genera un titular a partir de todos los demás titulares del evento. Se cambia el *flag has_ summary* del evento a verdadero y finalmente, el evento enriquecido se guarda en Elasticsearch mediante el *QueryMaker*.

[47] https://flask.palletsprojects.com/en/2.0.x/

Este proceso demora en promedio 10 minutos de ejecución, para un evento con cinco artículos. Este es sin duda el cuello de botella más grande de la aplicación y la única forma de afrontar esta situación es mediante el uso de hardware especializado: múltiples GPU de última generación trabajando en paralelo.

4.5. *Clustering*

4.5.1. Método con *embeddings*

Experimentación con BETO

Se realizan algunas experimentaciones con el modelo BETO[16] que vale la pena adjuntar en el presente documento, dado que fueron precursoras y claves para desarrollar la metodología final mediante el uso de *embeddings*.

La idea central es hacer uso del *output* de BETO junto a la capa de salida *Pooling* de SBERT [80] para el inglés. Bajo la teoría de que esta última capa no hace más que promediar y agregar peso a ciertos parámetros del *output* de BERT para inglés, esta ponderación podría funcionar para el español.

En primer lugar, se selecciona una muestra de 836 titulares recolectados desde tres medios, entre los días 12 y 14 de diciembre de 2021, inclusive. La distribución de artículos no es uniforme entre los medios y se muestra en la Figura 4.2.

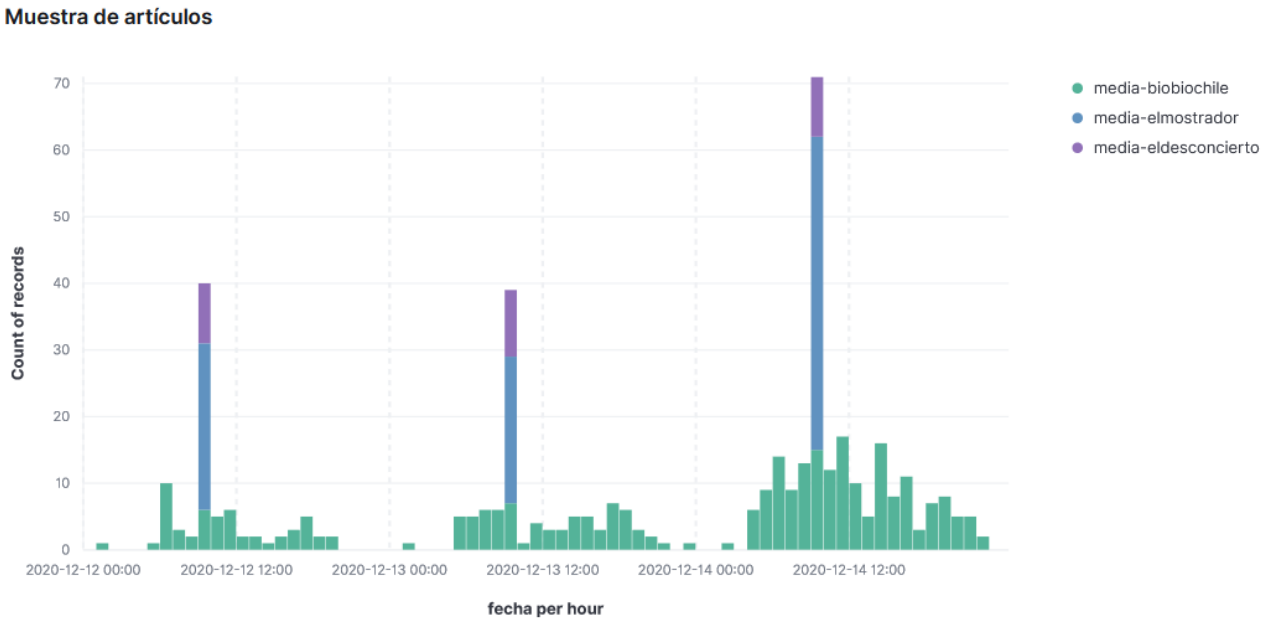

Figura 4.2: Distribución de artículos por medio, en muestra para experimentos

Con estos titulares se realizan tres tareas: (1) se grafican los *embeddings* reduciendo su dimensionalidad a dos, (2) se listan los diez pares de titulares más similares aplicando distancia coseno, y (3) se usa *Agglomerative clustering* con métrica euclidiana y método *ward* para visualizar el dendrograma asociado y formar *clusters* para un límite (*threshold*) estimado.

La primera visualización se encuentra en la Figura 4.3. A simple vista no parecen haber *clusters* densos ni definidos, sin embargo, esta visualización es poco fiable porque la dimensionalidad original de los *embeddings* es mucho más alta (128 dimensiones).

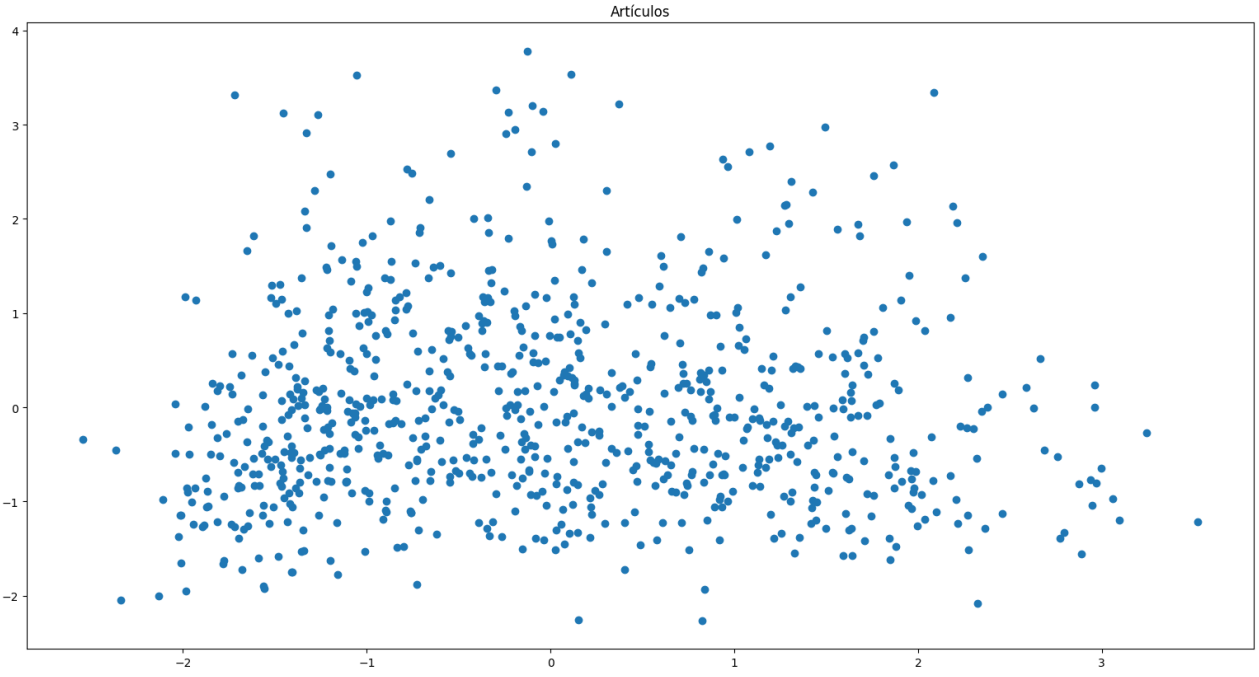

Figura 4.3: Visualización de *embeddings* de 836 titulares reduciendo *embeddings* a dos dimensiones

Los resultados del segundo experimento se encuentran tabulados en la Tabla 4.4. Analizando, es evidente notar que el procedimiento logra identificar los titulares que son prácticamente iguales o parafraseados. Esto valida en cierta forma la hipótesis inicial, ya que *embeddings* similares quedan en un vecindario similar, sobre todo si utilizan las mismas palabras, en casi el mismo orden. Sin embargo, esto no significa que dos *embeddings* con la misma semántica, pero escritos en forma distinta queden en el mismo vecindario, y esta es la principal debilidad de esta metodología.

Se procede con el *Agglomerative clustering*. Se emplea la métrica euclidiana para poder utilizar el método *ward* para medir distancias entre *clusters*. El dendrograma mostrado por la Figura 4.4 deja en evidencia que no es claro el *threshold* de distancia máxima para separar *clusters*. Sin embargo, se estima que en tres días de noticias debe haber al menos 50 hechos noticiosos distintos, se utiliza un *threshold* de 10 para así formar 67 *clusters*.

Medios	Titulares	Distancia coseno	Misma noticia
[BioBio Chile] [El Mostrador]	- Científicos desarrollan técnica para rastrear origen de covid-19 en cuatro horas - Científicos desarrollan técnica para rastrear origen de Covid-19 en 4 hora	0.9972	Sí
[El Mostrador] [BioBio Chile]	- Joe Biden y Kamala Harris elegidas las "personalidades del año" por la revista Time. - Joe Biden y Kamala Harris son las "personas del año" de la revista Time.	0.9756	Sí
[El Mostrador] [El Mostrador]	- Ministros Bellolio y Delgado rechazan proyecto de indulto general para presos del estallido social. - Presidenta del Senado anuncia proyecto de indulto general para presos del estallido social.	0.9693	No
[BioBio Chile] [El Mostrador]	- Human Rights Watch dice que "no hay presos políticos en Chile" y critica indulto general. - Director de Human Rights Watch: "No creo que existan en Chile presos políticos".	0.9942	Sí
[BioBio Chile] [El Desconcierto]	- Detienen a carabinero acusado de robar un vacuno que llevaba en un camión "clonado". - Carabinero queda en prisión preventiva por transportar ganado robado en un camión clonado.	0.9679	Sí
[BioBio Chile] [BioBio Chile]	- Canadá da luz verde a vacuna de Pfizer-BioNtech contra covid-19. - México se suma: regulador sanitario aprueba vacuna de Pfizer-BioNTech contra covid-19.	0.9652	No
[El Mostrador] [El Desconcierto]	- Exalcaldesa Karen Rojo no podrá ejercer por 5 años cargos públicos tras ser inhabilitada por el Tribunal Electoral. - Karen Rojo sancionada por el Tribunal Electoral Regional: Ex alcaldesa de Antofagasta no podrá ejercer cargos públicos durante cinco años.	0.9643	Sí
[BioBio Chile] [El Desconcierto]	- The Guardian eligió a Christiane Endler como la mejor arquera del mundo en 2020. - El gran año de Tiane Endler: The Guardian la elige como la mejor arquera del mundo este 2020.	0.9631	Sí
[BioBio Chile] [El Desconcierto]	- Diputados aprueban legalizar el aborto en Argentina y la iniciativa pasa ahora al Senado. - Ley de Aborto en Argentina da su primer paso: Diputados la aprueban y queda en manos del Senado.	0.9615	Sí
[El Mostrador] [El Desconcierto]	- Caso fiscal Chong: Corte de Apelaciones revoca libertad de siete de los ocho formalizados y los deja con prisión preventiva. - Todos en prisión preventiva: Corte revoca libertad para siete formalizados por amenazas a fiscal Chong.	0.9593	Sí

Tabla 4.4: Titulares más similares de acuerdo a *embeddings* con BETO y métrica coseno

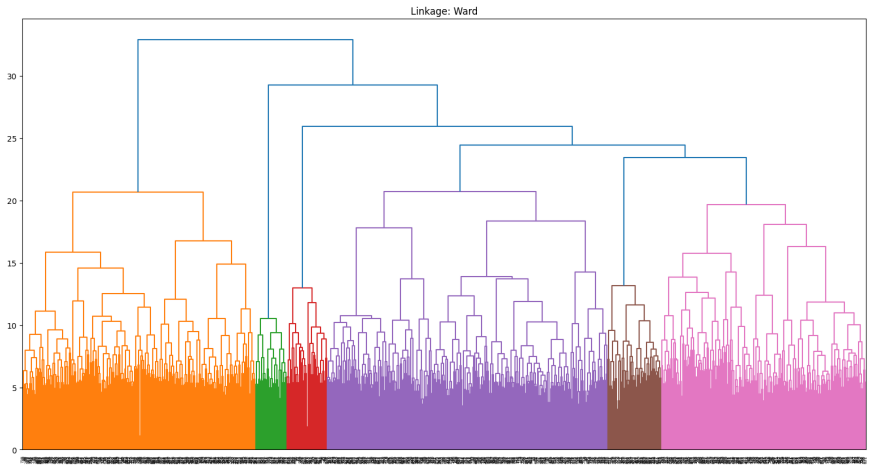

Figura 4.4: Dendrograma para 836 titulares utilizando el método *ward*

Algunos de los *clusters* formados se encuentran en la Tabla 4.5. A simple vista puede concluirse que la agrupación no es tan dispersa, ya que los titulares que pertenecen al mismo grupo contienen, en general, las mismas palabras en casi el mismo orden. Esto es esperable, por la misma razón anterior, donde *embeddings* que usen las mismas palabras en casi el mismo orden, son enviadas por la capa *pooling* a un vector similar.

Si bien esta experimentación pudiese parecer prometedora, no es ideal en ningún caso, ya que la capa *pooling* de SBERT, que adecua el *sentence embedding* para STS, está en otro idioma. Por otro lado, entrenar esta capa para el español es inviable al no existir un *dataset* STS para español.

Cluster N°	Algunos titulares dentro de los *clusters*	Tema en común
54	Manifestantes buscan avanzar a La Moneda en protesta por libertad de presos del estallido social Disturbios terminaron con el incendio de cuatro buses Red en el sector de Las Rejas Nuevo viernes de protestas: manifestantes se enfrentan con Carabineros en el sector de la Alameda con Paseo Ahumada Grupo de personas saquea sucursal bancaria en el centro de Santiago Personas se manifiestan en estación del Metro Irarrázaval exigiendo la libertad de los presos en el estallido social	Manifestaciones
64	Caso Pío Nono: Defensa de carabinero imputado solicitará audiencia de cambio de medidas cautelares Caso Masvida: ausencia de expresidente de holding marca inicio de audiencia de formalización Fiscalía realiza segunda solicitud de prisión preventiva contra uno de los acusados por caso Masvida Caso Masvida: Fiscalía solicita primera prisión preventiva y arresto domiciliario para exgerentes Segundo retiro: TC acoge a trámite y declara admisible el recurso de constitucionalidad de Piñera	Caso, solicitud, prisión preventiva
23	Senador Latorre y el Frente Amplio: "Hay algunos que lo dan por muerto y ojo que podemos dar algunas sorpresas en abril". "Lideré a este club en la quiebra": la carta a ByN con la que Espina explicó su renuncia a Colo Colo "Si fuera cobarde me hubiese ido": Quinteros enfrentó rumores de renuncia a Colo Colo Barticciotto y peticiones de jugadores: "El que exige que le hagan contrato el 2021 es un cara dura" En Huachipato sospechan por suspensión de partidos: "Queda claro que los rumores eran ciertos"	Fútbol, Colo-Colo, Jugadores
66	Meteorología pronostica nublado y chubascos para el día del eclipse en regiones del Biobío y La Araucanía Eclipse Total de Sol: conoce el pronóstico del tiempo y porcentaje de visibilidad para tu región La Nasa transmitirá eclipse solar de este lunes en vivo y en español "Cazador de eclipses" entrega tips para fotografiar con tu celular el próximo Eclipse Total de Sol ALMA transmitirá por streaming el Eclipse Total de Sol del próximo lunes	Clima, Eclipse, Sol, lunes
41	Elecciones del Colmed: Izkia Siches saca ventaja y denuncian llamado de diputado UDI pidiendo votar por su rival RN acuerda llevar a exdiputada Andrea Molina en la papeleta para alcaldía de Viña del Mar Exalcaldesa Karen Rojo no podrá ejercer por 5 años cargos públicos tras ser inhabilitada por el Tribunal Electoral Tras renuncias en RD y salida del Partido Liberal del bloque: diputado Patricio Rosas se va del Movimiento Unir y del Frente Amplio. Agenda Criteria de noviembre: Jadue, Lavín, Jiles y Matthei lideran preferencias presidenciales	elecciones, cargos públicos
16	Presidente Piñera dice que espera que vacunación de Pfizer empiece en Chile "los próximos días o semanas" Reino Unido inicia vacunación masiva: mujer de 90 años es la primera en recibir inoculación contra Covid-19 Zoológico español confirmó contagio de covid-19 de cuatro leones Hamilton da negativo a últimos tests de COVID-19 y estará para competir en Abu Dabi Aubrey Plaza reveló secreto de Kristen Stewart: estuvo contagiada con covid-19 en febrero	Vacuna, Covid19
1	Lo más destacado en El Mostrador: la calculadora de la derecha con los escaños reservados y la fórmula "K" de la UDI Ministro Briones dice que está listo para "apretar el botón" del IFE cuando sea necesario Ministro Delgado ratifica retroceso de la RM a Fase 2 y emplaza a la juventud: "Es el esfuerzo final" Piñera asegura que retroceso de la RM a fase 2 es "preventiva" y que "existe la posibilidad de movernos al paso siguiente para Navidad y Año Nuevo". Presidenta del Colmed por retroceso de la RM a fase 2: "Una medida difícil, pero acertada"	Fase 2, RM
29	Ministro Paris asegura que Chile está preparado para aplicar vacuna de Pfizer contra el Covid-19 Ministro Paris adelanta que vacunas contra el COVID-19 de los laboratorios Pfizer y Sinovac llegarán a Chile en enero Rusia ofrece compartir su vacuna Sputnik V con otras farmacéuticas Canadá da luz verde a vacuna de Pfizer-BioNtech contra covid-19 Hackean documentos de la vacuna Pfizer/BioNTech en Europa	Pfizer, Covid-19, vacuna
32	Informe del DEIS: Chile superó las 21.000 muertes por Covid, entre confirmados y probables, y Concepción sigue como la comuna con más casos activos. Balance COVID-19: Minsal reportó 1.531 casos y 8 fallecidos en las últimas 24 horas Covid-19 en Chile: Minsal reporta 1.389 casos nuevos y 17 fallecidos en última jornada Balance diario del COVID-19: hay 1.247 casos nuevos y 10 fallecidos inscritos en las últimas 24 horas Muertes por covid-19 en España entre marzo y mayo son casi un 70 % más que cifra oficial	casos, fallecidos, COVID-19

Tabla 4.5: Resultados de *clustering* con *embeddings* con BETO y *Agglomerative clustering*

Nils Reimers, uno de los autores de SBERT, afirma[48]: *"Using a BERT model out-of-the-box produces rather bad sentence embeddings. What is required is to have some fine-tuning on some labeled data to get BERT to create good sentence embeddings"*. Añade también *"Also, performance on classification tasks (as shown in the BETO repo) is not a good indicator if the model is suitable to derive good sentence embeddings. In fact, many of the more recent models like Electra, ALBERT etc. produce worse sentence embeddings than BERT"*. Finalmente, concluyendo *"Some large scale labeled data (like paraphrases in Spanish) would be needed to tune BETO so that it creates good sentence embeddings."* en referencia a ajustar (*fine-tuned*) BETO para un *dataset* STS en español, creando una capa *pooling* para el español.

[48](Niels Reimers [Comunicación personal]. 28 de abril de 2021)

Sentence embeddings: SBERT

Se hace uso de la librería *sentence-embeddings*[49] de Python para calcular *sentence embeddings* en inglés. La librería soporta distintos modelos pre-entrenados y ajustados (*fine-tuned*) para STS, que sirven para crear *embeddings* únicos para oraciones completas.

En su sitio web se muestra el rendimiento de distintos modelos, para hasta 14 tareas PLN distintas (Ver Tabla 4.6). Como la prioridad es obtener la mayor calidad posible, independiente del coste computacional (apuntando a una futura escalabilidad), se escoge utilizar el modelo *"all-mpnet-base-v2"*, desarrollado originalmente, por Microsoft.

Model Name	**Performance Sentence Embeddings (14 Datasets)**	**Performance Semantic Search (6 Datasets)**	**Average Performance**	**Speed**
all-mpnet-base-v2	69.57	57.02	63.30	2800
multi-qa-mpnet-base-dot-v1	66.76	57.60	62.18	2800
all-distilRoBERTa-v1	68.73	50.94	59.84	4000
all-MiniLM-L12-v2	68.70	50.82	59.76	7500
multi-qa-distilbert-cos-v1	65.98	52.83	59.41	4000
all-MiniLM-L6-v2	68.06	49.54	58.80	14200
multi-qa-MiniLM-L6-cos-v1	64.33	51.83	58.08	14200
paraphrase-multilingual-mpn ...	65.83	41.68	53.75	2500
paraphrase-albert-small-v2	64.46	40.04	52.25	5000
paraphrase-multilingual-Min ...	64.25	39.19	51.72	7500
paraphrase-MiniLM-L3-v2	62.29	39.19	50.74	19000
distiluse-base-multilingual ...	61.30	29.87	45.59	4000
distiluse-base-multilingual ...	60.18	27.35	43.77	4000

Tabla 4.6: Modelos pre-entrenados disponibles en librería *sentence-embeddings*

La transformación concreta de texto a *embeddings* es bastante simple: se crea un objeto *SentenceTransformer*, al cual se le especifica el nombre del modelo, y si se quiere operar en GPU o CPU. Luego, se usa el método *encode* al cual se le provee el texto plano. Finalmente, el *output* es el *embedding* (vector numérico) deseado.

Este procedimiento se encapsula en una clase denominada *EmbeddingModels*. Esta clase está pensada para utilizar diversos modelos presentes de la librería, pero sólo limitándose a ella. Los métodos *compute_embeddings* y *compute_embeddings_batch*, iteran sobre todos los modelos, generando *embeddings* que calzan con los *mappings* de los artículos, especificando el texto y el lenguaje del texto, para utilizar sólo modelos adecuados para ese lenguaje.

Esta clase se usa dentro de *TextProcessing*, que además de realizar los cálculos PLN, calcula las *embeddings* también. Recordar que esta clase se utiliza en el *Preprocess service*. Antes de calcular los *embeddings*, se efectúa la traducción del titular mediante la clase *TranslationESEN*.

[49]https://www.sbert.net

Si bien la clase *EmbeddingModels* soporta varios modelos de la librería, en la práctica sólo se utiliza el modelo *"all-mpnet-base-v2"*, porque es el modelo que brinda *embeddings* mayor calidad (Ver Tabla 4.6). Además, se ahorra tiempo, evitando generar *embeddings* subóptimos.

El problema con esta clase es su rigidez, porque se restringe a utilizar modelos de la librería *sentence-embeddings*. Para soportar métodos alternativos, se tendría que realizar un *refactor* de esta clase, y llegar a un resultado similar al sistema de clases descrito en la Sección 4.4.1.

Clustering no supervisado

Mediante un conjunto de *embeddings* de SBERT (que provienen de titulares de noticias traducidos al inglés), se pretende correr varias veces un algoritmo de *clustering* y escoger el resultado que tenga mejor *Silhouette* o *Calinski score*.

Se extiende el código del *pull request* #6948 abierto de la librería *scikit-learn*[50], mencionado en la Sección 2.3.3. Se modifica la clase *OptimalNClusterSearch* para soportar, además de *KMeans*, los siguientes algoritmos de *clustering* presentes en *scikit-learn*: *Affinity Propagration*[51], *Agglomerative clustering*[52], *DBSCAN*[53], *OPTICS*[54] y *Spectral clustering*[55].

Cada algoritmo tiene un parámetro de búsqueda, y también parámetros fijos. Estas configuraciones se muestran en la Tabla 4.7.

La decisión de los parámetros fijos se efectúa basándose en una investigación hecha para cada algoritmo. Como se observa en la tabla, los posibles valores del parámetro $n\_clusters$ equivale al número de ejemplos. Sin embargo, este rango puede acotarse mediante dos criterios: (1) establecer un número mínimo de *clusters*: $min\_n\_clusters$, y (2) establecer un número de *outliers* o ruido: $n\_outliers$. De esta forma, el nuevo rango para el parámetro $n\_clusters$ será $[n\_outliers + n\_clusters + 1, n - n\_clusters]$, donde n es el número de *embeddings*. Notar que existe la restricción $2 * min\_n\_clusters \geq (n - n\_outliers)$. En la práctica el número de *outliers* se expresará de manera porcentual entre 0 y 1 con respecto a n, mientras que el mínimo número de *clusters* como un número entero.

Por su parte, los rangos de interés para los parámetros *damping*, *eps* y *xi* de los algoritmos *Affinity propagration*, *DBSCAN* y *OPTICS*, deben ser buscados en forma experimental. Por un lado, *damping* tiene un rango bien definido, y en efecto, sólo se debe buscar con mayor fineza un rango adecuado. Por otro lado, para *eps* y *xi* se establece como límite superior la máxima distancia entre dos puntos (*embeddings*), para las métricas euclidianas y Minkowski respectivamente, porque valores superiores a estos no hacen sentido. Como límite inferior se utiliza el valor $1 * 10^{-6}$. La idea es probar al menos cien valores distintos en cada caso, escogidos en manera equidistante en ese intervalo.

[50] https://github.com/scikit-learn/scikit-learn/pull/6948
[51] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html
[52] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html
[53] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
[54] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.OPTICS.html
[55] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html

Algoritmo	**Parámetro de búsqueda**	**Rango de parámetro de búsqueda**	**Parámetros fijos**
KMeans	n_clusters	[1, n_samples]	random_state: 33 max_iter: 400 tol: 1e-4 init: k-means++ n_init: 30 algorithm: elkan
AffinityPropagration	damping	[0.5, 1.0]	random_state: 33 max_iter: 500 convergence_iter: 30
AgglomerativeClustering	n_clusters	[1, n_samples]	linkage: ward affinity: euclidean
DBSCAN	eps	]0, infinte[	metric: euclidean min_samples: 2 algorithm: ball_tree leaf_size: 10 p: 2
OPTICS	xi	]0, infinite[	metric: minkowski min_samples: 2 algorithm: ball_tree leaf_size: 10 p: 2 cluster_method: xi
SpectralClustering	n_clusters	[1, n_samples]	random_state: 33 n_init: 100 gamma: 1.0

Tabla 4.7: Configuración de algoritmos de *clustering* y parámetros de búsqueda

Además de integrar más algoritmos de *clustering*, esta modificación contempla la creación de la clase *EmbeddingCLustering*, que encapsula el uso de la clase *OptimalNClusterSearch* modificada, para soportar los valores de $min\_n\_clusters$, $n\_outliers$ y también para soportar configuraciones personalizadas para cada algoritmo, es decir, para cambiar los parámetros fijos definidos anteriormente. Esta clase implementa un método llamado *fit*, que recibe los puntos a agrupar y retorna las *labels* de cada *embedding*, después de ejecutar el *clustering* no supervisado, con el algoritmo, parámetros, rango y criterios establecidos.

Experimentación y elección de *pipeline*

Se realizan tres experimentos para decidir qué procedimiento o *pipeline* efectuar para agrupar las *sentence embeddings*, debido a que en este punto existen muchas posibilidades.

El primer experimento, consiste en evaluar preliminarmente los criterios de selección no supervisada de resultados de *clustering*, ante diversos *datasets* sintéticos de *embeddings* de dos

dimensiones. El objetivo es observar el comportamiento de estos criterios ante *datasets* simples y acotados, antes de utilizarlos en *embeddings* de 128 dimensiones. Para ello, sólo se hacen 6 agrupaciones por cada *pipeline*, a excepción de *DBSCAN*, *OPTICS* y *Affinity propagation* que buscan el mejor resultado entre 100 combinaciones.

El segundo experimento, busca medir los tiempos de ejecución de las distintas combinaciones de algoritmos y criterios de selección, ante un *dataset* de *embeddings* de dos dimensiones, pero incrementando el número de agrupaciones a 30 en cada combinación. La idea es ver qué tan factible es el uso de ciertas combinaciones ante un caso de uso más cercano a la realidad, ya que nuevamente, se están usando *embeddings* de tan sólo dos dimensiones.

El último experimento, busca evaluar la utilización de estas *pipelines* ante un caso de uso real, usando una muestra de 300 artículos obtenidos entre las fechas 20 y 23 de diciembre de 2021. La idea es medir los tiempos de ejecución y evaluar cualitativamente la calidad de los resultados, para identificar las mejores *pipelines*.

Primer experimento

Se usan cinco *datasets* sintéticos distintos para el primer experimento, en el código extendido. La librería *scikit-learn* permite generar *datasets* de este estilo especificando el número de puntos (*embeddings*), el número de *clusters*, la forma de los *clusters* (esféricos, anillos circulares o en forma de luna) y el ruido presente. En este caso, se usa un *dataset* con 3 centroides esféricos, otro con cinco, un *dataset* con dos anillos circulares, otro con dos *clusters* en forma de luna, y uno generado con puntos en posiciones aleatorias. Se prueban seis algoritmos y cinco criterios de selección, sobre los cinco *datasets* descritos.

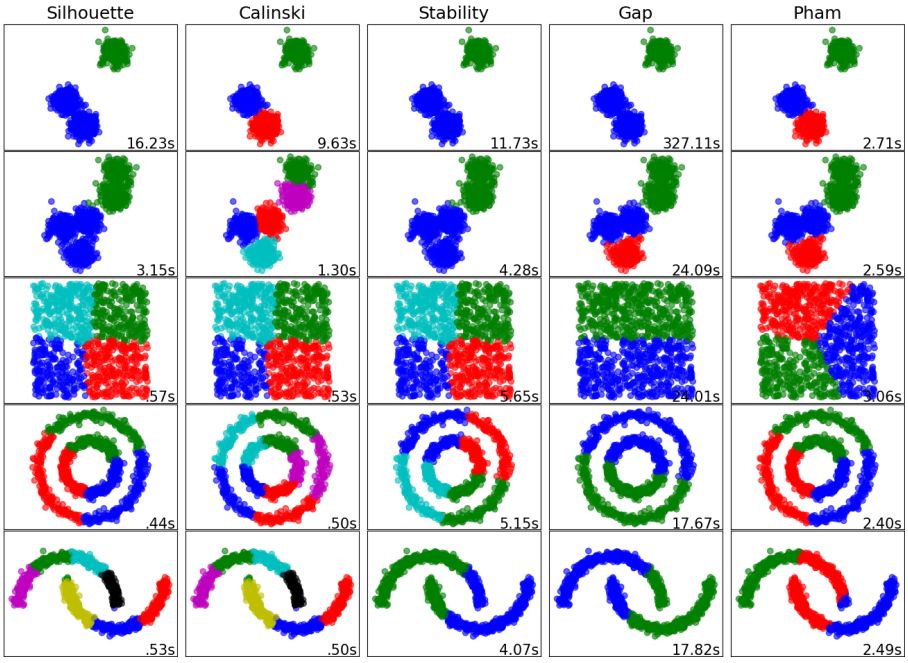

Figura 4.5: Resultados de experimento #1 para *KMeans*

La Figura 4.5 evidencia cómo el algoritmo *KMeans* no puede en ningún caso identificar los *clusters* de anillos circulares y de luna correctamente. Esto se debe a la naturaleza del algoritmo que funciona mejor para *clusters* esféricos. Es más, es evidente que el *dataset* de tres y cinco centroides los identifica correctamente, mediante el criterio de *Calinski*. Sin embargo, no identifica bien estos *clusters* con los demás métodos, sólo el criterio *gap* y *pham* se le acercan. Por último el *dataset* aleatorio es interesante porque en los casos de *Silhouette*, *Calinski* y *stability* genera *clusters* con centroides casi equidistantes.

Por otro lado, la Figura 4.6 muestra los resultados del algoritmo *Affinity propagation* que corresponde al mejor resultado entre 100 agrupaciones cada vez, variando el parámetro *damping*. Debido a que el espacio de búsqueda es mayor, se pueden observar que en todos los casos, los *clusters* son cúmulos bastante densos, cercanos y pequeños. Esto se traduce en una alta especificidad de los *clusters* encontrados. Con este ejemplo no es claro que un criterio de selección sea mejor que otro. Notar los grandes tiempos de ejecución.

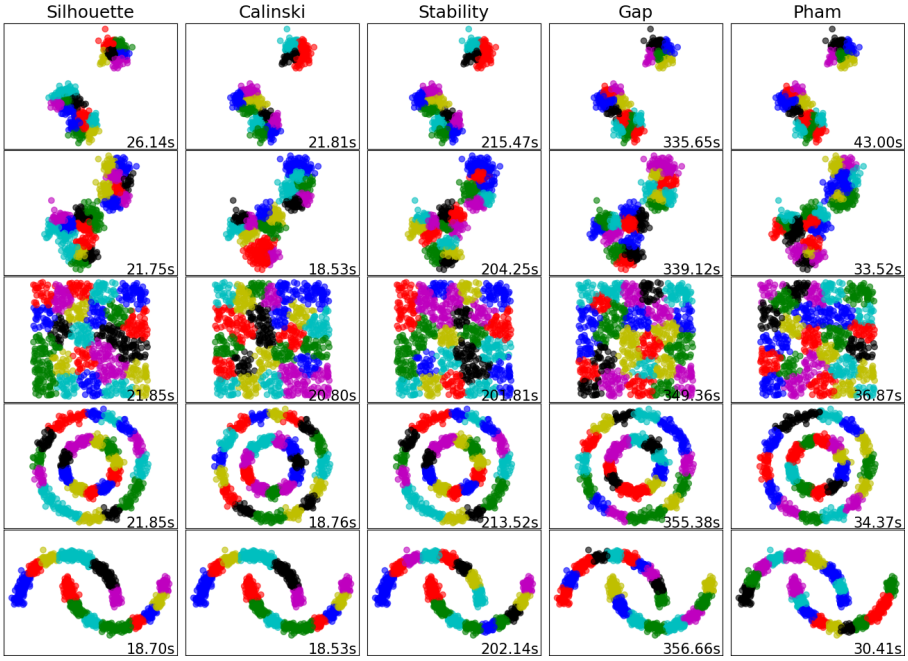

Figura 4.6: Resultados de experimento #1 para *Affinity propagation*

El resultado de *Agglomerative clustering* se muestra en la Figura 4.7. Aquí destaca el criterio *Calinski* ya que logra identificar muy bien los *clusters* de los *datasets* de tres y cinco centroides. También destaca el criterio *pham* en los mismos casos. Nuevamente, los *datasets* de anillos circulares y de luna no son identificados correctamente. Notar que los tiempos de ejecución son bastante menores a *KMeans*, donde ambos tienen el mismo espacio de búsqueda.

Por su parte, los resultados de *DBSCAN* expuestos en la Figura 4.8 muestran la gran cualidad de este algoritmo basado en densidad, ya que identifica correctamente con los criterios *Silhouette*, *Calinski*, *stability* y *gap* los *clusters* de anillos circulares y de luna. Sin embargo, el *dataset* de tres centroides es sólo identificado con los criterios *gap* y *pham*, mientras que el *dataset* de cinco centroides no es identificado correctamente en ningún caso, de hecho, en el caso del criterio *gap*, la mayoría de los puntos son detectados como ruido debido a la sensibilidad

del parámetro *eps*[56]. Notar además, que en todos los casos el *dataset* de puntos aleatorios es identificado como un solo *cluster*.

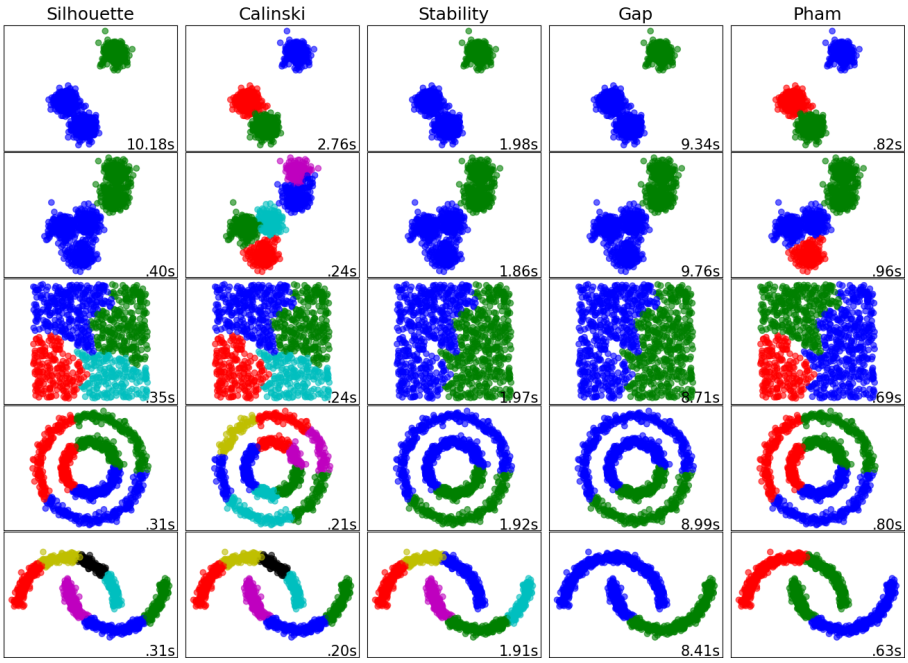

Figura 4.7: Resultados de experimento #1 para *Agglomerative clustering*

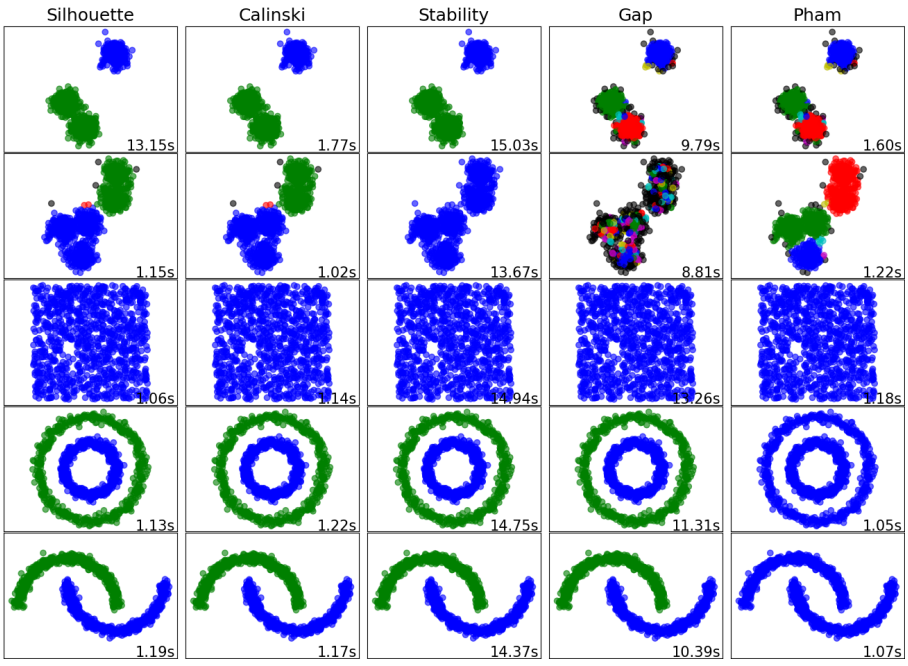

Figura 4.8: Resultados de experimento #1 para *DBSCAN*

[56]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

OPTICS, algoritmo también basado en densidad, detecta en todos los casos muchos puntos (*embeddings*) como ruido, y por esto la mayoría de los puntos son pintados en negro en la Figura 4.9. Esto quiere decir que solamente junta zonas densas y que el resto lo retorna como ruido. Esto sucede porque el parámetro *xi* es bastante sensible y generalmente se ajusta al *input* en forma experimental. Que este parámetro sea seleccionado con los criterios de elección no supervisada, en ningún caso garantiza resultados de calidad y este es un ejemplo de aquello.

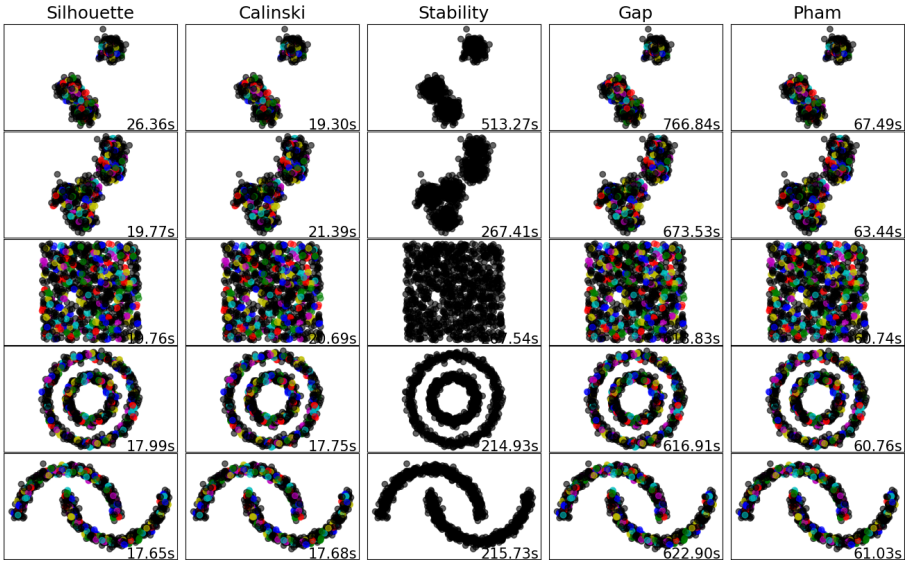

Figura 4.9: Resultados de experimento #1 para *OPTICS*

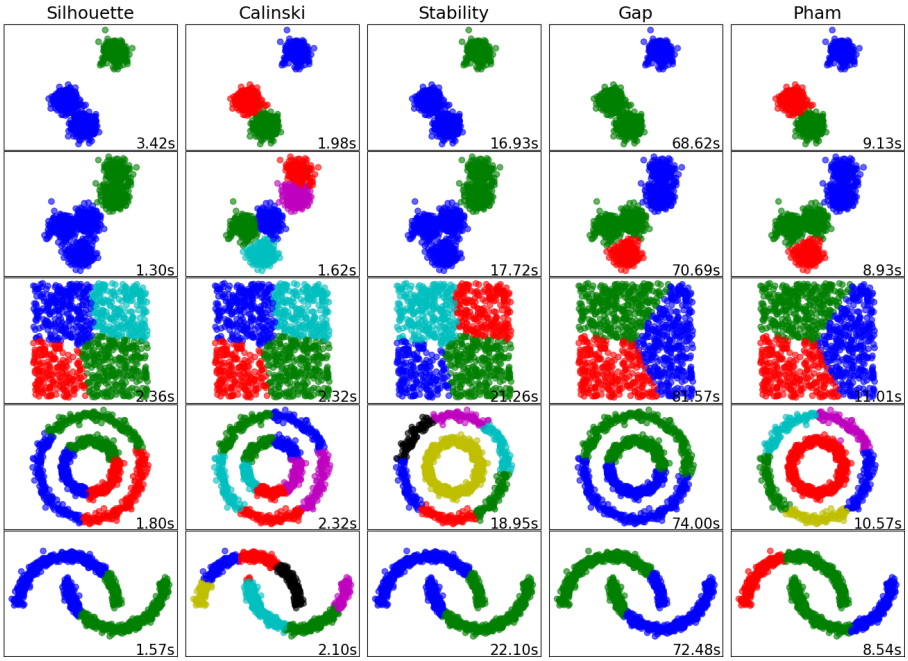

Figura 4.10: Resultados de experimento #1 para *Spectral clustering*

La Figura 4.10 muestra los resultados de *Spectral clustering*, que son bastante similares a los de *KMeans*, pero con tiempos de ejecución mucho menor. Destacan el criterio *Calinski* para el *dataset* con tres y cinco centroides, y el criterio *pham* para el *dataset* con tres centroides.

De estos resultados, es evidente apreciar que el tiempo de ejecución de los criterios de selección *stability*, *gap* y *pham* son mucho más altos que los criterios *Silhouette* y *Calinski*, independiente del algoritmo utilizado. En cuento a la calidad de los resultados, era esperable que *DBSCAN* acertara con los *clusters* de anillos circulares y de luna, mientras que *KMeans* con los *datasets* esféricos. Lo destacable de estos casos, es que la elección de los mejores resultados se obtiene en forma no supervisada.

El criterio *gap* sólo fue útil para *DBSCAN* en algunos casos, pero no para los cinco algoritmos restantes. El criterio *stability* no identificó en ningún caso los *clusters* correctos del *dataset* de tres y cinco centroides. Los criterios *pham*, *Silhouette* y *Calinski* acertaron en casos puntuales, siendo *pham* el más demoroso.

Los algoritmos *OPTICS* y *DBSCAN* basados en densidad demuestran la sensibilidad de sus parámetros *xi* y *eps* respectivamente. Los puntos son incluidos en un sólo *cluster*, o bien, todos los puntos son detectados como ruido. Esto hace pensar que es difícil obtener buenos resultados con estos algoritmos para *embeddings* de 128 dimensiones, por el problema de la métrica euclidiana, y también porque se estima que la distribución espacial de los *embeddings* sea similar al caso del *dataset* de puntos aleatorios. Ambos algoritmos caen en este comportamiento no deseado.

Por otro lado, *Affinity propagation* logra encontrar *clusters* densos y específicos, por lo que es probable que al aplicar este algoritmo a los titulares, se encuentren *clusters* de titulares parafraseados, bien agrupados, pero perdiendo la generalidad de un tópico u evento. En este caso, se debe encontrar una forma de juntar estos *clusters* específicos nuevamente.

De esta forma, los algoritmos *Agglomerative clustering*, *Spectral clustering* y *KMeans*, junto a los criterios *Silhouette* y *Calinski* demuestran ser los más prometedores al momento de realizar un *clustering* no supervisado.

Segundo experimento

El objetivo en este caso es medir los tiempos de ejecución de cada procedimiento, porque de ser demorosas para puntos con dos dimensiones, serán mucho más demorosas para puntos con 128 dimensiones, que es el caso de las *sentence embeddings*. Se utiliza un *dataset* más pequeño de 200 puntos de 2 dimensiones, con 40 centroides de forma esférica, y un 5 % de ruido.

El rango a buscar para el parámetro $n\_clusters$ será de [26, 185], dado que se usa un $min\_n\_clusters = 15$ y $n\_outliers = 0{,}05$ intentando que los métodos no supervisados encuentren el mejor resultado de *clustering* cercano a 40, pero al mismo tiempo, estando en un escenario donde hay que buscar entre muchos resultados posibles (160 en este caso). Los algoritmos *Affinity propagation*, *OPTICS* y *DBSCAN* se configuran para probar 160 agrupaciones distintas.

En la Tabla 4.8 se muestran los resultados de este experimento para todas las *pipelines* disponibles. En primer lugar, es evidente apreciar que los criterios de elección no supervisada

	Silhouette	Calinski	pham	stability	gap
Agglomerative clustering	2.59	**0.21**	0.4	0.65	3.39
KMeans	**25.81**	29.04	101.47	317.66	1113.5
OPTICS	190.96	**86.06**	383.8	1501.9	4127.23
Spectral clustering	**153.03**	153.34	734.70	1807.53	4961.2
Affinity propagation	19.56	**7.26**	30.58	83.13	307.29
DBSCAN	16.65	**5.72**	6.56	18.9	47.32

Tabla 4.8: Tiempos de ejecución en segundos de *clustering* no supervisado para distintos algoritmos y criterios, para 200 datos sintéticos de 2 dimensiones, y 179 posibles resultados.

gap y *stability* son altamente demorosos incluso para un *dataset* pequeño como este, por lo tanto, su uso práctico es descartado. Además, como se muestra en el experimento anterior, *gap* no fue útil nunca en términos de calidad, y *stability* sólo en casos puntuales.

Por otro lado, el criterio *pham* también es demoroso, y además está diseñado para escoger un número de *clusters* entre 0 y 10 [76], por lo tanto, no sirve para el rango de búsqueda que se pretende usar, de al menos 40 *clusters*.

De esta forma, los criterios *Silhouette* y *Calinski* se muestran como alternativas prácticas y eficientes para seleccionar el mejor resultado de *clustering* cada vez, en términos de rendimiento y calidad, debido a que funcionan bien tanto para algoritmos basados en densidad, como para algoritmos que buscan *clusters* esféricos.

Al mismo tiempo, es posible observar que los algoritmos *KMeans*, *OPTICS* y *Spectral clustering* son mucho más demorosos (en órdenes de magnitud) que *Agglomerative clustering*, *DBSCAN* y *Affinity propagation*, y por lo tanto, estos últimos se presentan como alternativas bastante eficientes.

Sin embargo, como *DBSCAN* es sensible breves variaciones de su parámetro *eps*, no se considera una opción práctica, a pesar de que su tiempo de ejecución es relativamente bajo.

Finalmente, *Agglomerative clustering*, y *Affinity propagation*, junto a los criterios *Silhouette* y *Calinski*, se instalan como las *pipelines* más prometedoras. Sin descartar obviamente el uso de los otros algoritmos por ahora.

Tercer experimento

Se realiza un experimento seleccionando 300 artículos entre el 20 y 23 de diciembre de 2021. Se utiliza un *min_num_clusters* de 20, mientras que un porcentaje de *outliers* del 25 %, lo cual da un rango de *n_clusters* de [71, 180], es decir, 110 resultados posibles. Los algoritmos *DBSCAN*, *OPTICS* y *Affinity propagation* se configuran para probar 110 resultados también.

Se prueban todos los algoritmos, pero sólo con los criterios *Silhouette* y *Calinski*. Como la dimensionalidad de los *embeddings* es 128, no es posible graficar los resultados de *clustering*. En cambio, se aplicará una reducción de dimensionalidad mediante el algoritmo PCA[57], que selecciona las dimensiones con mayor varianza, para tener una visualización referencial.

[57]https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

Analizando los resultadps, tanto *Affinity propagration* como *DBSCAN*, no convergen a un resultado coherente: se forma un sólo y gran *cluster*, o bien, se todos los *embeddings* con catalogados como *outliers*. Esto sucede debido a la sensibilidad de parámetros. Para corroborar esta hipótesis, se ejecuta otra búsqueda con mayor fineza, esta vez probando entre 300 posibles resultados de *clustering*, pero se observa el mismo comportamiento. En la Figura 4.11 se aprecia cómo todos los puntos son identificados como ruido para el caso de *DBSCAN*.

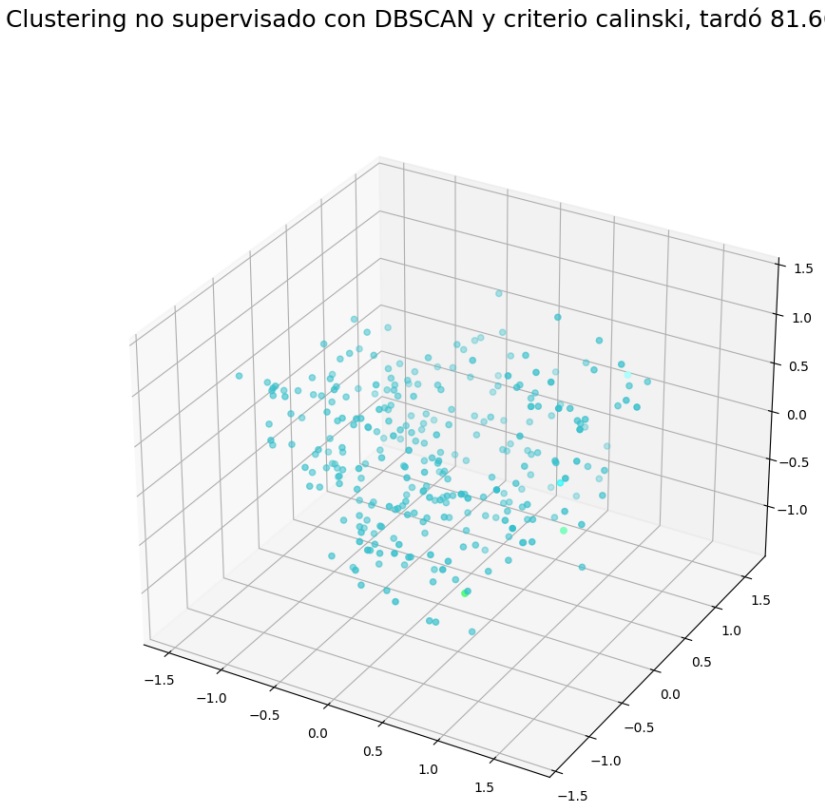

Figura 4.11: Visualización de *embeddings* reducidos a 3 dimensiones para experimento #3 y algoritmo *DBSCAN*, criterio *Silhouette*.

Y no tan sólo esto, para *DBSCAN* se intenta además modificar alguno de los parámetros fijos definidos anteriormente como: *min_samples, metric, algorithm, leaf_size* y *p*, pero aún así no se encuentran zonas para *eps* que retornara resultados coherentes. Por este motivo, se descarta el uso de *DBSCAN*. Por otro lado *Affinity propagation* no tiene más parámetros que se puedan modificar, y por lo tanto también se descarta. Se tiene la hipótesis de que, además, la alta dimensionalidad de los *embeddings* está afectando el comportamiento de estos algoritmos.

Un comportamiento levemente similar se encuentra con los algoritmos *Spectral clustering* y *KMeans*, para ambos criterios de selección (como pasa también en el primer experimento). La mayoría de los titulares son *clusters* en sí mismos, mientras que muy pocos son agrupados en *clusters* de tamaño 4 a 8. En la Figura 4.12 se aprecia cómo *KMeans* asigna prácticamente una etiqueta (color) distinto a cada *embedding*.

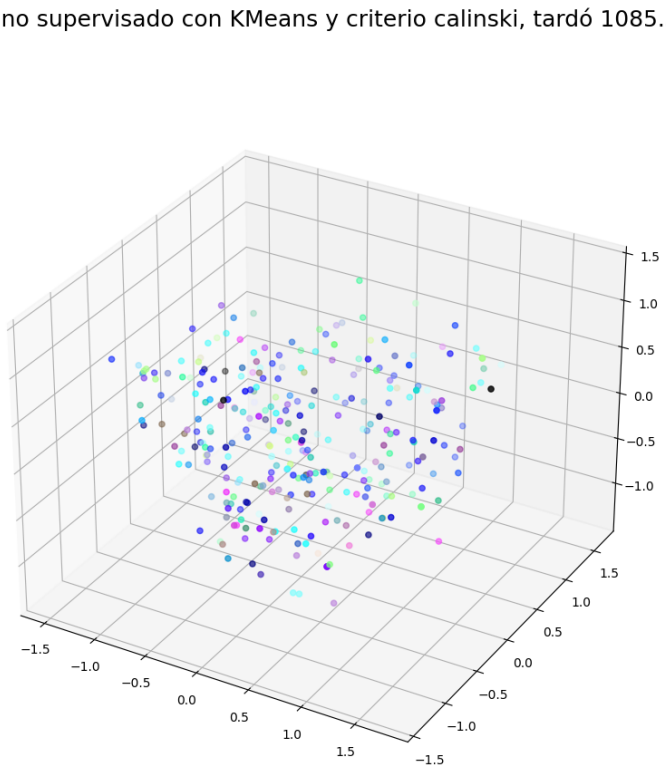

Figura 4.12: Visualización de *embeddings* reducidos a 3 dimensiones para experimento #3 y algoritmo *KMeans*, criterio *Calinski*.

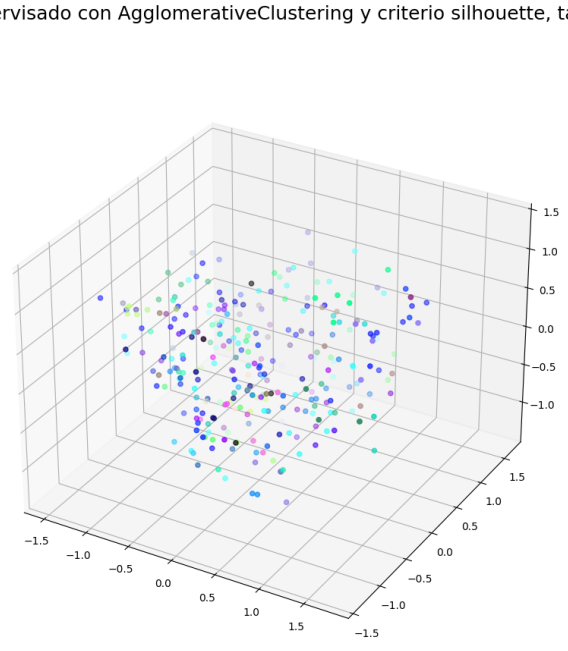

Figura 4.13: Visualización de *embeddings* reducidos a 3 dimensiones para experimento #3 y algoritmo *Agglomerative clustering*, criterio *Silhouette*.

En contraste a esto, los resultados de *Agglomerative clustering* y *OPTICS* son preliminarmente bastante buenos. Con el primer algoritmo y el criterio *Calinski* se obtienen malos resultados como en los casos anteriores. Pero si se usa el criterio *Silhouette*, se obtienen muy buenos resultados, ya que el algoritmo encuentra varios *clusters* de titulares de un mismo tópico. *OPTICS* por su parte, identifica a muchos titulares como ruido, pero los *clusters* que genera son de buena calidad.

En la Figura 4.13 se observan las distintas etiquetas de los *embeddings* mediante el algoritmo *Agglomerative clustering* y criterio *Silhouette*. Por otro lado, en la Figura 4.14 se observa cómo el algoritmo *OPTICS*, junto al criterio *Calinski*, etiqueta a muchos *embeddings* como ruido (color celeste), pero al mismo tiempo genera bastantes *clusters* también.

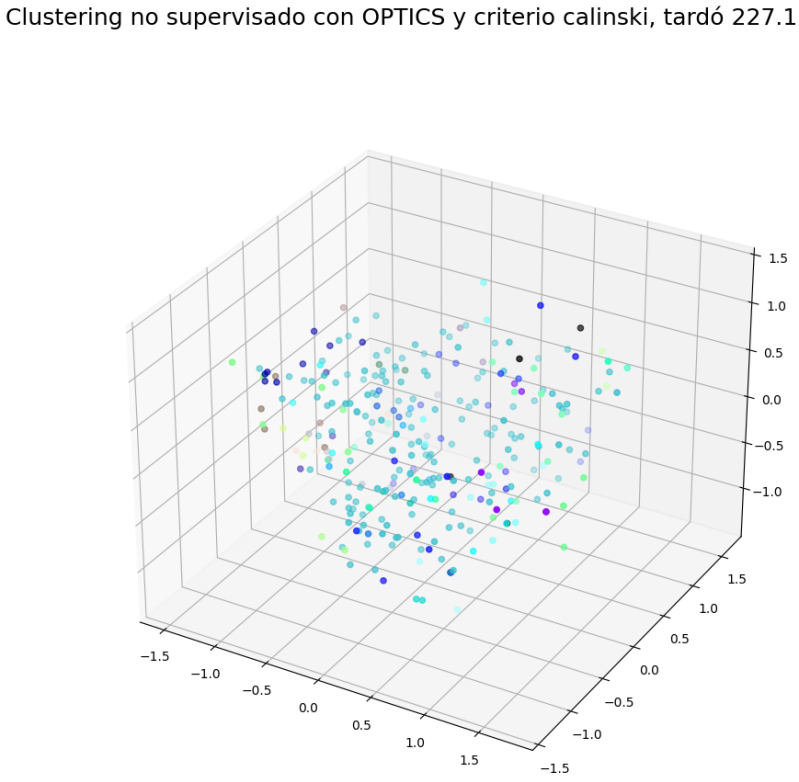

Figura 4.14: Visualización de *embeddings* reducidos a 3 dimensiones para experimento #3 y algoritmo *OPTICS*, criterio *Calinski*.

Es importante mencionar que *OPTICS* puede funcionar con base en *DBSCAN*, pero también en base como a un algoritmo alternativo denominado *xi* propuesto la publicación original e implementado por *scikit-learn*. Se hacen pruebas de *OPTICS* funcionando a partir de *DBSCAN*, y se obtiene el mismo comportamiento obtenido con *DBSCAN* (a secas), por lo tanto, se escoge utilizar el algoritmo *xi* que brinda mejores resultados.

#	**Clusters**
32	Confirman 25 contagios nuevos de Covid-19 en Ñuble. Rapa Nui registra 9 casos nuevos de COVID-19 tras llegada de un vuelo desde el continente. Confirman nueve casos de Covid-19 en Rapa Nui tras 15 meses libres de contagios. Seremi de Salud confirma nueve casos de Covid-19 en Rapa Nui tras casi 15 meses sin contagios.
10	Guilherme Boulos: un ícono de la renovación en el progresismo brasileño. Diputado Evópoli Francisco Undurraga: "Apoyar a Kast fue una derrota a nuestro propio proyecto". El triunfo de la campaña contra José Antonio Kast. CRISIS TOTAL EN LA DERECHA: Fachos se están apuñalando todos con todos tras el fracaso de Kast. Senador Ossandón afirma que J.A. Kast "no va a ser líder de la oposición" y que el Presidente- Piñera "debería retirarse de la política".
27	Seremi de Salud y Municipalidad de Rancagua hacen llamado al autocuidado durante Fiestas de Fin de Año. Comité Santa María apoya Navidad en Los Lirios y Requínoa. Alta congestión vehicular y de transeúntes en los días previos a Navidad. Trineo móvil navideño recorre diferentes sectores de Las Cabras. Hospital Regional de Rancagua vive la navidad.
3	¿Qué esperan los deportistas de Ñuble del nuevo Presidente de Chile?. Xi Jinping felicita a Boric y dice que buscará elevar la asociación estratégica integral entre China y Chile. Justin Trudeau felicita a Gabriel Boric y lo anima a fortalecer relaciones con Canadá. Presidentes de Chile. Cita en La Moneda: Presidente electo Gabriel Boric llega al Palacio para reunirse con Mandatario- Sebastián Piñera. UE felicita a Gabriel Boric por su triunfo en el balotaje y esperan "fortalecer" su relación con Chile. Gabriel Boric nuevo Presidente de Chile.Ídolo de la UC felicitó a Boric tras ser elegido como nuevo- Presidente de Chile.
24	Se viven horas claves: ANFP fijó plazo para defensa de Melipilla tras denuncia de dobles contratos. Marcianeke entrega su versión por incidente en resort: "Ellos no respetaron el horario que pagamos". Caso Melipilla suma antecedentes graves: Amenazas, extorsión, fotos de familiares y coimas. Melipilla acumula nuevos problemas: Directorio de la ANFP se suma a denuncia en su contra.
39	Tras derrame en Quintero: ¿Qué sucede con los animales al sufrir derrames de petróleo en su hábitat?. Qué hacer si encontramos aves o animales silvestres extraviados o heridos. "Nos están matando el verano": Pescadores y autoridades reaccionan ante derrame de petróleo en Quintero. Nuevo derrame de petróleo ocurrió en las cercanías del Puerto Ventanas en Quintero.
20	Contraloría denunció millonarias irregularidades en la gestión del ex alcalde Alessandri en Santiago. Ex alcalde de Chile Vamos es condenado a 10 años de presidio por malversación de caudales- públicos: perjuicio fiscal fue de $351 millones. Contraloría envía a Fiscalía antecedentes sobre millonarios pagos por horas extras que no habrían- sido trabajadas durante alcaldía de Alessandri. Horas extra: Contraloría confirma millonarias irregularidades en administración Alessandri.

Tabla 4.9: Siete *clusters* ejemplares de *OPTICS* junto a criterio *Calinski*.

En cuanto a la calidad de resultados, en la Tabla 4.9 se encuentran siete *clusters* de titulares obtenidos con el algoritmo *OPTICS* y criterio *Calinski*. Si bien existen agrupaciones que no son correctas como el *cluster* #24, en general, se observa que la calidad de los resultados es bastante buena. Aclarar que esta tabla sólo es una muestra entre los 46 *clusters* formados.

Por otro lado, la Tabla 4.10 expone también una muestra de resultados de *clustering* pero para el algoritmo *Agglomerative clustering* y criterio *Silhouette*, donde se encuentran 146 *clusters* distintos. Notar que la calidad de los *clusters* también es buena. Sin embargo, hay *clusters* que no están en la tabla, que son agrupaciones erróneas o a veces, de los cuatro titulares que componen un *cluster*, dos hablan de un tema y los otros dos son ruido. Esto evidencia que el algoritmo tampoco es perfecto.

#	Clusters
10	Boric tras reunirse con Presidente Piñera adelanta gabinete paritario y reitera reformas económicas "paso a paso". SNA pide que gobierno de Boric "incluya la agricultura, sus tradiciones y a las regiones". Boric asegura un "traspaso de mando ordenado" con pdte. Piñera y adelanta detalles de su gabinete. Tras quema de viviendas en Lago Lanalhue, gobernador Díaz pide a Boric adelantar designación de autoridad para abordar conflicto en la zona sur.
49	Jiles proyecta cómo sería el gabinete de Boric: "Hay un movimiento de presentación de currículums impresionante". Jackson y definición del próximo gabinete: "Todavía no estamos con nombres sobre la mesa". Jackson cita a Arjona en referencia a gabinete de Boric: "Una amalgama perfecta entre experiencia y juventud".
61	En San Fernando Carabineros detuvo a joven acusado de dejar a dos heridos graves. Lo detuvieron por receptación y agresión a un carabinero. Detienen a hombre por presunto parricidio frustrado en Chillán: habría agredido a su bebé de 2 meses.
25	Comisión de Hacienda de la Cámara vota hoy la Pensión Garantizada Universal. Con Pensión Garantizada Universal como indicación: Gobierno pone discusión inmediata a ley corta de pensiones. Pensión Garantizada Universal: Gobierno espera que se apruebe "lo antes posible" y se convierta en ley durante enero. La propuesta de pensiones de Boric: Mantener la propiedad de los ahorros y la creación de un ente público y autónomo. Comisión de Hacienda de la Cámara Baja aprueba la Pensión Garantizada Universal.
55	Covid: Estados Unidos aprueba la primera pastilla de uso doméstico contra la enfermedad para personas de alto riesgo. Israel comenzará a aplicar cuarta dosis de Covid-19. EE.UU. autoriza uso de emergencia de la pastilla de Pfizer contra el COVID-19. Autorizan píldora de Pfizer como tratamiento contra el Covid-19 en Estados Unidos.
129	Tras derrame en Quintero: ¿Qué sucede con los animales al sufrir derrames de petróleo en su hábitat?. "Nos están matando el verano": Pescadores y autoridades reaccionan ante derrame de petróleo en Quintero. Nuevo derrame de petróleo ocurrió en las cercanías del Puerto Ventanas en Quintero.
5	Avión recibió un disparo mientras apagaba un incendio en Contulmo. Ataque incendiario en el sector del Lago Lanalhue terminó con seis inmuebles quemados. Al menos 27 viviendas destruidas en ataque incendiario en Contulmo.

Tabla 4.10: Siete *clusters* ejemplares de *Agglomerative clustering* junto a criterio *Silhouette*.

Por otro lado, comparando los tiempos de ejecución de cada algoritmo, expuestos en la Tabla 4.11, es posible evidenciar la misma tendencia que en el caso anterior: los algoritmos *KMeans*, *Spectral clustering* y *OPTICS* son los más demorosos, mientras que *Agglomerative clustering* es el algoritmo que demora menos. Notar que por ejemplo, *KMeans* bajo el criterio *Calsinki* demora un 3736 % más, en comparación con los datos sintéticos del experimento anterior. Si la tendencia fuera lineal, utilizar el criterio *gap* en este caso demoraría 11.5 horas, dejando en evidencia la inviabilidad de los demás criterios.

	Silhouette	Calinski
Agglomerative clustering	21.18	**7.46**
KMeans	1176.63	**1085.36**
OPTICS	322.63	**227.13**
Spectral clustering	1051.27	**924.55**
Affinity propagation	130	**129.91**
DBSCAN	**69.87**	81.66

Tabla 4.11: Tiempos de ejecución en segundos de *clustering* no supervisado para distintos algoritmos y criterios, para 300 artículos de noticias.

Elección de *pipeline*

Ponderando la calidad preliminar de los *clusters* y los tiempos de ejecución, se determina que utilizar *OPTICS* o *Agglomerative clustering*, junto a los criterios *Silhouette* o *Calsinki* es la mejor opción. Incluso se puede decir que *a priori* los resultados mediante el criterio *Silhouette* son mejores, sólo por las observaciones hechas en los experimentos anteriores.

Con estos resultados prometedores, se puede concluir preliminarmente que los procesos de traducción, cálculo de *sentence embeddings*, *clustering*, y criterios de selección no supervisada, son suficientes para afrontar el problema de agrupación de titulares. Además, cada parte del proceso tiene parámetros modificables y, por lo tanto, abre las puertas a nuevas formas de agrupación, que perfeccionen los resultados. Se exponen en la Sección 5.1.2, diversos casos de uso donde se analiza la calidad de estos resultados agrupando más titulares.

4.5.2. Método con heurística

Para este método, se toma de base el código desarrollado por los autores de la publicación expuesta en la Sección 3.4.3, disponible en Github[58]. Se reutilizan dos componentes fundamentales: la clase *Tokenizer* y la función *detect_keywords* que implementa el Algoritmo 1. Lo primero permite ejecutar el proceso de limpieza del *input*, mientras que lo segundo ejecuta el algoritmo de detección de eventos.

El *Tokenizer* transforma cada palabra de un titular en *tokens*. Cada *token* se compone de una o más palabras, y tiene una etiqueta que lo identifica como verbo, sustantivo (entidad), pronombre, etc. Luego, dependiendo de la etiqueta de cada palabra se hace un proceso de filtro, donde ciertos *tokens* se quitan del titular inicial. Por ejemplo, las *stopwords* (el, la, lo , y, etc.) son eliminadas, así como los espacios en blanco y las puntuaciones. También las palabras son estandarizadas a minúscula, etc.

Para esto la clase *Tokenizer* se apoya en la librería spacy[59], que mediante un vocabulario definido, desempeña el proceso de etiqueta y reconocimiento de entidades (tarea PLN conocida como *token classification*). En este caso, se utiliza el vocabulario para español *es_core_news_sm*[60].

La principal ventaja del proceso de etiqueta e identificación entidades, aparte de eliminar y estandarizar palabras, es poder detectar aquellas entidades que constan de más de una palabra, como por ejemplo, "Alexis Sánchez" o "Gabriel Boric". De modo que un titular del estilo "Alexis Sánchez celebra el triunfo de su equipo", tendrá los *tokens* "Alexis Sánchez", "celebra", "triunfo" y "equipo". Al mismo tiempo, esto presenta una desventaja para el Algoritmo 2, de asignación de titulares a eventos, porque por ejemplo, un evento que tenga las palabras "Sánchez" y "triunfo" sólo tendrá *una* palabra en común con el caso anterior, ya que "Alexis Sánchez" es distinto de "Sánchez" a secas.

Por esto, la primera modificación que se efectúa a la clase *Tokenizer*, es que para entidades que tengan más de una palabra, se retornen todas las combinaciones posibles de las palabras que

[58]https://github.com/mquezada/Twitter-event-detection
[59]https://spacy.io/usage/linguistic-features
[60]https://spacy.io/models/es#es_core_news_sm

la componen. Para el mismo titular, la *tokenization* incluirá además ahora "Alexis" y "Sánchez", y por lo tanto, ahora las palabras en común entre este titular y el evento será de dos, no de uno. Para controlar este comportamiento, se añade el *flag individual_tokens* en la clase *Tokenizer*.

También se experimenta el proceso de *tokenization* con el vocabulario *es_dep_news_trf*[61], desarrollado por estudiantes de la Universidad de Chile y disponible en spacy, que contiene más palabras para el español. Sin embargo, carece de la identificación de entidades, por lo tanto, se descarta su uso.

Una modificación importante es el reemplazo de la lista de *stopwords*, que estaba definida para el vocabulario en inglés. Se busca una extensa propuesta de 732 *stopwords* para el español y se cambia[62].

Por otro lado, la función *detect_keywords* no se modifica en absoluto. Puede recibir conjuntos de titulares *tokenizados* y realizar la generación de eventos sin problema. Luego, la implementación del Algoritmo 2 de asignación a eventos, fue bastante sencilla de llevar a cabo.

Finalmente, todo este proceso se encapsula en una clase denominada *KeywordClustering*, que recibe como parámetro el *threshold*, y el *flag individual_tokens*. Su método *fit* recibe los titulares de noticias, en texto plano o ya *tokenizados*. Ejecuta la función *detect_keywords* para crear eventos, y luego mediante el algoritmo de asignación, asigna una *label* a cada titular.

Experimentando con este algoritmo, se encuentra un caso de interés: las elecciones de Perú, realizadas el día 11 de abril de 2021. Para 300 artículos recolectados entre el 10 y 11 de abril de 2021, mediante el uso de la heurística, se forman dos grandes *clusters* referentes de la elección: unos con titulares referentes a Keiko Fujimori y otro con titulares referentes a Pedro Castillo (Ver Tabla 4.12). Sin embargo, ambos *clusters* se componen de titulares con palabras bastante similares como: "elecciones", "voto rural" y "Perú". Además, titulares de un *cluster* tienen *keywords* del otro y viceversa. Si el evento es el mismo (elecciones en Perú), estos *clusters* deberían estar juntos. Además, otros cinco *clusters* más pequeños también se refieren a este tópico.

Para afrontar situaciones como esta, se propone usar el Algoritmo 3. La idea es ejecutar este algoritmo después de la asignación de titulares a eventos, para unir en forma iterativa aquellos *clusters* más similares, bajo el siguiente criterio de similitud: si un *cluster* A tiene un alto porcentaje de titulares que tienen *keywords* del *cluster* B, y viceversa, entonces estos *clusters* son altamente similares. Los *clusters* son entonces juntados, uniendo sus elementos y sus *keywords*.

Se implementa este algoritmo final, controlable mediante el *flag join_similar* en la clase *KeywordClustering*, y se observa una mejoría en los resultados. Ahora no sólo dos *clusters* principales referentes a las elecciones quedan en un mismo *cluster*, sino que un *cluster* más pequeño también es integrado al gran *cluster* sobre las elecciones en Perú, logrando el comportamiento deseado (Ver Tabla 4.13).

[61] https://spacy.io/models/es#es_dep_news_trf
[62] https://github.com/stopwords-iso/stopwords-es/blob/master/stopwords-es.txt

Algoritmo 3 Heurística de unión de eventos similares

Input: A set of N sets of words, $A = \{A_1, A_2, ..., A_N\}$,
a set of N sets of labels, $L = \{L_1, L_2, ..., L_N\}$,
a set of k sets of keywords, $E = \{E_1, E_2, ..., E_k\}$,
and positive integers β, γ acting as *main* and *weighted* thresholds

Output: N labels and E set of k keywords

$prev_L \leftarrow$ [] empty list
while $prev_L != L$ **do**
    $M \leftarrow k \times k$ empty matrix ▷ Construct $k \times k$ similarity matrix
    for every pair of indices $i, j \in \{1, 2, ..., k\}$ such that $i \neq j$ and $i > j$ **do**
        $h_A \leftarrow$ subset of elements $A_w \in A$ such that $L_w == i$ for $w \in \{1, 2, ..., k\}$
        $h_B \leftarrow$ subset of elements $A_w \in A$ such that $L_w == j$ for $w \in \{1, 2, ..., k\}$
        $M_{ij} \leftarrow |h_A|$ such that $|\hat{a} \cap E_j| > 0$ for $\hat{a} \in h_A$
        $M_{ji} \leftarrow |h_B|$ such that $|\hat{b} \cap E_j| > 0$ for $\hat{b} \in h_B$
    end for
    $join\_list \leftarrow$ [] empty list ▷ Apply heuristic
    for every pair of indices $a, b \in \{1, 2, ..., k\}$ such that $a \neq b$ and $a > b$ **do**
        $L_A \leftarrow$ |subset of elements $L_a \in L$| such that $L_a == a$
        $L_B \leftarrow$ |subset of elements $L_b \in L$| such that $L_b == b$
        if $L_A == 0$ **or** $L_B == 0$ **then**
            break
        end if
        if $M_{ab}/L_A \geq \beta$ **and** $M_{ba}/L_B > \beta$ **then**
            if $(M_{ab}/L_A + M_{ba}/L_B)/2 > \gamma$ **then**
                $join\_list \leftarrow join\_list + (min(a, b), max(a, b))$ ▷ Add tuple
            end if
        end if
    end for
    $prev_L \leftarrow L$ ▷ Change labels and keywords
    for tuple (a, b) in $join\_list$ **do**
        $L_b \leftarrow L_a$
        $E_a \leftarrow E_a \cup E_b$
        $E_b \leftarrow \{\}$ empty set
    end for
end while
return L, E

#	Keywords	Tamaño	Titulares del cluster
0	Fujimori, Keiko, Keiko Fujimori	26	Pelea voto a voto en Perú: Castillo recorta cada vez más su distancia con Fujimori Elecciones de infarto en Perú: conteos desvanecen ventaja de Keiko Fujimori Castillo se fortalece y Keiko se debilita minuto a minuto en inesperado conteo de votos en Perú Más que "de infarto": Castillo supera a Keiko en el conteo de las elecciones presidenciales peruanas Voto rural: la clave de Castillo sobre Keiko Fujimori Keiko Fujimori denuncia intentos de "boicotear la voluntad popular" en segunda vuelta presidencial [...]
1	Pedro Castillo, Pedro, Castillo	6	Mercados peruanod sufren notoria caída ante repunte de Pedro Castillo en elecciones presidenciales Comicios en Perú: Castillo dice que será el primero en hacer respetar "la voluntad popular" tras acusación ... Avance de Castillo hacia la Presidencia de Perú sacudió la bolsa y disparó el dólar: Análisis al impacto so ... La importancia del voto rural, el factor que podría darle la victoria a Pedro Castillo en Perú. [...]
2	resultados, Perú	1	Presidente de Perú y resultados electorales en su país: "Son una clarinada de alerta"
3	votos, Perú	1	Ante lento conteo de votos en Perú, editorial del diario El Comercio llama a la "calma"
4	mercados, peruanos, Castillo caída	1	Mercados peruanos profundizan caída ante la ventaja que tomó Castillo frente a Fujimori ...
5	inesperado, Perú	1	Crónica de un domingo inesperado en Perú: el vilo de unos comicios "infartantes"
6	elecciones, Perú	2	El Mostrador en la Clave: la demanda que afecta al Grupo Saieh, el análisis de los proyectos de buscan ... Acciones de empresas chilenas con gran exposición en Perú caen con fuerza y golpean al IPSA tras elecc ...

Tabla 4.12: Resultados de *clustering* con heurística sin optimización

#	Keywords	Tamaño	Titulares del cluster
0	Fujimori, Keiko, Keiko Fujimori, Pedro Castillo, Pedro, Castillo, peruanos, caída, mercados	26	Pelea voto a voto en Perú: Castillo recorta cada vez más su distancia con Fujimori Elecciones de infarto en Perú: conteos desvanecen ventaja de Keiko Fujimori Castillo se fortalece y Keiko se debilita minuto a minuto en inesperado conteo de votos en Perú Más que "de infarto": Castillo supera a Keiko en el conteo de las elecciones presidenciales peruanas Voto rural: la clave de Castillo sobre Keiko Fujimori Keiko Fujimori denuncia intentos de "boicotear la voluntad popular" en segunda vuelta presidencial Mercados peruanos sufren notoria caída ante repunte de Pedro Castillo en elecciones presidenciales Comicios en Perú: Castillo dice que será el primero en hacer respetar "la voluntad popular" tras acusación ... Avance de Castillo hacia la Presidencia de Perú sacudió la bolsa y disparó el dólar: Análisis al impacto so ... La importancia del voto rural, el factor que podría darle la victoria a Pedro Castillo en Perú Mercados peruanos profundizan caída ante la ventaja que tomó Castillo frente a Fujimori ... [...]
2	resultados, Perú	1	Presidente de Perú y resultados electorales en su país: "Son una clarinada de alerta"
3	votos, Perú	1	Ante lento conteo de votos en Perú, editorial del diario El Comercio llama a la "calma"
5	inesperado, Perú	1	Crónica de un domingo inesperado en Perú: el vilo de unos comicios "infartantes"
6	elecciones, Perú	2	- El Mostrador en la Clave: la demanda que afecta al Grupo Saieh, el análisis de los proyectos de buscan ... - Acciones de empresas chilenas con gran exposición en Perú caen con fuerza y golpean al IPSA tras elecc ...

Tabla 4.13: Resultados de *clustering* con heurística con optimización

Se tiene la teoría de que este último algoritmo es útil para agrupar eventos que impliquen grandes y extensas coberturas, como lo son los eventos masivos, guerras, etc.

Se observa además, que los parámetros β (*main_threshold*) y γ (*weighted_threshold*) del Algoritmo 3, son sensibles. Si estos valores son muy pequeños, entonces se forma un solo *cluster* que contiene casi todos los artículos del *input*. Por otro lado, valores mayor a 0.5, no provocan ningún efecto sobre el resultado original. Sólo mediante experimentación, se estima que un valor de 0.3 y 0.5 para β (*main_threshold*) y γ (*weighted_threshold*) respectivamente, son adecuados.

#	Titulares	Etiqueta
0	La última entrevista de Humberto Maturana: "Las doctrinas son enemigas de la reflexión" 5 reflexiones del fallecido Humberto Maturana que evidencian su legado. Discípulo de Maturana tras muerte: "Su vida intelectual es trascendente, qué duda cabe, inspiradora". ¿Qué es la autopoiesis? La teoría de Humberto Maturana que le dio prestigio y fama internacional. Muere Premio Nacional de Ciencias Humberto Maturana a los 92 años. Piñera decreta duelo y lamenta muerte de Maturana: "Uno de los grandes pensadores de nuestro país". Piñera decreta duelo oficial por muerte de Humberto Maturana. Chile despide al científico más influyente de los últimos años: A los 92 años fallece Humberto Maturana.	Muerte H. Maturana
1	Pablo Chill-E queda con arresto domiciliario total tras ser detenido en persecución en La Dehesa.	Pablo Chill-E
2	Eduardo Frei y expresidentes de Iberoamérica acusan "ruptura del estado de derecho" en El Salvador.	Eduardo Frei
3	Senado pide a Piñera solicitar a Israel que vacune a la población palestina.	Israel
4	28 comunas avanzan a Transición: Todos los cambios en el Plan Paso a Paso anunciados por el Minsal. Plan Paso a Paso: ¿Qué comunas salieron de su cuarentena hoy jueves 6 de mayo?. Cuarentena en Chile: ¿Cuáles son las comunas que retrocedieron a la Fase 1?. Plan Paso a Paso: ¿Cuáles son las comunas que salieron de cuarentena total?.	Cuarentenas y Paso a paso
5	Paulina Núñez responde a críticas por "carnet verde", "No pretendemos instalar la obligatoriedad de la vacuna". Ministerio de Economía anuncia plan especial de vacunación para trabajadores y dispone sitio para quejas.	Cacunas
6	Minsal compara indicadores en la previa de elecciones: Menos casos activos, pero más hospitalizados. Informe Epidemiológico: Casos activos bajan la barrera de los 50 mil por primera vez desde marzo.	Casos activos
7	OMS admite su lenta respuesta ante el COVID-19 en los primeros meses de pandemia. Cuarentenas, cordones sanitarios y prohibiciones: Las medidas activas en Chile por el COVID-19. Israel reconoce error y descarta "variante chilena" del COVID-19. Fundadores de BioNTech esperan que la pandemia dure hasta mediados de 2022. 3.198 casos nuevos de COVID-19: Se registró la cifra más baja desde marzo. Director (s) ISP: "Descartamos que exista una variante chilena" de COVID-19. Minsal informa más de 6.000 casos diarios de COVID-19: 40 comunas avanzan en el Plan Paso a Paso.	COVID-19
8	Daza: Conversaciones entre UC y Sinovac por producción de vacunas en Chile son "preliminares"	Sinovac

Tabla 4.14: Muestra de 27 titulares y 9 tópicos para evaluar preliminarmente la heurística

Además, se realiza un pequeño experimento a modo de encontrar preliminarmente la mejor combinación de los parámetros *threhold*, el *flag join_similar* y el *flag individual_tokens*, y también para evidenciar la importancia de eliminar las *stopwords*. Se hace una selección manual de 27 titulares extraídos entre el 6 y 7 de mayo de 2021, y se añaden 9 etiquetas (*labels* en inglés) para poder evaluar las agrupaciones en forma supervisada. Esta agrupación manual se puede revisar en la Tabla 4.14.

Se utilizan las métricas *rand score*, *completness*, *homogeneity* y *V measure* [81] para evaluar la calidad de los *clusters* encontrados. La regla general para estas métricas, es que un valor mayor representa mayor calidad. Los resultados explícitos de este experimento (las *labels*) se encuentran en Anexo C.1, mientras que la evaluación de los mismos bajo las métricas señaladas, se encuentran en la Tabla 4.15.

Los resultados muestran la importancia de eliminar las *stopwords*, debido a que es evidente que no filtrarlas baja bastante la calidad de los resultados. Por otro lado, el *flag join_similar* mejora los resultados en el caso de *threshold=2* e *individual_tokens=False*. Sin embargo, para ese mismo valor de *threshold*, no es claro que el *flag individual_tokens* mejore los resultados.

Por otro lado, un valor de *threshold* de 1 muestra obtener buenos resultados, pero se tiene la hipótesis de que relajar esta condición demasiado podría generar *clusters* con mucho ruido. Por otro lado, un *threshold* de 3 y 4 demuestra ser muy exigente con los titulares, ya que hay titulares que tienen pocas palabras y, por lo tanto, es aún más difícil que tengan 3 o 4 palabras en común con otros. En consecuencia, se concluye que 2 es un buen valor para *threshold*.

Configuración	Completness	Homogeneity	Rand	V measure
threshold=1 individual_tokens=False join_similar=True	0.736	0.802	0.9	0.768
threshold=1 individual_tokens=True join_similar=True	0.736	0.802	0.9	0.768
threshold=2 individual_tokens=False join_similar=True	**0.760**	0.858	**0.908**	**0.806**
threshold=2 individual_tokens=False join_similar=False	0.691	0.905	0.877	0.784
threshold=2 individual_tokens=True join_similar=True	**0.771**	**0.912**	**0.923**	**0.836**
threshold=2 individual_tokens=True join_similar=False	**0.771**	**0.912**	**0.923**	**0.836**
threshold=3 individual_tokens=False join_similar=False	0.678	0.905	0.886	0.775
threshold=3 individual_tokens=False join_similar=True	0.71	0.905	0.905	0.796
threshold=4 individual_tokens=False join_similar=False	0.6	0.955	0.846	0.737
threshold=4 individual_tokens=False join_similar=True	0.6	**0.955**	0.846	0.737
threshold=2 individual_tokens=False join_similar=True *Sin filtro de stopwords	0.52	0.175	0.415	0.262
threshold=5 individual_tokens=False join_similar=True *Sin filtro de stopwords	0.694	0.615	0.7	0.652
threshold=7 individual_tokens=False join_similar=True *Sin filtro de stopwords	0.561	0.839	0.811	0.673

Tabla 4.15: Métricas de calidad de *clustering*, para heurística sobre pequeña muestra de 27 titulares y 9 tópicos distintos

Lo destacable, es que en cualquiera de estos casos, el proceso de agrupamiento completo no demora más de 0.2 segundos, lo cual es tremendamente eficiente.

Con todo esto, se considera que la configuración *threshold=2*, *individual_tokens=False* y *join_similar=True* es la mejor.

Finalmente, es prudente concluir que, *a priori*, los resultados de esta metodología son prometedores, tanto por su eficiencia como en su calidad. Notar que el mecanismo de agrupación garantiza que los artículos de cada evento contengan al menos *threshold* palabras en común con las *keywords* de ese evento, y por lo tanto, es altamente probable que los artículos de cada evento sea del mismo tema. Aún más, si los artículos son del mismo día. En la Sección 5.1.2 se profundiza en la calidad de los *clusters* elaborados por esta metodología.

4.5.3. *Clustering service*

Se implementa un pequeño servidor en Flask, que mediante un único *endpoint* denominado */genCluster*, permite ejecutar todo el proceso de *clustering* de artículos y guardar los resultados en Elasticsearch. Este servicio cuenta con instancias de las clases *QueryMaker*, *EmbeddingClustering* y *KeywordClustering*. La primera permite efectuar solicitudes de lectura y escritura a la base de datos (se detalla en la Sección 4.6.1), la segunda hacer agrupaciones mediante los *embeddings* y la tercera realizar agrupaciones mediante la heurística.

El *frontend* es el único componente que interactúa con este servicio. Mediante una solicitud tipo POST, se especifican los dos parámetros principales para efectuar una agrupación: rango de fechas y método de *clustering* (*embeddings* o heurística), y también todos los parámetros configurables para cada método. Por un lado, para *embeddings* es posible especificar el algoritmo a utilizar, el criterio de elección no supervisado, el número mínimo de *clusters* y el porcentaje de *outliers*. Por otro lado, para la heurística se puede especificar el *threshold*, y los *flags join_similar* (para ejecutar o no el Algoritmo 3) e *individual_tokens* (para usar sólo palabras separadas).

Dependiendo de la configuración espeficada en el *frontend*, se usa la clase *EmbeddingClustering* o *KeywordClustering*. Por cada *cluster* identificado se crea un objeto tipo *Evento* que coincida con el *mapping* definido en la Sección 3.5.1. Luego, se genera el objeto *Agrupación*, donde se coloca la lista de *ids* de los *Eventos* creados, junto a los parámetros de configuración. Finalmente, la respuesta de este servicio es la *id* de la *Agrupación*, que luego el *frontend* usa para mostrar los resultados de la agrupación.

Notar que la generación de título y resumen de cada evento, queda delegada al *Summary service*, por lo tanto, los eventos son guardados con el *flag has_summary* en falso. La idea es que el *Control service* detallado en la Sección 4.9.1, identifique los eventos sin resumen ni título en la base de datos, y proceda a hacer las llamadas al *Summary service* para su cálculo. Por esto, las agrupaciones recientes no tienen título ni resumen.

Una debilidad de este servicio es la carencia de *feedback* al usuario en el *frontend*, por lo que es imposible conocer en qué fase se encuentra la agrupación en cada momento. Por otro lado, al ocurrir un error se retorna un mensaje genérico, lo cual hace difícil el *debugging* el código.

4.6. Base de datos

4.6.1. Clase *QueryMaker*

Se utiliza el cliente de Python de Elasticsearch[63] para realizar todas las interacciones con la base de datos. Este cliente permite abstraer las solicitudes HTTP planas mediante métodos simples. Se crea un objeto llamado *Elasticsearch* especificando el *host* y *url* de la base de datos. Este objeto permite, por ejemplo, efectuar una búsqueda sobre un *index*, mediante el método *search* donde se especifica el *index* y el cuerpo de la búsqueda en formato JSON.

En primer lugar, se desarrolla una serie de métodos para crear y modificar los *index* de *Artículos*, *Eventos* y *Agrupaciones*, así como de la carga de documentos de ejemplo. Si bien se provocan algunos problemas con el formato de la fecha, estos son superados. Por otra parte, para traer todos los resultados de un *index*, se hace necesario investigar del parámetro *scroll*, que básicamente actúa como paginador de resultados. Para ver el reflejo de estas operaciones en Elasticsearch, se usa el *software* Kibana, que permite visualizar y navegar en los documentos presentes en la base datos en forma sencilla. Por ejemplo, en la Figura 4.15 se usa Kiabana para ver la distribución en el tiempo de los artículos extraídos, mientras que en la Figura 4.16 se observa el volumen de artículos para el medio La voz de los que sobran, en una ventana de 20 días.

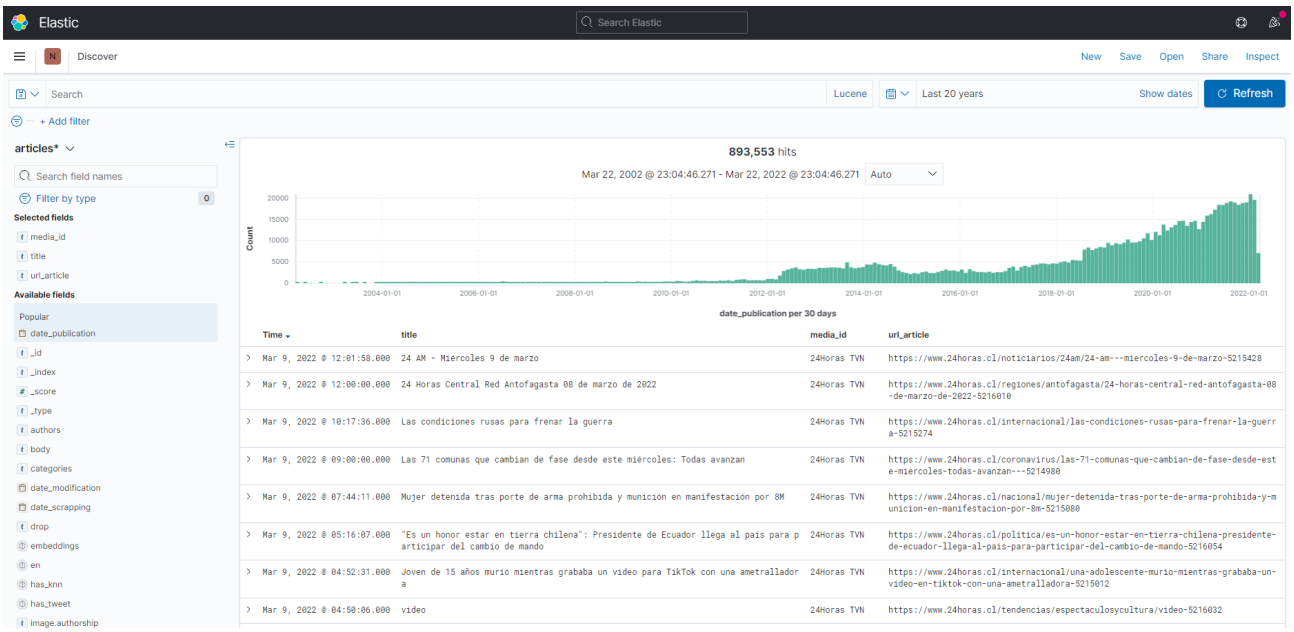

Figura 4.15: Kibana: volumen de los 893.553 artículos extraídos en el tiempo

[63]https://elasticsearch-py.readthedocs.io/en/v8.1.1/

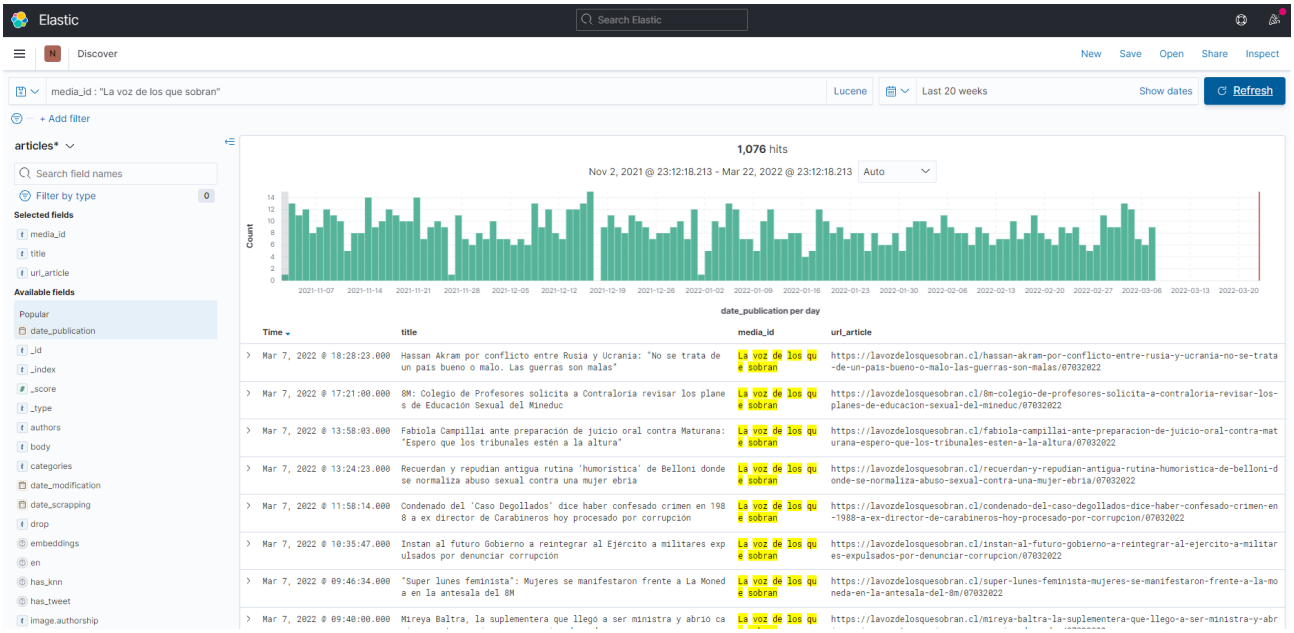

Figura 4.16: Kibana: volumen artículos para el medio La voz de los que sobran, en una ventana de 20 días.

La gran variedad de consultas, hace necesaria la creación de la clase *QueryMaker*. Esta clase recibe una instancia del cliente *Elasticsearch* y permite acceder a una serie de métodos. Dentro de los más importantes se encuentran: *run_query_all_indices*, que permite para obtener todos los documentos de algún *index* en forma genérica; *searchWithWords*, el principal método utilizado por el buscador avanzado; *filter_nlp*, para filtrar resultados de búsqueda por métricas PLN; *article_exists* para verificar la existencia de un artículo y una serie de *getters* para *Eventos*, *Agrupaciones* y *Artículos*.

Estas consultas son usadas en distintas partes del sistema. Por ejemplo, el método *article_exists* lo usan los *scrappers* para evitar duplicados en la base de datos; *searchWithWords* es utilizado por el servicio del buscador y los *getters* son generalmente utilizados por el *Clustering service*.

La implementación de cada uno de estos métodos se encuentra documentada *in situ*. El propósito de tener una clase que encapsule todos estos tipos de consultas, es que basta con instanciar un objeto *QueryMaker* para poder acceder a todas las funcionalidades desarrolladas, lo cual es un proceso bastante directo y sencillo.

Es importante señalar que se toma la decisión de no implementar algunas funcionalidades relacionadas al *QueryMaker*. Por ejemplo, el método que dado un rango de fechas, un *article_id*, y un número k, retorne los *k-nearest neighors* (KNN) artículos vecinos, para el artículo dado. Este método debe ser usado por el *KNN service* que tampoco es implementado. Esto sucede por dos motivos: (1) se priorizan otros componentes del sistema, y (2) no es directo establecer una visualización clara y simple en el *frontend*. Todo esto queda planteado para un futuro.

4.6.2. *Query service*

Este servicio es principalmente utilizado por el *frontend*. Utiliza Flask, cuenta con una instancia de *QueryMaker* y tiene tres *endpoints* principales: */clusterings*, */getCluster* y */getEvent*. El primer *endpoint* brinda al *frontend* de todos las agrupaciones guardadas en Elasticsearch, se usa en la Vista lista de agrupaciones. El segundo *endpoint* trae una agrupación en particular, con todos los eventos pertenecientes a esta agrupación, y todos los artículos pertenecientes a cada evento, se utiliza en la Vista agrupación (Ver Sección 3.8.2). Por último, el tercer evento trae toda la información de un evento, y de cada artículo asociado, se emplea en la Vista detalle de evento (Ver Sección 3.8.2).

4.7. Buscador avanzado

4.7.1. Método principal

Como se menciona anteriormente, el método *searchWithWords* dentro del *QueryMaker*, es el principal método usado por el buscador avanzado de artículos. Este genera una consulta acumulativa a realizar a la base de datos. En primer lugar, incluye en la consulta un filtro por rango de fechas, luego genera una *query* recursiva respecto a las palabras a buscar en los artículos, considerando los *sources* que se indican en la búsqueda: *title, drop, body, tags* o *categories* inclusive.

Esta consulta recursiva se implementa en forma separada, *gen_ text_ query* toma un *source* junto a tres grades conjuntos de palabras: aquellas especificadas en los parámetros "frase exacta", aquellas especificadas como "palabras a incluir" y por último, las especificadas como "palabras a excluir". Para incluir todos los *sources* basta ejecutar este método para cada *source* y unir las consultas mediante el operando AND.

Este método elimina todos los espacios en blanco en las palabras proveídas. Luego, arma una *query* acumulativa: primero toma las palabras de "frase exacta" y genera una *query* del estilo: "(title: palabra1) AND (title: palabra2) ...", que permite buscar una frase exacta en el *source* tipo *title*, en este caso. Después, se forma la *query* conformada por las palabras de "palabras a incluir" del estilo "(title: palabra1) OR (title: palabra2) ...", que permite buscar de manera menos estricta, documentos que incluyan alguna de las palabras especificadas. Finalmente, con el último conjunto de palabras se forman *queries* del estilo "(NOT (title: palabra1)) AND (NOT (title: palabra2)) ... ", para buscar textos cuyo *title* no tengan las palabras especificadas en "palabras a excluir". Finalmente, se ejecuta un operador AND entre estas tres *queries* y la *query* global queda lista.

Hasta este punto, el filtro por fecha y las *queries* de palabras para cada *source* están listos. El filtro por medios, se logra mediante una *query* del estilo "(media_id: media1) OR (media_id: media2) OR ..." donde media1, media2, etc. son los medios especificados en la consulta. Se procede de modo similar para generar una *query* que filtre por autor. Finalmente, si el usuario especifica el filtro de los artículos que tienen un *tweet* asociado, se añade una última *query* del estilo "(has_tweet: true)". Todas estas *queries* se juntan mediante el operando AND, para

que tengan efecto en forma simultánea, y luego se utiliza el método *run_ query_ all_ indices*, de *QueryMaker*, para ejecutar la primera búsqueda y traer toda la información de cada artículo.

Como las métricas PLN y de Twitter son en realidad una lista de objetos, no se pueden aplicar los filtros numéricos de Elasticsearch directamente, y deben ser procesados aparte. Este es el coste de la flexibilidad y escalabilidad alcanzada por el *mapping* actual de los artículos.

El buscador permite incluir una serie de filtros numéricos definidos por las tuplas *(parameter, criteria, value)*, donde *parameter* puede ser: *subjectivitty, positivity, neutrality, negativity*, *retweets* o *likes*. Por su parte, *criteria* puede ser: menor, menor o igual, igual, mayor o igual, o mayor. Y *value* corresponde al valor del parámetro a filtrar.

Para aplicar cada uno de estos filtros numéricos, se procede de manera iterativa: para cada artículo de los resultados de la primera búsqueda, se ejecuta el método *filter_ nlp* presente en *QueryMaker*. Este método itera sobre todos los artículos, identifica el parámetro en cuestión, extrae el valor del artículo y según el criterio especificado, lo compara con el valor entregado por el usuario. Si la condición se cumple, el artículo se incluye como resultado, y si no se excluye.

Finalmente, el resultado de esta segunda búsqueda habrá aplicado todos los filtros especificados inicialmente, y se retornará una lista de artículos que cumplen con la búsqueda.

4.7.2. *Search service*

Este servicio es utilizado únicamente por el *frontend*. Implementado con Flask, contiene una instancia de *QueryMaker* que en su único *endpoint* denominado */search*, recibe mediante una *request* tipo POST, los filtros y parámetros de búsqueda necesarios para ejecutar el método *searchWithWords*. La respuesta es un JSON simple, que contiene una lista con todos los artículos que se obtienen de la búsqueda.

Esta implementación carece de paginamiento, por lo que se provoca un *overload* en el paquete de respuesta de existir muchos resultados de búsqueda. Esto es totalmente perfectible, pero no se modifica por falta de tiempo. Sin embargo, los parámetros innecesarios a mostrar en los resultados, como los *embeddings*, traducciones y *flags* son removidos de la respuesta.

La razón de que exista un servicio dedicado únicamente a la búsqueda de información, es la posibilidad de realizar escalamiento horizontal. Múltiples instancias de este servicio pueden, eventualmente, satisfacer la demanda de miles de solicitudes.

4.8. Visualización

4.8.1. Herramientas y metodología

Se decide utilizar ReactJS[64] para el desarrollo del *frontend*, dada la alta experiencia del estudiante con el *framework*.

ReactJS se usa junto a MaterialUI[65], el *framework* más popular para ReactJS utilizado por empresas como la NASA, Walmart y UNIQLO. En palabras simples, permite utilizar componentes como botones, selectores, campos de texto e iconos, predefinidos. Estos componentes son totalmente personalizables y permiten ahorrar tiempo en la implementación de la lógica de la aplicación. Además tienen un diseño que sigue la línea de diseño de Google, y por lo tanto, representa un *plus* estético.

Para el desarrollo de gráficos, se usa ChartJS[66], *framework* también popular. Mediante un proceso simple y directo permite hacer gráficos de barra, de *pie*, de línea, etc.

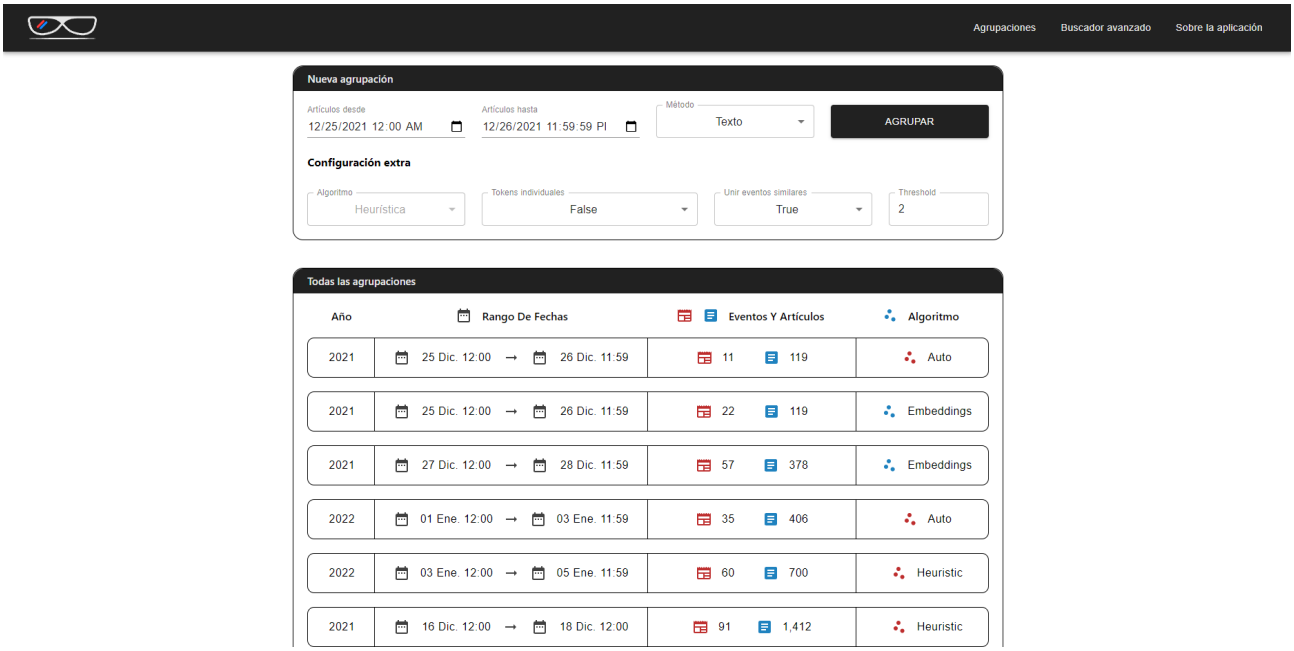

Figura 4.17: Vista lista de agrupaciones: parámetros para método de heurística

En las Figuras 4.17 y 4.18 se aprecia la implementación de la Vista lista de agrupaciones. La primera muestra los parámetros configurables para efectuar una nueva agrupación mediante la heurística, y la segunda mediante los *embeddings*. Como se aprecia, mediante el uso de selectores y casillas de texto, se pueden modificar varios parámetros del proceso de agrupación no supervisada, como el criterio de selección que en estos casos muestra *Silhouette score*.

[64]https://es.reactjs.org
[65]https://mui.com
[66]https://www.chartjs.org

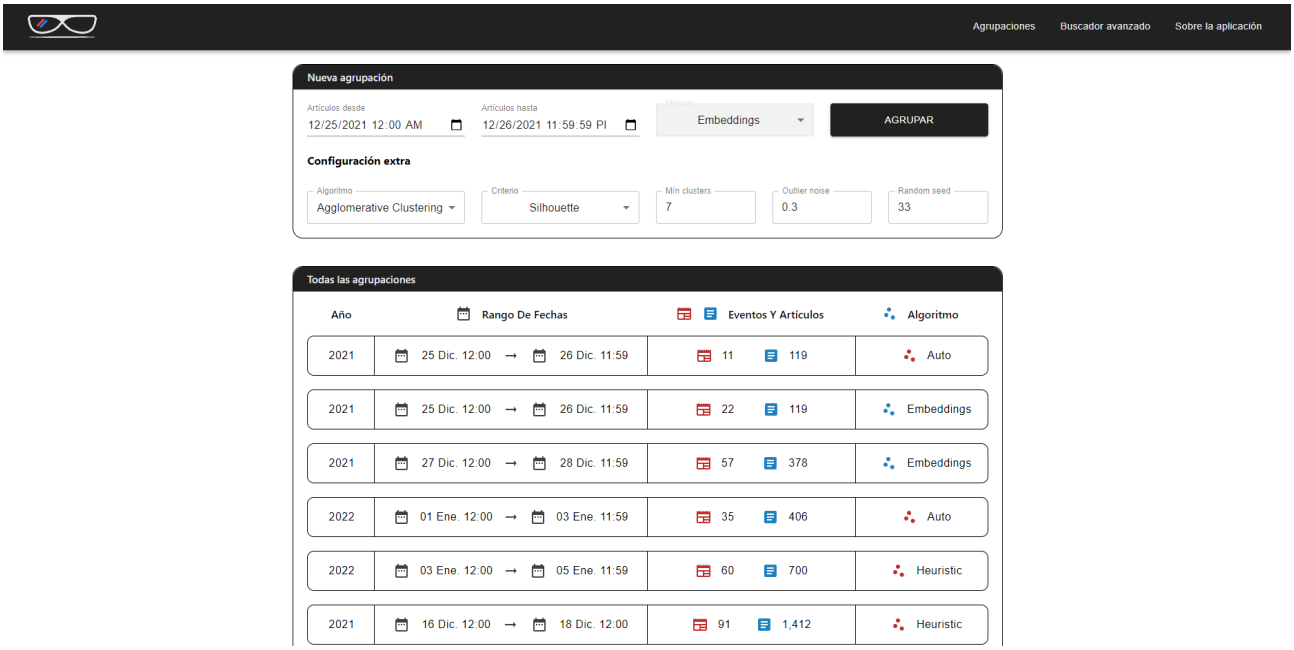

Figura 4.18: Vista lista de agrupaciones: parámetros para método de *embeddings*

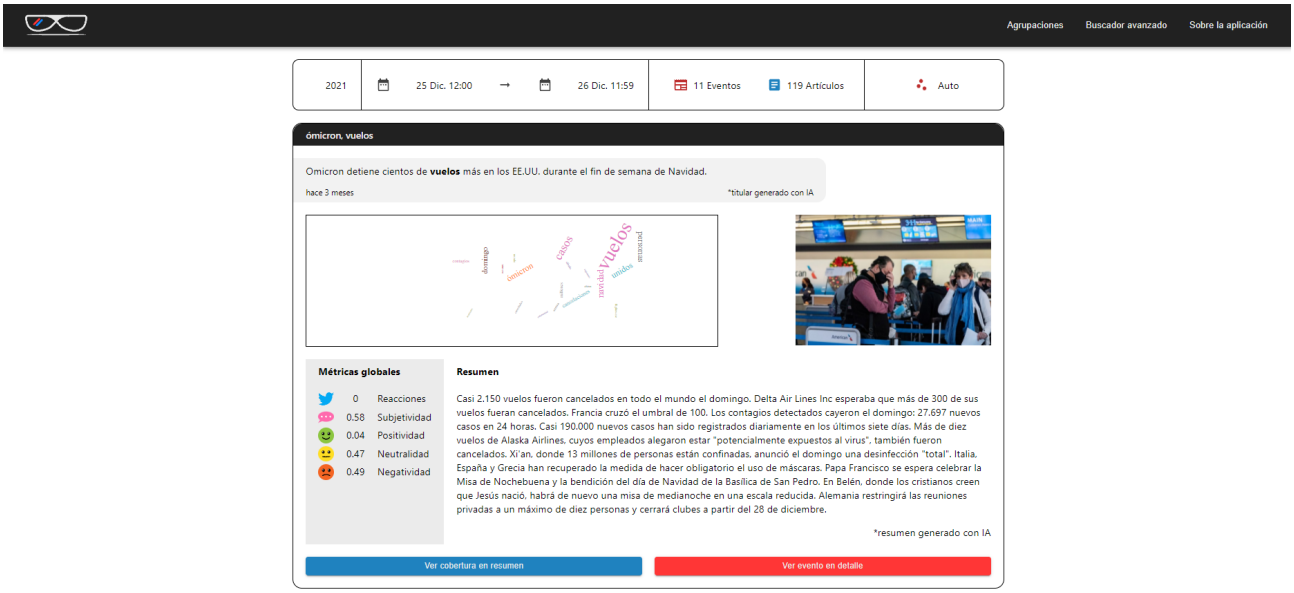

Figura 4.19: Vista agrupación: *layout* de tarjetas de evento.

Las Figuras 4.19 y 4.20 muestra un *layout* de eventos para una agrupación hecha entre el 25 y 26 de diciembre de 2021. Se puede apreciar de estas figuras que la implementación queda un poco distinta a los *mockups* propuestos inicialmente (Ver Figura 3.2), debido al cambio en la paleta de colores, la ubicación de las métricas globales y la incorporación del resumen (no presente en los *mockups*). La simbología tampoco está implementada, y los artículos se muestran como en los *mockups* de la Vista detalle de evento (Ver Figura 3.3). Estas diferencias

se producen por priorizar otras tareas, o bien, porque se determina que la implementación hecha es mejor a los *mockups* en términos de usabilidad (como la paleta de colores y la visualización de los artículos).

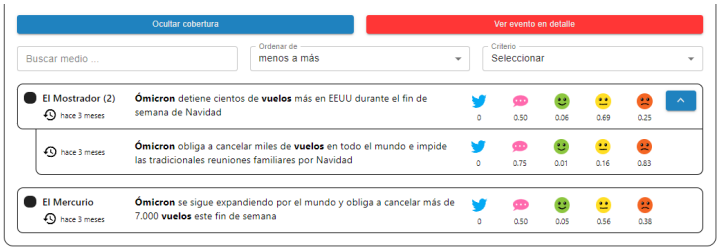

Figura 4.20: Vista agrupación: lista de artículos sobre un evento.

Por otro lado, cabe destacar que, los filtros de búsqueda de medios y de orden de artículos según las métricas del sistema (subjetividad, positividad, neutralidad y negatividad), quedan totalmente funcionales y permiten al usuario ordenar la información de acuerdo al criterio que estime conveniente.

En la Figura 4.21 se muestra la Vista detalle de evento: en ella se aprecia una vista similar a como se muestran los eventos en la Vista agrupación, pero ahora agregando los *tags* y categorías. Por otro lado, los artículos asociados a este evento son mostrados directamente después del resumen, y con los mismos filtros de orden. Debajo de la lista de artículos, se encuentra un gráfico de volumen de artículos por medio y un gráfico de frecuencia de palabras (Ver Figura 4.22). Más abajo, se muestra un gráfico de volumen de artículos en el tiempo, junto al primer y último artículo del evento (Ver Figura 4.23).

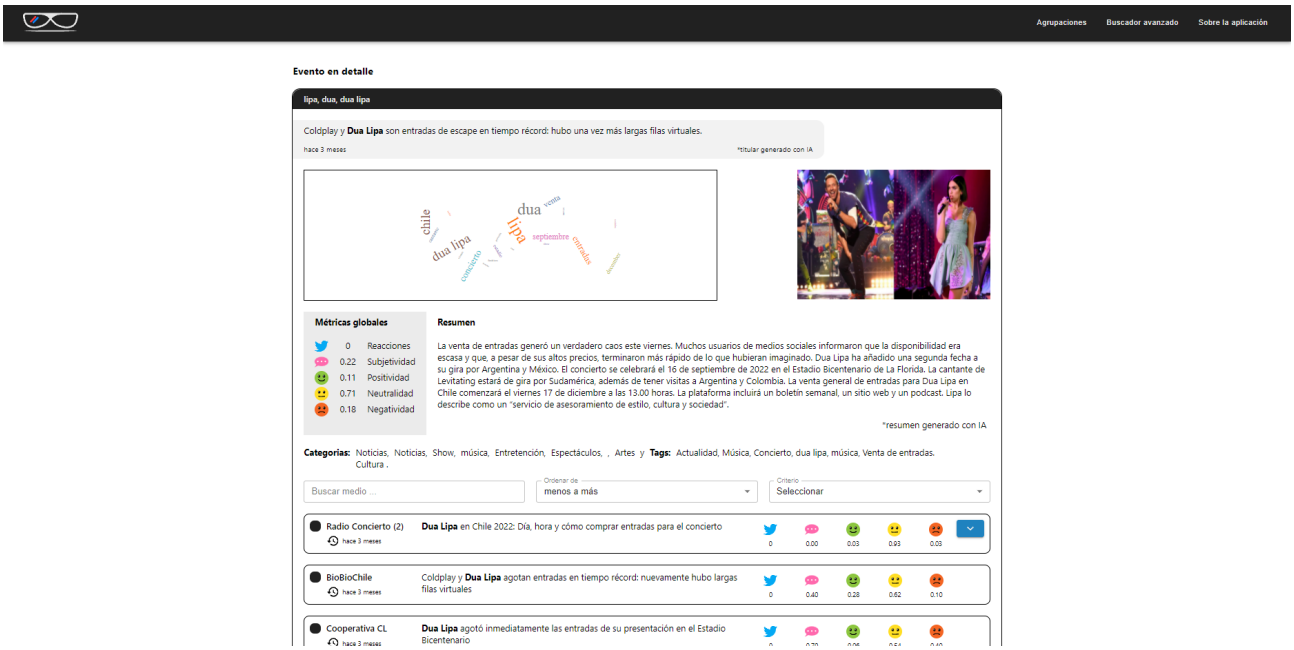

Figura 4.21: Vista detalle de evento: elementos generales y artículos

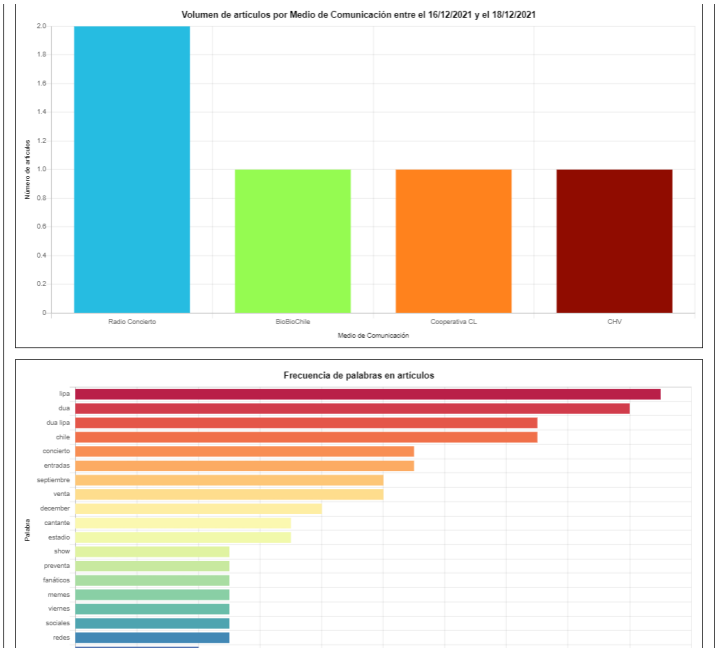

Figura 4.22: Vista detalle de evento: gráfico de volumen de artículos por medio, junto gráfico de frecuencia de palabras

Figura 4.23: Vista detalle de evento: gráfico de volumen de artículos en el tiempo, junto al primer y último artículo

Luego, se muestra un gráfico *pie* donde se observa el sentimiento predominante del evento (Ver Figura 4.24), junto a un gráfico polar alternativo (Ver Figura 4.25). Finalmente, si el usuario lo desea, se muestra un histograma de distribución de frecuencias para la subjetividad y polaridad (Ver Figura 4.26). Este último gráfico queda oculto y se activa mediante un *switch*.

Figura 4.24: Vista detalle de evento: gráfico de polaridad

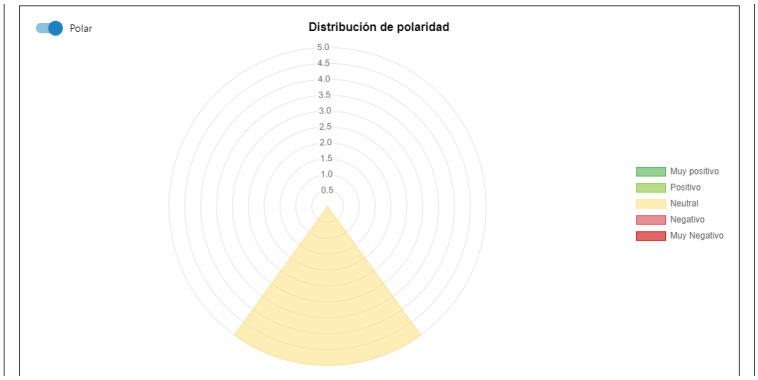

Figura 4.25: Vista detalle de evento: gráfico de polaridad alternativo

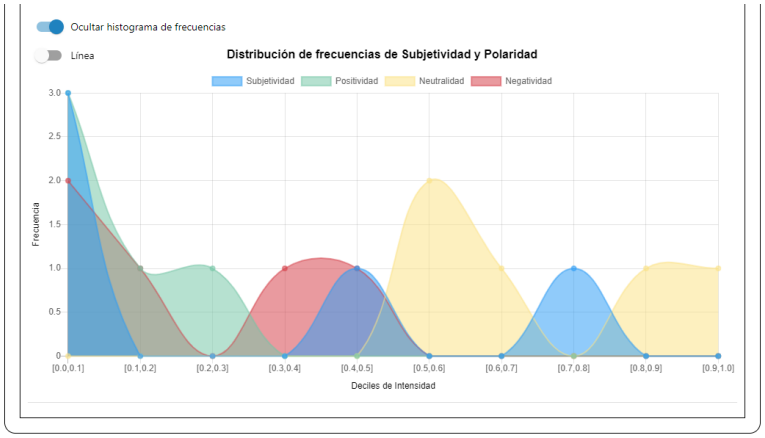

Figura 4.26: Vista detalle de evento: histograma de distribución de frecuencias para subjetividad y polaridad

En general, las vistas son implementadas de manera fidedigna a los *mockups* originales (Ver Figuras 3.3, 3.4 y 3.5), sin embargo, dado que el gráfico del histograma no tiene una utilidad directa, o bien es muy complejo, se decide por implementar el gráfico *pie*/polar para mostrar los sentimientos predominantes del evento en forma simple: elemento que no estaba en los *mockups*. Por otro lado, el paginamiento de los artículos presente en los *mockups* no se implementa por falta de tiempo, al igual que la simbología.

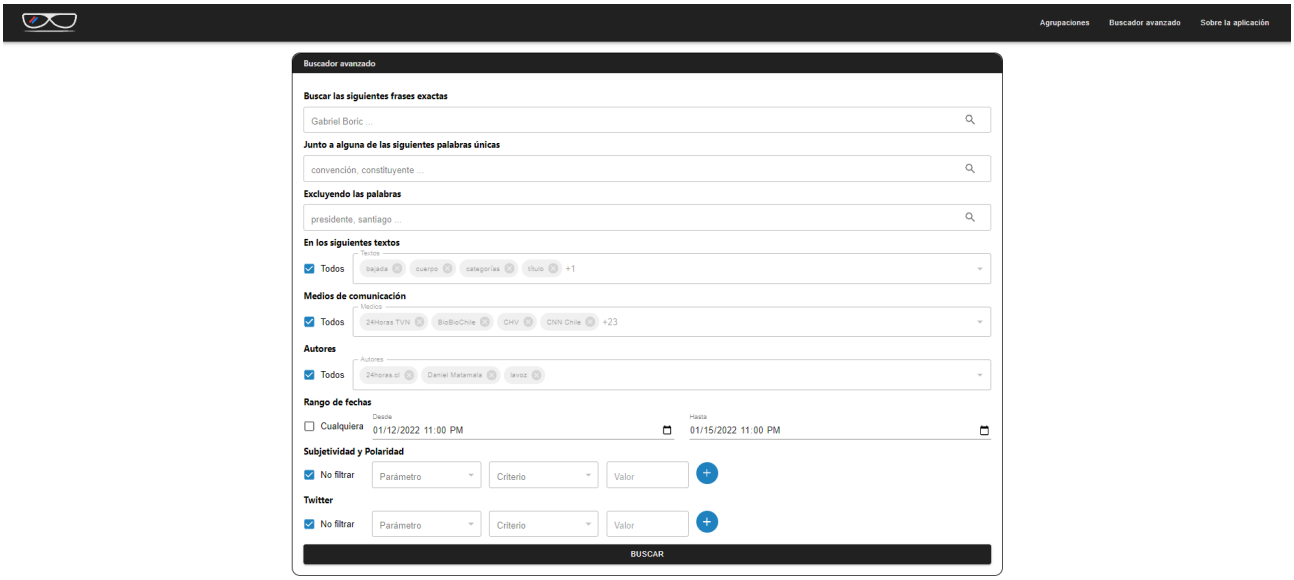

Figura 4.27: Vista buscador: filtros disponibles

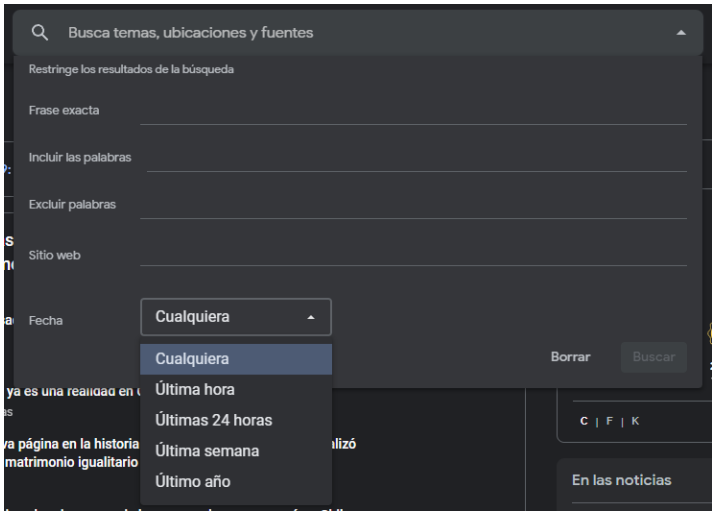

Figura 4.28: Google News: buscador

En la Figura 4.27 se ven todos los filtros disponibles para la Vista buscador. Notar que hay muchos más filtros que el de Google News (Ver Figura 4.28). Se destacan los filtros numéricos de subjetividad y polaridad en los filtros, que permiten buscar artículos según los sentimientos

predominantes y el grado de interpretación de los hechos. También destaca el filtro de Twitter que permite buscar artículos según su número de *retweets* o *likes*, para aquellos artículos que están asociados a un *tweet*.

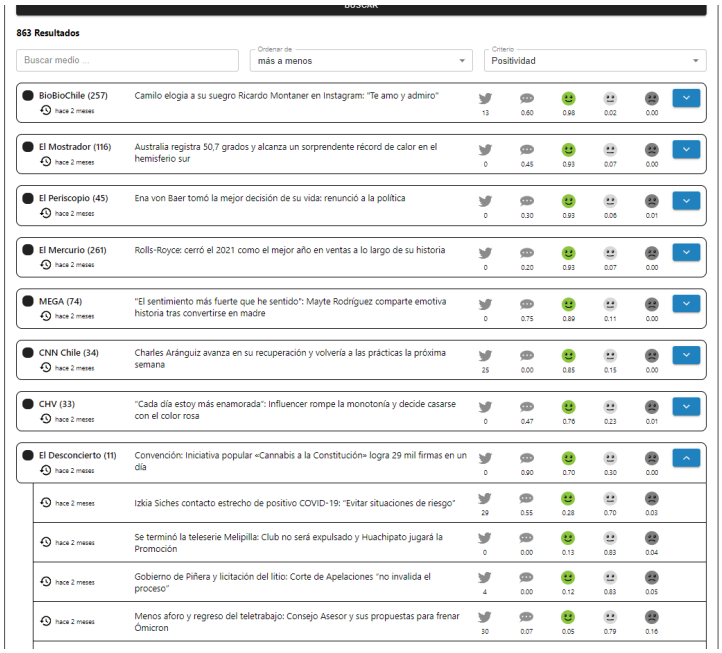

Figura 4.29: Vista buscador: ejemplo de resultados ordenados por positividad

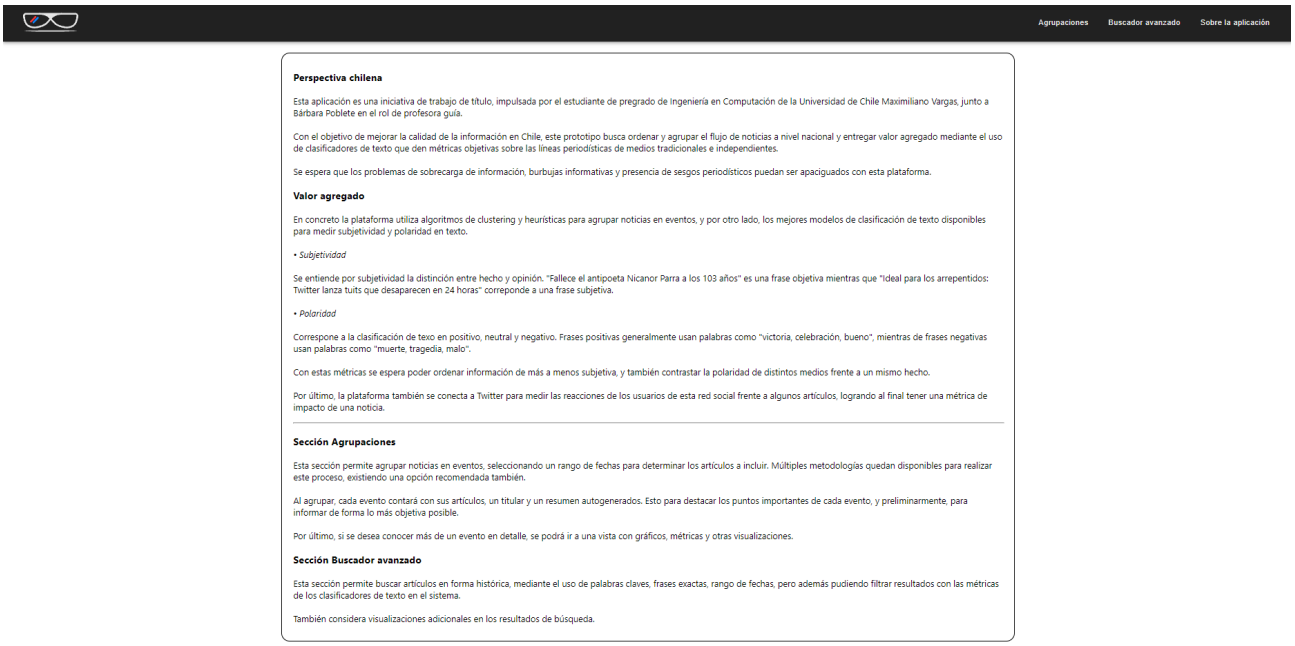

Figura 4.30: Vista informativa

En la Figura 4.29 se muestra un ejemplo de resultados ordenados por positividad. Los artículos se agrupan por medio para evitar un desorden. En las figuras no se muestra, pero al final de esta vista también se incluyen gráficos de volumen de artículos por medio, volumen de artículos en el tiempo y un gráfico de polaridad, que permiten identificar a los medios que publicaron más artículos en el periodo, y también identificar el sentimiento predominante del mismo.

Finalmente, una captura de la Vista informativa se aprecia en le Figura 4.30, que se compone de párrafos explicando las dos secciones principales del *software*. También da ejemplos de subjetividad y polaridad, y una breve introducción respecto de la motivación del *software*. Sin duda alguna, esta vista necesita mejora porque no es interactiva y contiene mucho texto, en el escenario ideal debería haber un video con animaciones para explicar las funcionalidades principales del *software* por ejemplo.

En general, el proceso de implementación fue sencillo de abordar, pues se tenía experiencia en el uso de estas herramientas. Dentro de las principales dificultades, se encuentra la implementación de los gráficos, ya que se debe hacer una agrupación de los artículos por medio, antes de graficar. Por su parte, el histograma de distribuciones también contaba con una agrupación previa, esta vez por métrica, que fue un poco compleja de implementar. Sin embargo, se obtienen los resultados esperados para cada gráfico.

Sin embargo, una debilidad importante del *frontend* es la rigidez de los parámetros PLN. Como se aprecia en la Figura 4.29, los artículos muestran un solo resultado de polaridad, cuando en realidad existe el cálculo de tres modelos distintos (*VaderSentiment, SentiStrenght* y *RoBERTa polarity*). Se fija usar el cálculo realizado por *RoBERTa polarity*, pero lo ideal sería que el usuario tenga un selector global que le permita escoger qué modelo utilizar. Este selector, podría ubicarse en la barra de navegación principal superior, como parámetro a controlar en forma genérica de la aplicación.

Por otro lado, existe una rigidez en la Vista buscador, donde los medios disponibles quedan fijos en el código. Lo ideal es tener algún *endpoint* que mediante una consulta simple, retorne los medios disponibles en el sistema. De manera similar, los parámetros para efectuar nuevas agrupaciones en la Vista agrupación, también quedan fijos en el código del *frontend*.

Otro punto débil, es que la aplicación no queda en modo *responsive* y por lo tanto, no sirve para dispositivos móviles. También que el sistema colapsa con algunas búsquedas debido al *overload* generado por la consulta al traer todos los resultados de búsqueda en un paquete.

Dentro de los puntos positivos, la implementación hecha es bastante fiel a los *mockups* realizados en la etapa de diseño. Por otro lado, algunas animaciones hacen que el sistema sea más interactivo, por ejemplo, al pasar el cursor sobre los artículos, este cambia de color indicando que se puede acceder al artículo externamente. Ver la Figura 4.31.

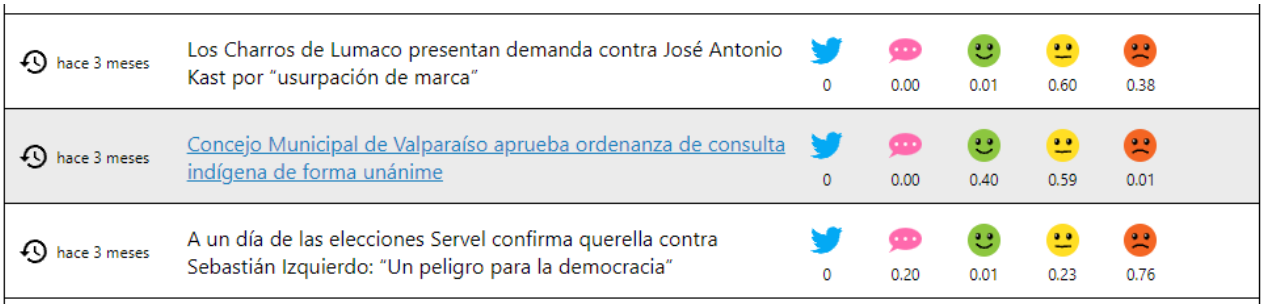

Figura 4.31: Cursor interactivo sobre artículos

4.9. Automatización de procesos

4.9.1. *Control service*

Este servicio se crea a fin de cumplir tres objetivos: (1) preprocesar artículos faltantes, (2) crear títulos y resúmenes faltantes, y (3) actualizar el número de *retweets* y *likes* de cada artículo. Se desarrolla un método para cada uno de estos objetivos, que de manera general, hace consultas a la base datos, procesa datos y luego actualiza documentos.

Para ello, se implementa un método *main*, que idealmente en paralelo, ejecuta cada uno de estos métodos y actualiza la base de datos en forma continua. Sin embargo, con la implementación actual, estos procesos se ejecutan de manera secuencial, porque juntos no caben en memoria GPU ni CPU, junto a la aplicación corriendo en paralelo.

Quizás en un futuro, dada lo específico de las tareas, sea prudente separar cada una en un microservicio independiente.

4.9.2. Otros procesos

Se crea un *script* de Powershell[67] que ejecuta el proceso de Elasticsearch junto a todos los *scrappers*, para que de manera periódica, se extraiga la información de los medios de comunicación. De esta forma, siempre se cuenta en el sistema con información lo más actualizada posible.

Sin embargo, el sistema carece de operaciones de mantenimiento. Tampoco se cuenta con un sistema de *logging* de los procesos, lo cual impide identificar fallas de ocurrir. Para esto Elasticsearch al menos, cuenta con Elastic Observability[68], que permite monitorear la base de datos. Sin embargo, sería ideal crear un sistema de excepciones personalizadas y estandarizadas para otras partes del sistema, así como de mensajes en consola relevantes.

[67]https://docs.microsoft.com/en-us/powershell/
[68]https://www.elastic.co/observability

Capítulo 5

Evaluación y validación

5.1. Evaluación de funcionalidades

Se analiza el comportamiento de las dos funcionalidades principales de la plataforma: buscador y agrupaciones, desde la perspectiva del usuario, pero también examinando los resultados desde un aspecto técnico. Se da importancia a los resultados de subjetividad y polaridad para los resultados, y se evalúa en forma cualitativa la calidad de los resúmenes, titulares y de las agrupaciones de noticias realizadas.

Se hacen dos experimentos para evaluar el uso de esta plataforma. El primero considera la ventana de tiempo entre el 16 de diciembre de 2021 a las 00:00, hasta el 17 de diciembre de 2021 a las 23:59. El segundo considera la ventana de tiempo desde el 23 de febrero del 2022 a las 22:00 horas hasta el 25 de febrero a las 00:00 horas (madrugada). Para el primer caso existen 1.412 artículos procesados mientras que para el segundo son 990.

En la primera ventana de tiempo ocurren tres eventos importantes en el plano nacional: (1) muere Lucía Hiriart, esposa del dictador Augusto Pinochet, (2) terminan las campañas electorales de Gabriel Boric y José Antonio Kast, y (3) la mañana del 17, Plaza Baquedano amanece mitad con flores y mitad con tierra. Y quizás también mencionar que se acaban las entradas de los conciertos de Coldplay y de Dua Lipa.

Por otro lado, la segunda ventana contiene una *breaking news*: la invasión de Rusia a Ucrania. La idea es observar cómo la plataforma se comporta ante ráfagas de artículos sobre un tema.

Lo ideal es probar todas las metodologías de agrupación ante distintos escenarios, pero es suficiente la experimentación a estas dos ventanas de tiempo, ya que se generan múltiples eventos que es posible analizar, demostrando las fortalezas y debilidades del sistema. Por otro lado, para el método de *embeddings* sólo se usa *Agglomerative clustering*, debido a los tiempos de ejecución necesarios para experimentar con *OPTICS*, ya que en estas instancias, se agrupan cerca de 1.000 *sentence embeddings* de 128 dimensiones.

Además, la herramienta de buscador avanzado sólo se evalúa en la primera ventana de tiempo, puesto que es suficiente para mostrar la utilidad de los filtros.

Finalmente, los titulares generados para los eventos mediante el modelo resumidor, se evalúan en dos instancias. Primero se analiza cualitativamente si estos titulares son representativos de los eventos, en los experimentos de las agrupaciones. Y en segundo lugar, se evalúa si estos titulares son más neutrales y menos subjetivos que los titulares de los artículos para cada evento, en un experimento por separado.

5.1.1. Buscador

16 al 17 de diciembre de 2021

Para la primera búsqueda, se coloca en los filtros el rango de fechas correspondiente, y luego en los campos de texto de palabras únicas, se ingresan las palabras *dignidad* y *baquedano* a fin de buscar información respecto al evento que ocurre en Plaza Baquedano (también nombrada Plaza Dignidad o Plaza Italia). Se agrega además, que los artículos no contengan la palabra *Boric* a fin de buscar medios que no hablen de Gabriel Boric, candidato presidencial de la época. Los resultados inmediatos se aprecian en la Figura 5.1 con 18 resultados de seis medios distintos, siendo CHV el medio con más artículos.

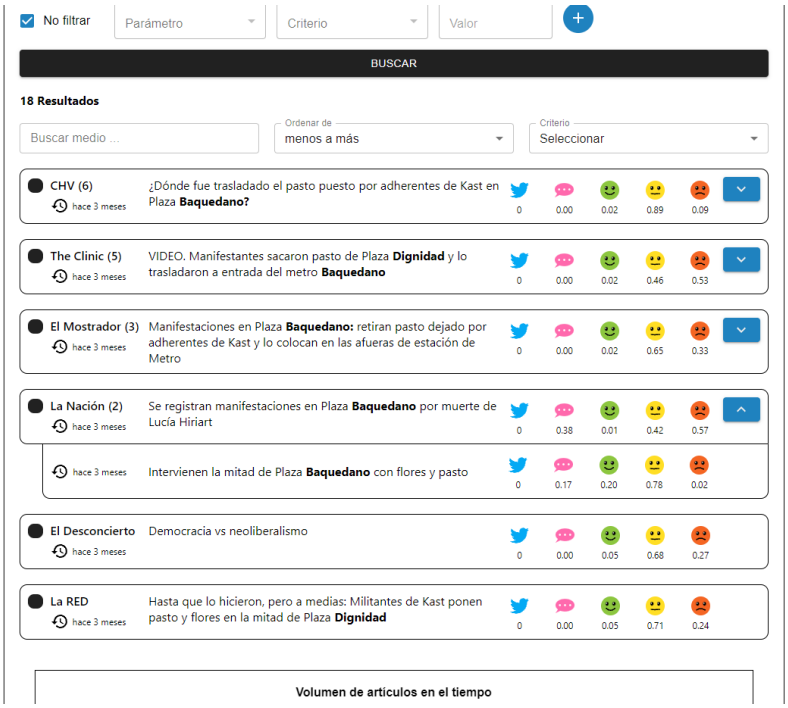

Figura 5.1: Búsqueda de palabras *dignidad* y *baquedano*, pero no *Boric*

Se puede observar que la mayoría de los titulares desplegados son más objetivos que subjetivos, mientras que la mayoría es altamente neutral también. El titular con mayor negatividad es aquel que tiene las palabras *manifestaciones* y *muerte* de parte del medio La Nación. Notar además que tres titulares aluden a adherentes de José Antonio Kast, el otro candidato presidencial de ese entonces.

Para la segunda búsqueda se mantienen todos los parámetros, pero ahora se buscan artículos que no contengan la palabra *Kast* a fin de buscar medios que no hablen de este candidato presidencial. La Figura 5.2 muestra los 12 resultados encontrados, que además ahora son explícitamente ordenados de más a menos neutralidad. A modo general, se aprecia que el sentimiento predominante sigue siendo neutral para este evento. El artículo más positivo es del medio La Nación y contiene las palabras *flores* y *pasto*, pero no palabras que den una negatividad a la noticia. Por otro lado, el más negativo es el mismo que el caso anterior, del medio La Nación y que contiene las palabras *manifestaciones* y *muerte*.

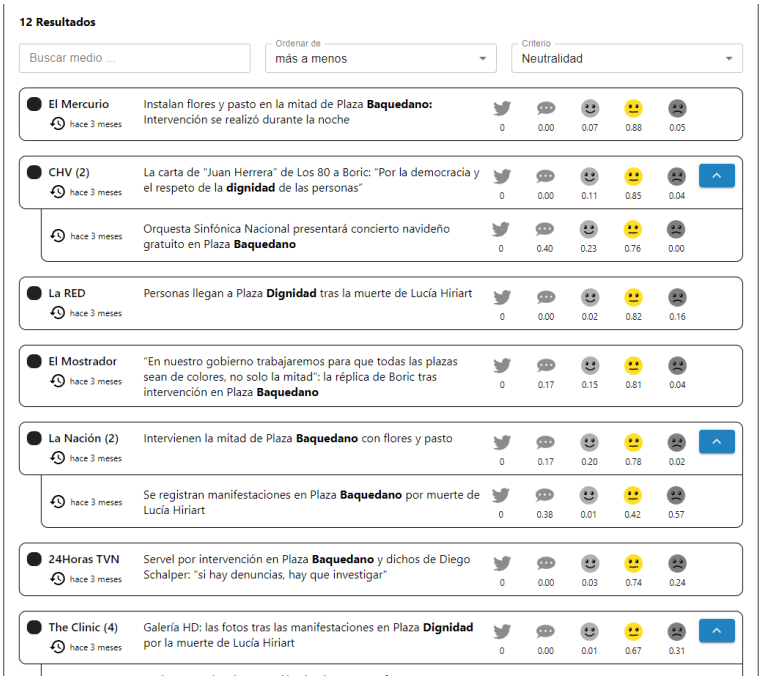

Figura 5.2: Búsqueda de palabras *dignidad* y *baquedano*, pero no *Kast*, ordenando resultados de más a menos neutralidad

Notar que existen dos resultados que no pertenecen a lo que se busca pero que tienen las palabras *dignidad* o *baquedano*. Notar además que hay dos titulares que tienen una cita del candidato presidencial Boric. Por otro lado, se observa un falso positivo de subjetividad para el titular "Orquesta Sinfónica Nacional presentará concierto navideño gratuito en Plaza Baquedano", lo cual es un hecho, pero que se califica con 0.4. Esto demuestra que este modelo no es perfecto.

Para la tercera búsqueda, se especifica el mismo rango de fechas, pero ahora se buscan las palabras *Lucía* y *Hiriart*. En la Figura 5.3 se muestran los 65 resultados ordenados de más a menos negativos. En los primeros resultados destacan palabras como "tiranía", "crímenes", "lesa humanidad", "corrupción", "saqueos", "muerte", "impunidad", "dolor", "división", "críticas", "fallecen", "dictadura", "brutales", etc. Lo cual valida el correcto funcionamiento de este criterio. Destaca el artículo con mayor neutralidad "A los 99 años de edad murió Lucía Hiriart" el cual carece de interpretación y que también está catalogado como objetivo.

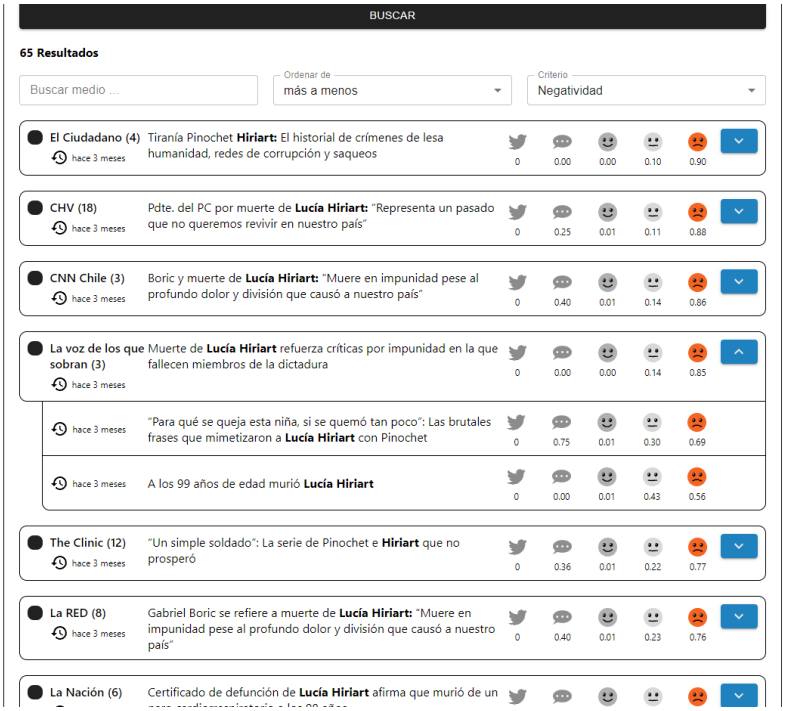

Figura 5.3: Búsqueda de palabras *Lucía* y *Hiriart*, ordenando resultados de más a menos negatividad

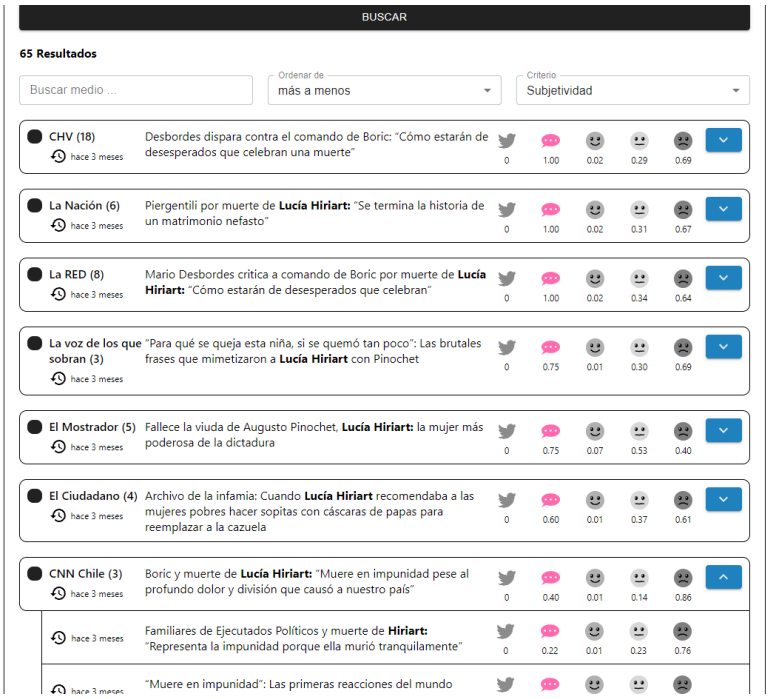

Figura 5.4: Búsqueda de palabras *Lucía* y *Hiriart*, ordenando resultados de más a menos subjetividad

Notar también que los medios El Ciudadano y La voz de los que sobran, son los más agresivos frente a esta noticia. Por otro lado, todos los titulares de la figura que tienen una subjetividad mayor a cero, contienen citas en su contenido, lo cual es correcto, ya que las citas son declaraciones de personas, y, por lo tanto, subjetivas.

La Figura 5.4 muestra los 65 resultados ordenados de más a menos subjetividad. Es notable de inmediato que la mayoría de los titulares contienen citas, o bien, realizan calificaciones a Lucía Hiriart. Tal es el caso del medio El Mostrador, con "Fallece la viuda de Augusto Pinochet, Lucía Hirirart: la mujer más poderosa de la dictadura", y el medio El Ciudadano, con "Archivo de la infamia: Cuando Lucía Hiriart recomendaba a las mujeres pobres hacer sopitas con cáscaras de papas para reemplazar la cazuela". Comportamiento favorable al modelo que evalúa la subjetividad.

La cuarta y última búsqueda contiene los mismos parámetros que la anterior, pero se agrega la palabra *Pinochet* y se añade el filtro para mostrar resultados cuya positividad sea mayor a 0.4. Los resultados se aprecian en la Figura 5.5, que además están ordenados de más a menos positividad. Notar de inmediato que los resultados se reducen a 7, evidenciando que la noticia es más negativa que positiva.

Notar que la mayoría de los titulares contienen declaraciones de familiares de Lucía Hiriart o de Iván Moreira (político). Contienen palabras o frases como "emocionante", "más relevantes", "gran mujer", "gran primera dama", "valorar", "grandiosa obra" o "cariño". Esta búsqueda representa una forma de buscar información sesgada de ciertos grupos de personas, o medios, que piensan distinto a la mayoría, en de una noticia controversial.

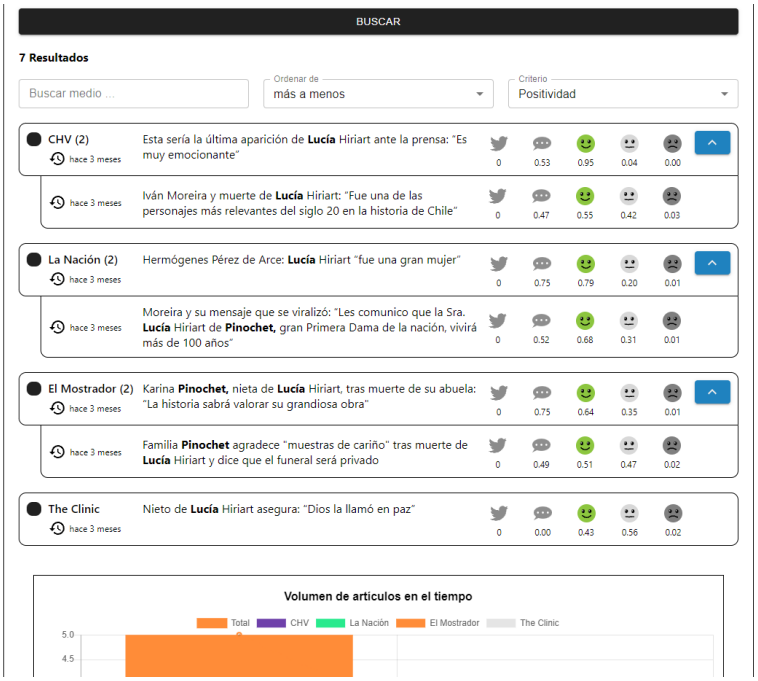

Figura 5.5: Búsqueda de palabras *Pinochet*, *Lucía* y *Hiriart*, cuya positividad sea mayor a 0.4 y ordenando resultados de más a menos positividad

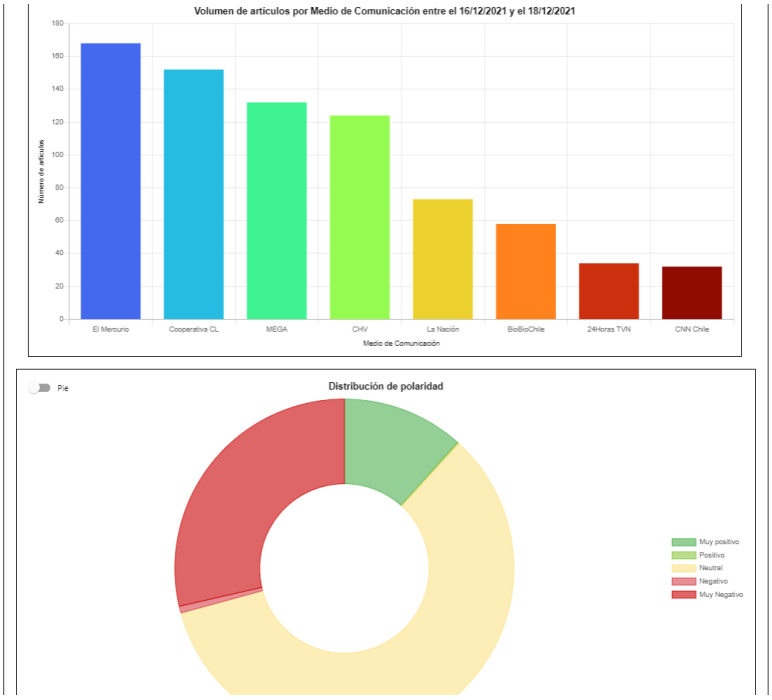

Figura 5.6: Volumen de artículos de medios tradicionales o masivos, y su gráfico de polaridad

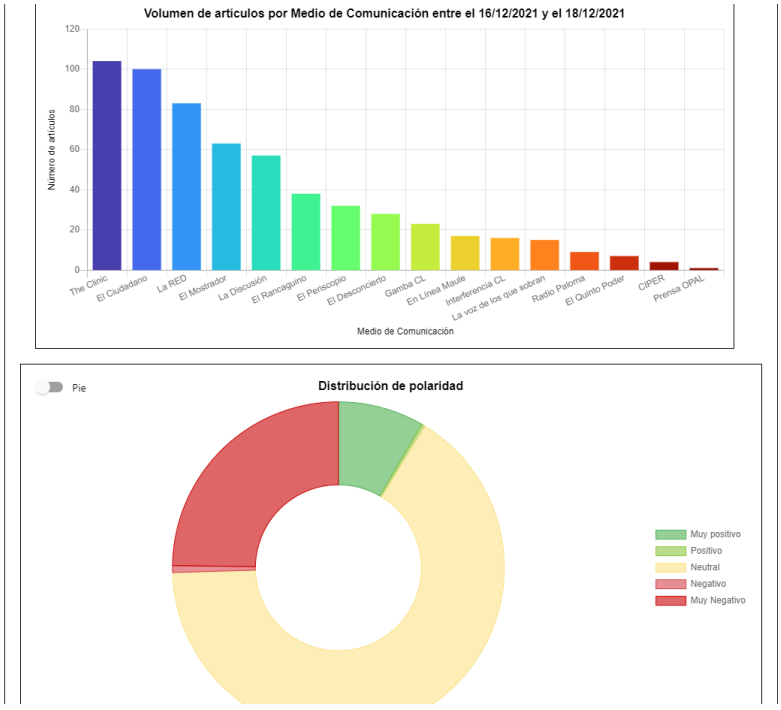

Figura 5.7: Volumen de artículos de medios independientes o alternativos, y su gráfico de polaridad

Ahora se procede a realizar la búsqueda de artículos en el mismo periodo, pero sin filtros de palabras ni de sentimientos, sino que filtrando por medios tradicionales e independientes. La idea es mostrar la utilidad de los gráficos que se encuentran debajo de los resultados de artículos, y sacar conclusiones respecto del comportamiento de estos grupos de medios.

La Figura 5.6 muestra el gráfico volumen de artículos de medios tradicionales o masivos, y también el gráfico de sentimiento de estos medios. Es evidente la capacidad periodística de estos medios, porque cuatro de ellos elaboraron más de 120 artículos en dos días, siendo El Mercurio el medio con más artículos, seguido de Cooperativa CL, MEGA y CHV. Del gráfico de sentimientos, es posible concluir que más de la mitad de los artículos del periodo son neutrales, pero que existen más artículos negativos que positivos.

La Figura 5.7 muestra los mismos gráficos anteriores, pero para 16 medios independientes, alternativos o regionales. Es notorio el cambio en el volumen de artículos por medio, ya que tan sólo cuatro medios elaboraron más de 60 y menos de 110 artículos para este periodo. La mayoría elaboró menos de 40. Por otro lado, el gráfico de sentimientos contiene una distribución bastante similar en comparación al gráfico de los medios tradicionales, lo cual indica que las tendencias de sentimientos se mantienen proporcionales.

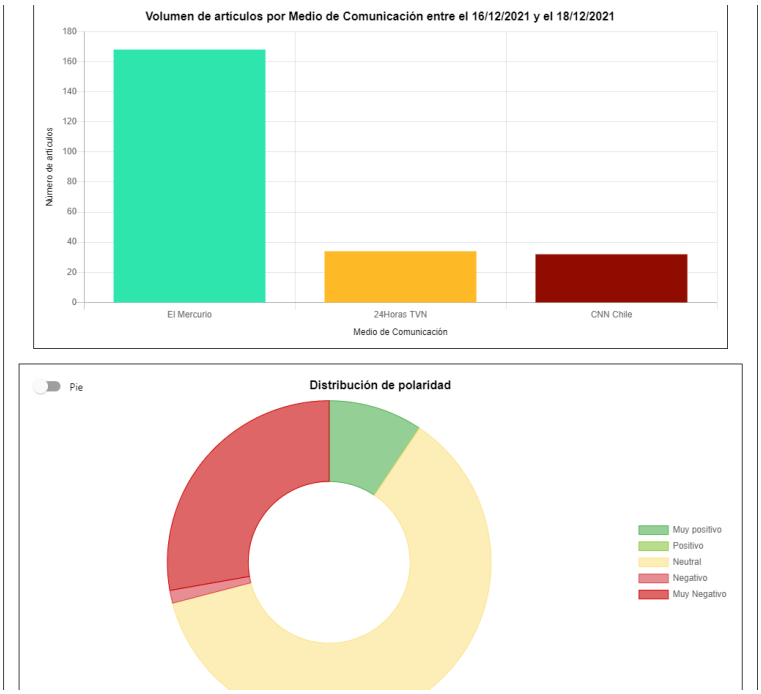

Figura 5.8: Volumen de artículos de medios para algunos tradicionales o masivos, y su gráfico de polaridad

Pero es también interesante analizar el comportamiento de casos aislados. Se seleccionan los medios tradicionales El Mercurio, 24 Horas TVN y CNN Chile, y los medios independientes o alternativos Gamba CL, Interferencia CL, Radio Villa Francia y Prensa OPAL. En las Figuras 5.8 y 5.9 se encuentran los gráficos de volumen y polaridad, respectivamente, para el mismo periodo. Por un lado, los medios tradicionales mantienen la tendencia de sentimientos vista

anteriormente, pero los independientes, marcados casi en totalidad por Gamba CL (dado que ni Interferencia CL, ni Radio Villa Francia elaboraron artículos), muestra que el sentimiento predominante es "muy negativo".

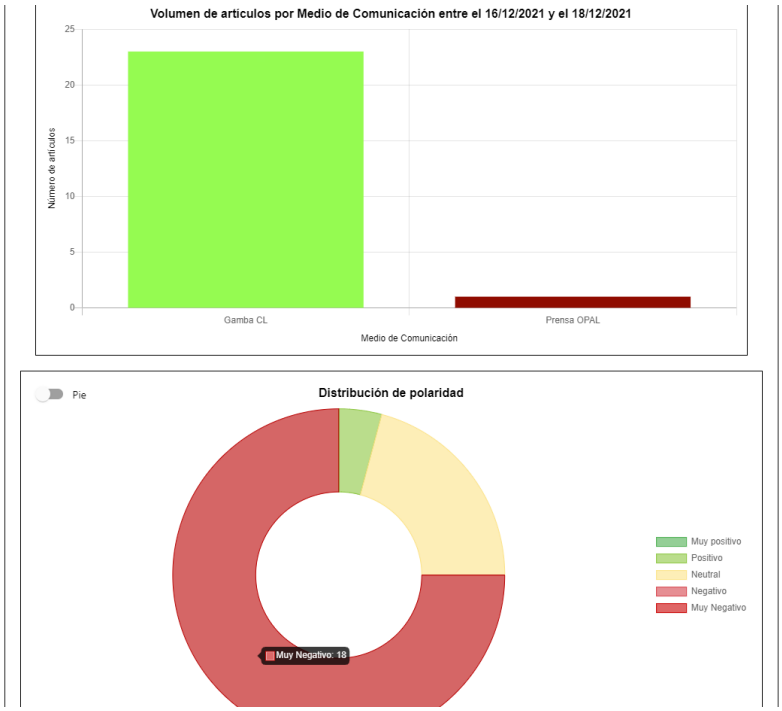

Figura 5.9: Volumen de artículos de medios para algunos independientes o alternativos, y su gráfico de polaridad

Todas estas búsquedas, demuestran el uso práctico de la Vista buscador y cómo los resultados pueden ordenarse bajo el criterio que sea más importante para el usuario. Además, permite constatar cómo medios independientes o alternativos tienden a la subjetividad al colocar citas de personas en sus titulares, o bien al elaborar titulares interpretativos, como en el caso de la búsqueda de las palabras *Lucía* y *Hiriart*.

5.1.2. Agrupador de noticias

16 al 17 de diciembre de 2021

La idea es poder ver los mismos 1.412 resultados encontrados mediante el buscador en el experimento anterior, pero ahora agrupados por eventos. Se utilizan dos metodologías de agrupación: mediante la heurística de palabras, y mediante *embeddings* junto al algoritmo *Agglomerative clustering* y el criterio *Silhouette*. Los tiempos de ejecución de cada cual fueron de 2 y 14 minutos respectivamente. Mencionar que se intenta emplear una agrupación mediante *embeddings* con el algoritmo *KMeans* y el criterio *Silhouette*, pero después de 25 horas de ejecución aún sin resultados, se descarta su utilización por ser claramente poco práctico.

Primero se describen a fondo los resultados para el método de heurística de palabras, evaluando no sólo la agrupación, sino que también los titulares y resúmenes generados. Para la metodología de *embeddings*, sólo se evalúa la agrupación.

Agrupación mediante heurística de palabras

Este método de agrupación detecta 91 eventos entre los 1.412 artículos de este periodo. Los *clusters* (eventos) más grandes son aquellos referentes a la muerte de Lucía Hiriart y al cierre de las campañas electorales. Se procede a describir la calidad de algunos de los 91 *clusters*, evaluando cualitativamente los aciertos y desaciertos de los modelos presentes en el sistema.

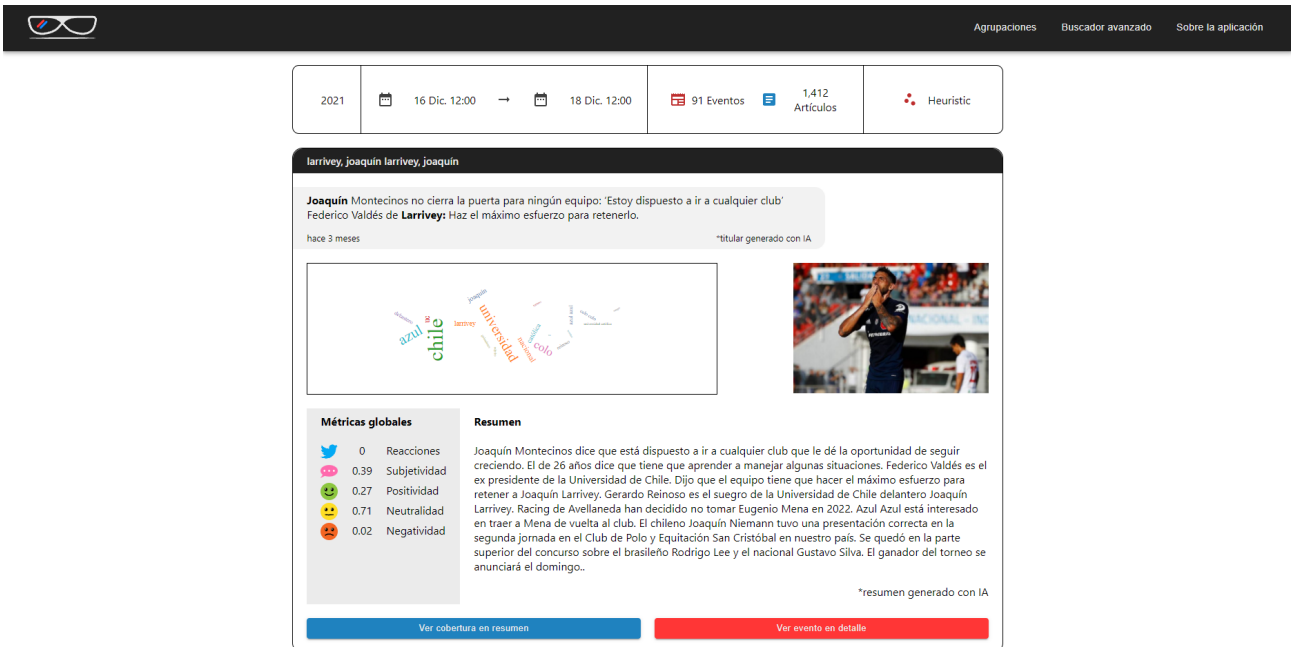

Figura 5.10: Vista del primer evento en la agrupación

En la Figura 5.10 se encuentra una captura del primer evento mostrado para esta agrupación, con las palabras clave "Joaquín", "Larrivey" y "Joaquín Larrivey" (nombre, apellido, y nombre y apellido juntos). Por el titular generado, pareciera tratarse del jugador de fútbol Joaquín Montecinos y su partida a otro club. Pero es un poco extraño, porque se compone de dos oraciones: una señala que "Joaquín Montecinos no se cierra a ninguna alternativa" y la segunda indica las declaraciones de Federico Valdés (presidente de Azul Azul, entidad dueña del club de fútbol Universidad de Chile de la época) sobre Larrivey (jugador de la Universidad de Chile) y la retención de este jugador. En definitiva, cosas distintas.

Si se analizan los artículos que componen el evento (no mostrados en la figura). Es fácil evidenciar que hay noticias distintas mezcladas por un alcance de palabras. Por ejemplo, un titular de La Red menciona "Joaquín Montecinos no le cierra la puerta a ningún equipo: "Estoy dispuesto a ir a cualquier club", mientras que uno de Cooperativa menciona "Joaquín Niemann logró sostenerse en la cima en la segunda jornada en el Club de Polo" y uno de El Periscopio

menciona “Desde la UC sueñan con la llegada de Joaquín Larrivey”. Todos con alguna de las palabras clave del evento.

Al estar mezcladas las tres noticias, el resumen intenta proporcionar información de manera forzosa. Lo que acá es clave, es el proceso de *tokenization* de la heurística, porque si bien, relajar la condición de entidades tipo persona sean *tokenizadas* con su nombre, apellido, y nombre y apellido juntos, para abarcar más titulares, puede provocar casos como este, donde se habla de tres Joaquín distintos: Joaquín Niemman, Joaquín Montecinos y Joaquín Larrivey. Sin embargo, cinco de los siete artículos pertenecientes al evento (no mostrados en la figura) hablan de Joaquín Larrivey y todos tienen la palabra “Larrivey” en el titular. Sólo algunos añaden Joaquín. Por ello, se plantea que en un futuro, para entidades tipo persona, sólo se considere apellido, y nombre y apellido juntos, para evitar este tipo de colisiones por nombre.

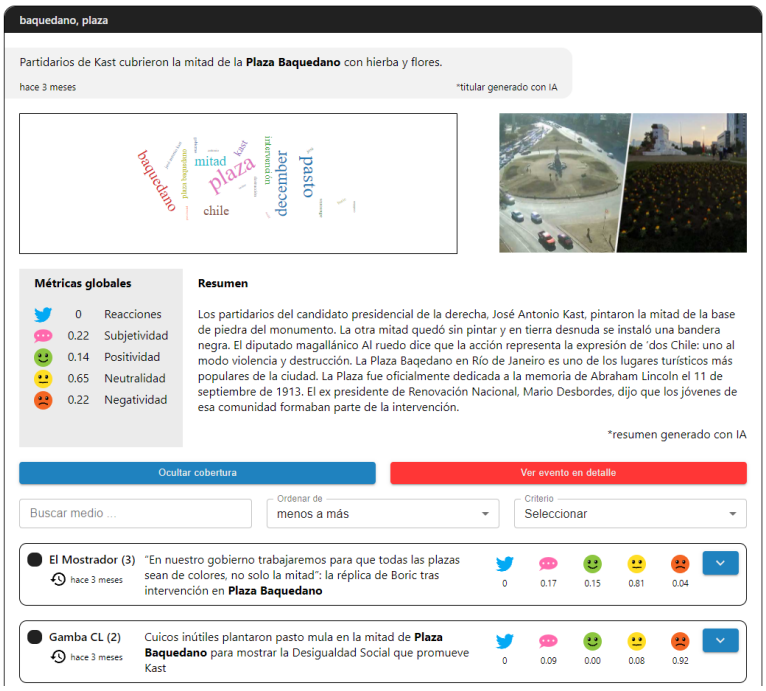

Figura 5.11: Evento en Plaza Baquedano: vista general

En la figura 5.11 se aprecia el evento de palabras clave “baquedano” y “plaza”. Su titular dice “Partidarios de Kast cubrieron la mitad de la Plaza Baquedano con hierba y flores”. La nube de palabras destaca las palabras más usadas en los cuerpos de las noticias. La foto desplegada se extrae en forma aleatoria de un medio y refleja con fidelidad el evento. Las métricas globales sugieren que el evento se aborda de manera neutral.

El resumen contiene aciertos e incoherencias. Las primeras dos oraciones describen muy bien el evento. La tercera habla del diputado magallánico “Al ruedo” y termina hablando de “dos Chile” pero sólo describe uno. El diputado por Magallanes para ese periodo se llama Carlos Bianchi. La siguiente frase habla de “La Plaza Baqedano en Río de Janeiro” lo cual, en primer lugar está mal escrito, y en segundo lugar, es totalmente falso. La siguiente frase sobre Abraham Lincoln es falsa y la última habla de las declaraciones de Mario Desbordes, muy

bien contextualizado como expresidente del partido político Renovación Nacional, pero con la declaración "Los jóvenes de esa comunidad formaban parte de la intervención", pero nunca se especifica qué es "esa comunidad".

Esto refleja que el procedimiento mediante el cual se están elaborando los resúmenes, (traducción de español a inglés, resumen disjunto, traducción al español nuevamente) tiene deficiencias claras. Lo que más llama la atención es la frase que habla de Abraham Lincoln, ya que al examinar cada uno de los cuerpos de las noticias de todos los artículos de este evento, ninguno contiene la palabra "Lincoln", por lo tanto, esta frase es atribuible al *dataset* con el cual fue entrenado el modelo para resumir. Lo mismo sucede con la frase que contiene "Río de Janeiro": ningún artículo contiene estas palabras.

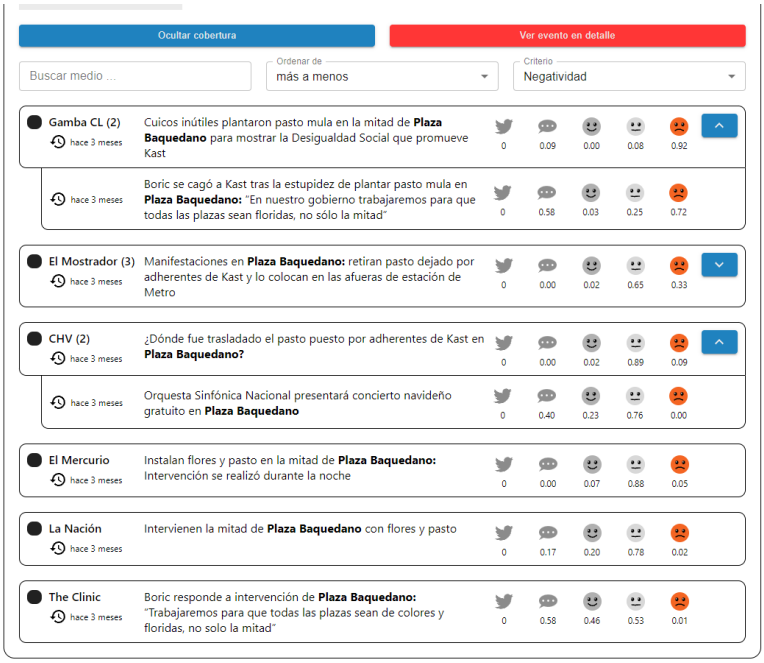

Figura 5.12: Evento en Plaza Baquedano: orden de artículos de más a menos negatividad

En cuanto a los diez artículos de los eventos mostrados en la Figura 5.12, sólo uno no pertenece al evento, proveniente de CHV, y se titula "Orquesta Sinfónica Nacional presentará concierto navideño gratuito en Plaza Baquedano". Por otro lado, en la misma figura es evidente que el medio más negativo es Gamba CL, utilizando palabras como "inútiles" y "estupidez".

Al ordenar los artículos por positividad (Ver Figura 5.13) se encuentra el titular del medio The Clinic: "Boric responde a intervención de Plaza Baquedano: "Trabajaremos para que todas las plazas sean de colores y floridas, no solo la mitad". Las palabras "colores" y "floridas" hacen que el titular sea más positivo y el modelo de polaridad lo detecta bien. Por otro lado, el mismo titular del concierto navideño sale como el segundo más positivo, porque la palabra "navideño" suele asociarse a cosas positivas.

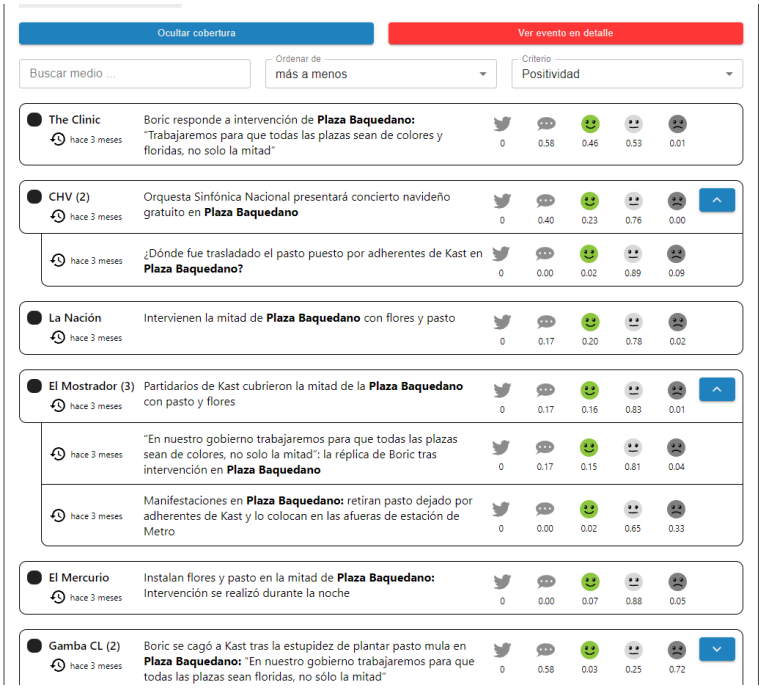

Figura 5.13: Evento en Plaza Baquedano: orden de artículos de más a menos positividad

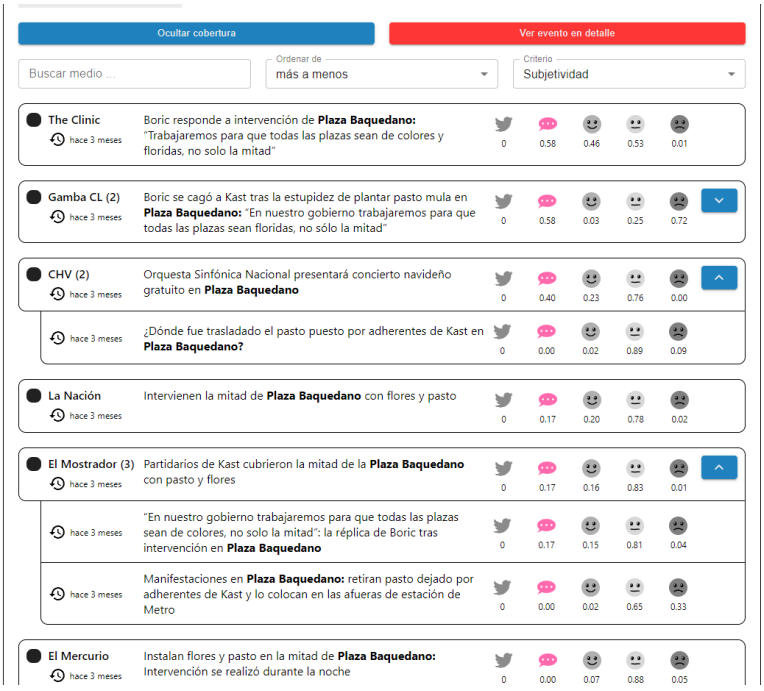

Figura 5.14: Evento en Plaza Baquedano: orden de artículos de más a menos subjetividad

Por último, al ordenar los artículos de más a menos subjetivos (Ver Figura 5.14) nos encontramos con el mismo titular de The Clinic, ya que tiene la cita de una persona, y el segundo

titular más subjetivo tiene la misma cita. Los artículos más objetivos se titulan "Instalan flores y pasto en la mitad de Plaza Baquedano: Intervención se realizó durante la noche" y "Manifestaciones en Plaza Baquedano: retiran pasto dejado por adherentes de Kast y lo colocan en las afueras de estación de Metro", lo cual parece razonablemente neutro.

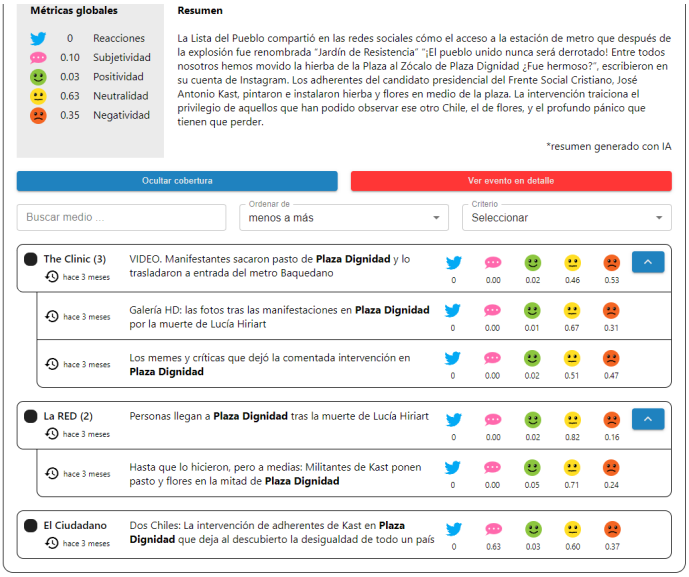

Figura 5.15: Evento en Plaza Baquedano: mismo evento pero refiriéndose a *Plaza dignidad*

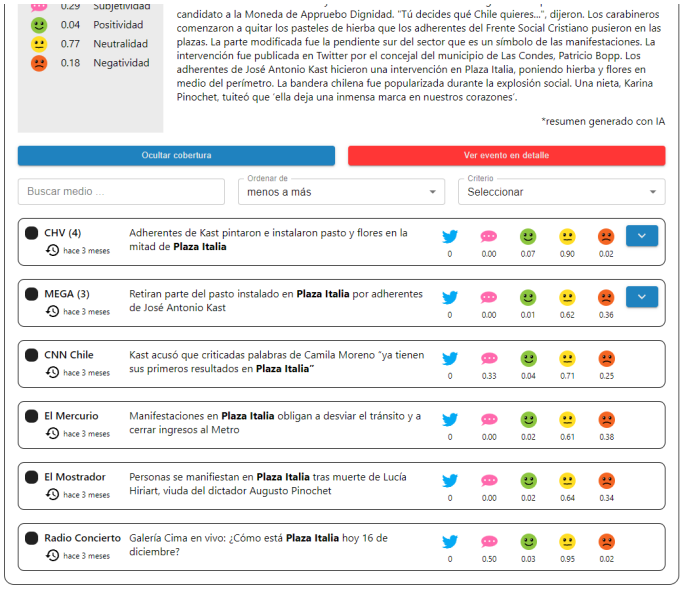

Figura 5.16: Evento en Plaza Baquedano: mismo evento pero refiriéndose a *Plaza italia*

Una observación bastante interesante, es que existen dos *clusters* con artículos disjuntos que se refieren a este evento (Ver Figuras 5.15 y 5.16). El primero tiene las palabras clave "plaza dignidad", "plaza" y "dignidad" y el segundo las palabras "plaza italia", "italia" y "plaza". Es imposible que el algoritmo de heurística de palabras sepa que "plaza baquedano", "plaza

italia" y "plaza dignidad" corresponden a la misma entidad, porque podrían ser perfectamente tres plazas distintas. Esta situación plantea el desafío de poder introducir algún mecanismo de clases de equivalencia entre entidades, para que el algoritmo en algún punto, junte a estos *clusters* que hablan de la misma entidad, pero que se conoce por varios nombres.

Se procede a analizar otro evento: la muerte de Lucía Hiriart. En la Figura 5.17 se muestra el *cluster* más grande reconocido por la heurística para esta noticia. Contiene múltiples palabras clave, la nube de palabras destaca "Lucía", "Hiriart", "Pinochet" y "Chile" y la imagen seleccionada muestra a esta persona de negro. Las métricas globales sugieren que es un evento más negativo que positivo, pero al mismo tiempo abordado mayoritariamente en forma neutral.

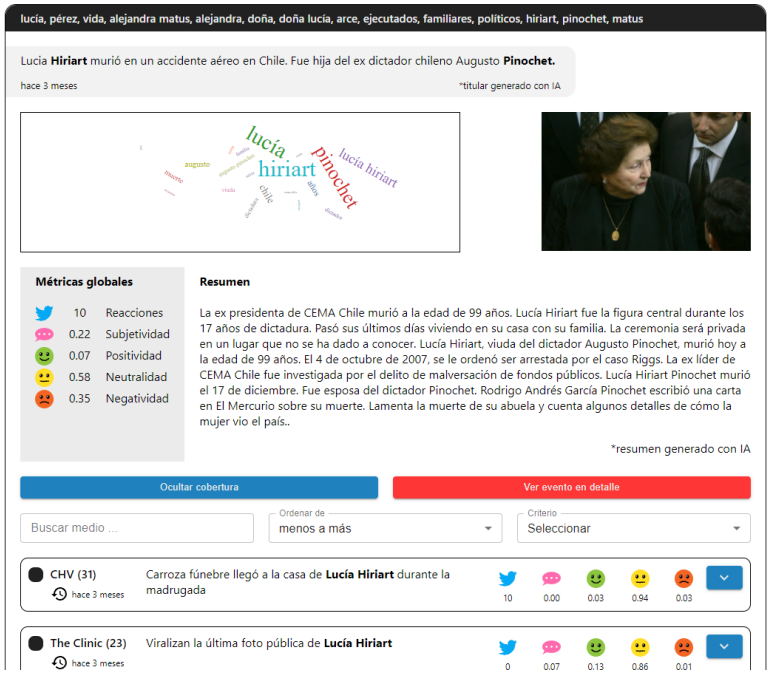

Figura 5.17: Evento de Lucía Hiriart: vista general

En este caso, el resumen generado es totalmente preciso, coherente y no redundante (salvo mencionar que fue expresidenta de la CEMA dos veces). Evidencia un muy buen caso donde el procedimiento implementado para resumir, pese a todas sus falencias, funciona y funciona bien. Sin embargo, acá el título es el que difunde información falsa, ya que Lucía Hiriart no muere en un accidente aéreo en Chile (su causa es desconocida). Se desconoce la causa que fuerza al modelo a generat esta información, debido a que ningún titular contiene la palabra "accidente". Una hipótesis, es que el modelo fue entrenado de forma tal que la palabra "muerte" se asocia más a "aéreo".

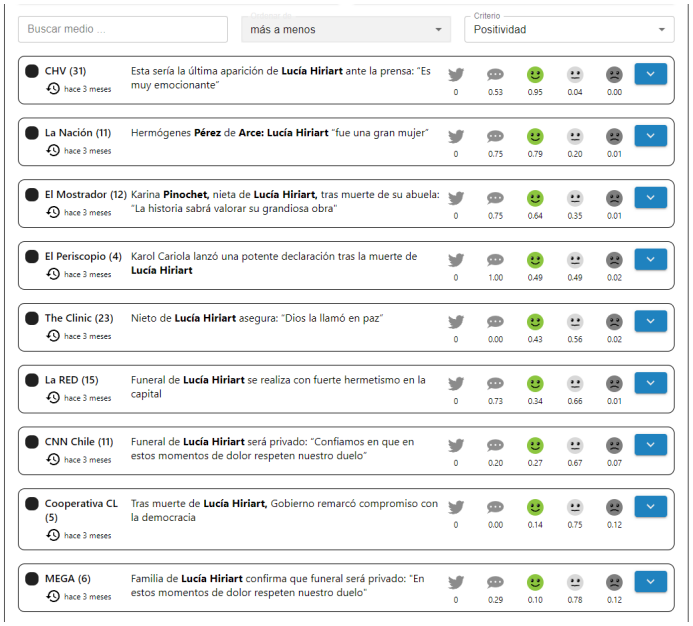

Figura 5.18: Evento de Lucía Hiriart: orden de artículos de más a menos positividad

La figura 5.18 muestra los artículos más positivos de este evento. En general, son los mismos artículos que aparecen anteriormente en el buscador: pertenecientes a familiares o admiradores. Por otro lado, en la Figura 5.19 se aprecian los artículos más negativos. Al igual que el caso anterior, son los mismos artículos que se encuentran en el buscador, y que usualmente contienen palabras de carácter negativo.

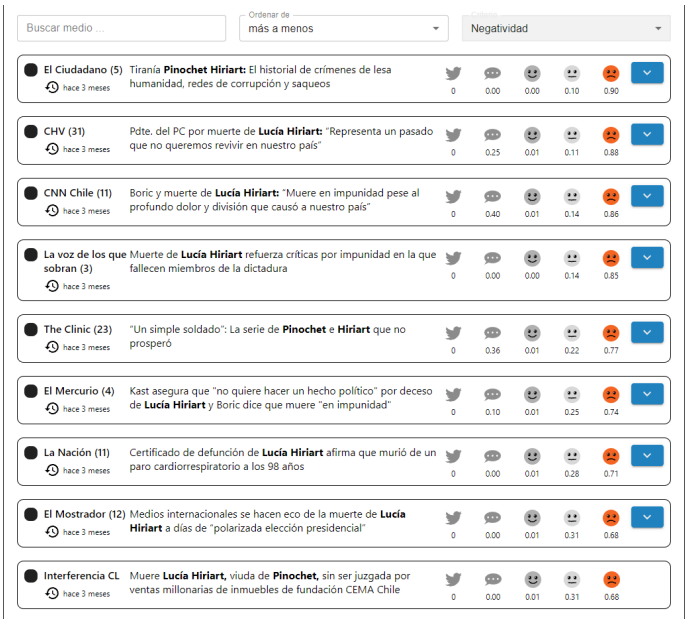

Figura 5.19: Evento de Lucía Hiriart: orden de artículos de más a menos negatividad

Por último, mencionar, que se encuentra otro *cluster* disjunto referente a esta noticia, pero con las palabras clave "Mónica González", "González" y "Mónica" (Ver Figura 5.20). Los artículos que la componen son declaraciones de esta persona respecto de la muerte de Lucía Hiriart. La nube de palabras del evento evidencia que el tema del evento gira en torno a Lucía Hiriart. La foto no es cargada (por algún error) y tanto el titular como el resumen son deficientes.

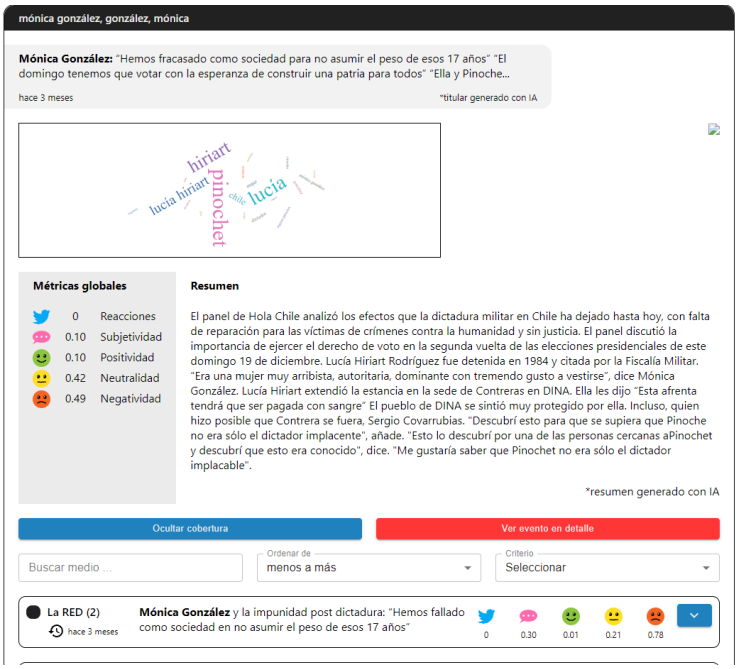

Figura 5.20: Evento de Lucía Hiriart: evento secundario

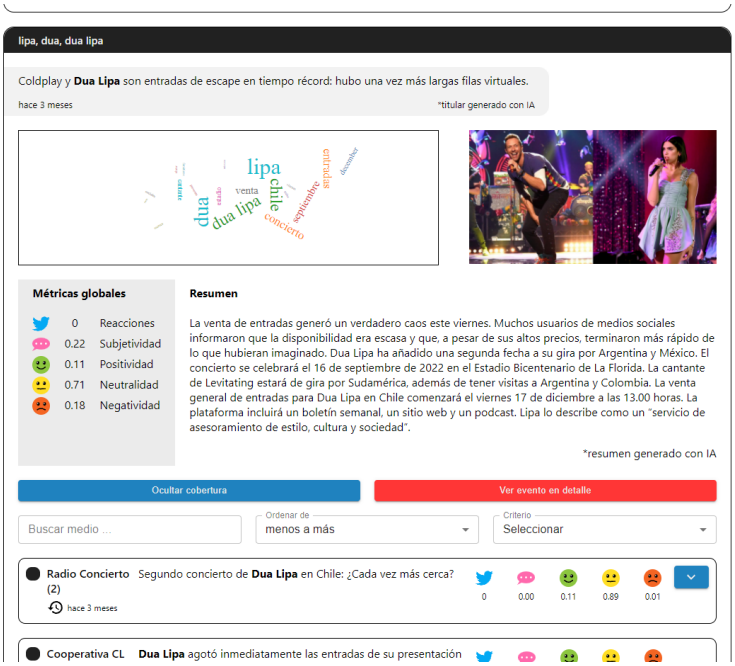

Figura 5.21: Evento de Coldplay y Dua Lipa: vista general

En esta agrupación se encuentra otro evento: el agotamiento de entradas para los conciertos de Coldplay y Dua Lipa. El evento encontrado por la heurística se aprecia en la Figura 5.21. Tanto la nube de palabras como la imagen permiten identificar fácilmente que el evento se trata de estos dos artistas. Sin embargo, las palabras clave del evento son "Lipa", "Dua" y "Dua Lipa", lo cual hace pensar que este evento está centrado en esta artista. Esto lo respalda el hecho de que cuatro de los cinco artículos del evento, no mostrados en la figura, hablan únicamente de Dua Lipa. Solamente el artículo restante habla de ambos artistas.

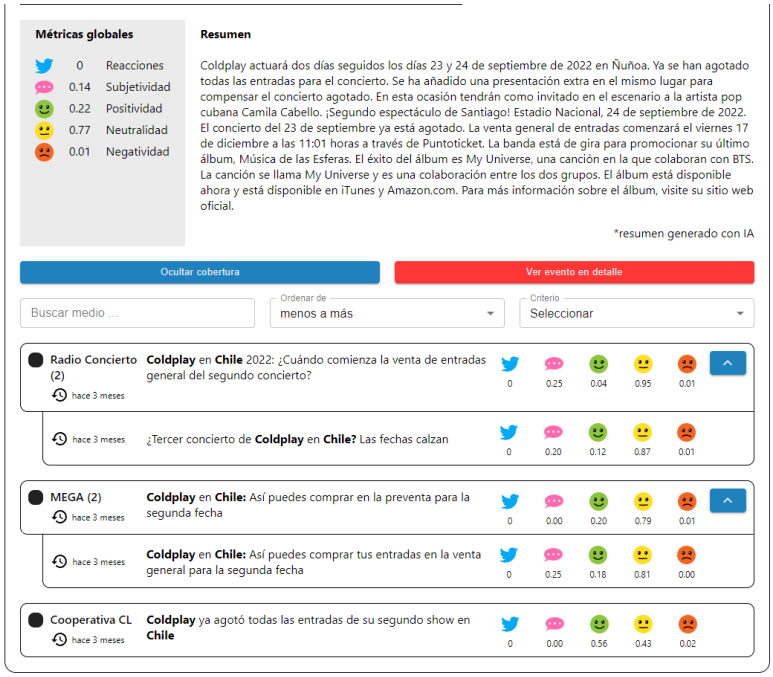

Figura 5.22: Evento de Coldplay y Dua Lipa: evento secundario

Por su parte, el titular es un poco incoherente "Coldplay y Dua Lipa son entradas de escape en tiempo récord: hubo una vez más largas filas virtuales". Por otro lado, el resumen es decente, pero las últimas dos oraciones parecen hablar de una plataforma o servicio de la cual no se brinda contexto: esto debe ser consecuencia del problema conocido como *context fragmentation* que se menciona en la Sección 4.4.5.

Por último, señalar que hay otro *cluster* con las palabras clave "Chile" y "Coldplay" con cinco artículos hablando sobre este artista únicamente (Ver Figura 5.22). De todo esto es posible concluir que la heurística es capaz de separar las noticias de ambos artistas en *clusters* distintos, pero la generación del primer titular donde se menciona a ambos artistas confunde al usuario al hacer creer que un *cluster* contendrá información de ambos artistas, cuando no es el caso. Acá el fallo mayor es sin duda la generación del titular.

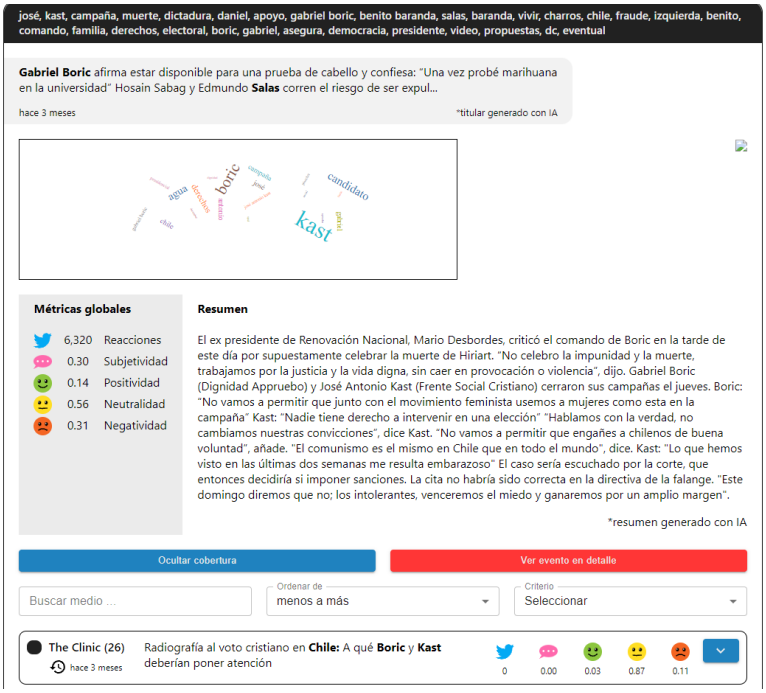

Figura 5.23: Evento de Cierre de campaña presidencial: vista general

Otro evento con amplia cobertura es el cierre de la campaña presidencial. La heurística de palabras logra identificar este evento (Ver Figura 5.23) con múltiples palabras claves, que incluyen el nombre de ambos candidatos, junto a palabras como “democracia”, “electoral”, “presidente”, etc. Notar que en las métricas globales las reacciones son 6.320 porque hay un artículo que fue vinculado a un *tweet*, en este caso, uno del medio CHV[1]. Esto demuestra que esta parte del *software* está operativa, pero que no logra capturar los *tweets* de la mayoría de los artículos.

Analizando en profundidad, el titular contiene al menos dos eventos distintos y por otro lado, el resumen contiene casi en su totalidad citas de frases dichas por ambos candidatos. Esto sucede porque el evento tiene 159 artículos (no visibles en la figura) y que contienen sub-eventos. Por ejemplo, algunos hablan de las reacciones de los candidatos respecto de la muerte de Lucía Hiriart. Otros del cierre de actividades de ambos comandos en Chillán. Otros del rechazo de Kast por los derechos LGBTIQ+. Y así también declaraciones, entrevistas cortas, etc.

Seguramente, este *cluster* es tan grande debido a la modificación que se hizo de la heurística para juntar a *clusters* similares (Ver Algoritmo 3). Es difícil establecer si este *cluster* es correcto o incorrecto, porque es cierto que todos los artículos que lo componen hablan de ambos candidatos presidenciales, pero al mismo tiempo, hay muchos sub-eventos en el *cluster*. Quizás para un futuro se podría experimentar ejecutando la heurística con los artículos de este *cluster* únicamente, para nuevamente separar los artículos en *clusters* de sub-temas.

A pesar de esto, es positivo que todos los artículos de ambos candidatos (o la mayoría de estos) queden juntos en un solo *cluster*, pues de esta forma se cumple el objetivo de evitar la sobrecarga informativa, y permite, por ejemplo, navegar por noticias que no sean referentes a

[1]https://twitter.com/CHVNoticias/status/1471327217191899141

este evento. Se plantea la hipótesis de que este mismo comportamiento sucede con *breaking news*, donde se publica una ráfaga de artículos sobre un tema, que, al mismo tiempo, opaca noticias con menos cobertura.

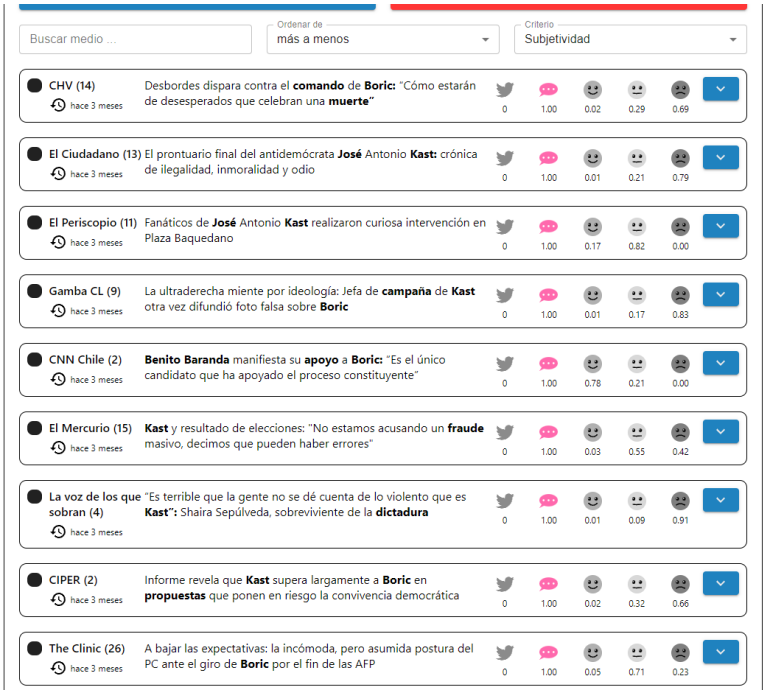

Figura 5.24: Evento de Cierre de campaña presidencial: orden de artículos de más a menos subjetividad

Si se ordenan los artículos de más a menos subjetividad (Ver Figura 5.24) es evidente que los titulares más subjetivos son cuñas, o bien titulares interpretados como "La ultraderecha miente por ideología: Jefa de campaña de Kast otra vez difundió foto falsa sobre Boric", o "El prontuario final del antidemócrata José Antonio Kast: crónica de ilegalidad, inmoralidad y odio", por parte de los medios independientes Gamba CL y El Ciudadano.

Por último, mencionar que existe otro *cluster* disjunto referente al cierre de campañas con las palabras clave "elecciones" y "presidenciales" con cuatro artículos (Ver Figura 5.25). Todos hablan en torno a las elecciones, pero de temas distintos, uno sobre transporte, otro sobre el Partido de la Gente (del ex-candidato presidencial Franco Parisi de la época), otro sobre los mercados, y el último sobre la "fractura en Chile". Por esta misma razón, el resumen intenta una vez más incorporar toda esta información de forma forzosa, provocando oraciones fuera de contexto. Por su parte, el titular no es más que un parafraseo del titular de un medio localista.

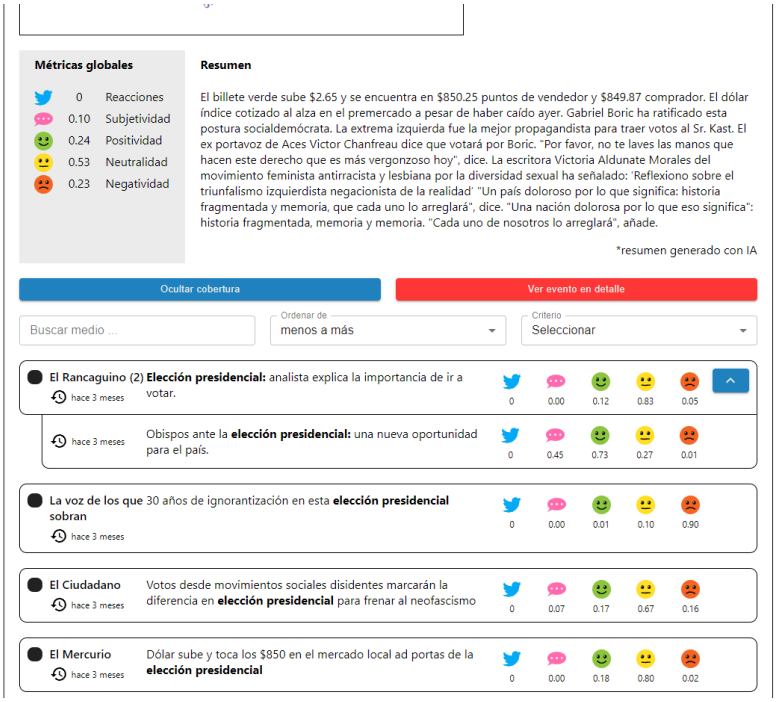

Figura 5.25: Evento de Cierre de campaña presidencial: evento secundario

Nuevamente, es difícil clasificar de correcta o incorrecta esta agrupación, ya que todos los artículos hablan del mismo tema, pero con distintos matices. Todo depende entonces del grado de fineza con el cual se quieran agrupar los artículos y es un desafío a afrontar en un futuro.

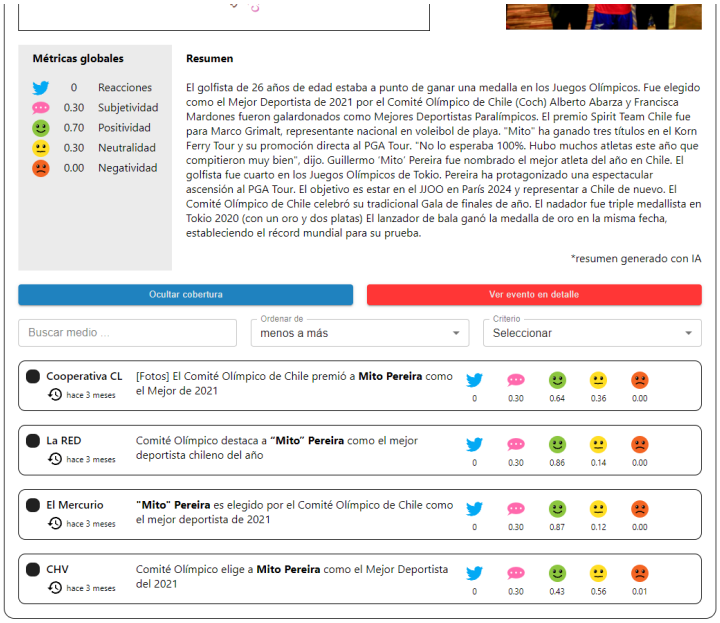

Figura 5.26: Evento de deportista Mito Pereira: vista general

Finalmente, en la Figura 5.26 expone un caso exitoso de agrupación de eventos (un evento sobre el golfista Mito Pereira). En la Figura 5.27 otro caso exitoso (sobre la muerte de un malabarista en Panguipulli), y finalmente, en la Figura 5.28 una última buena agrupación (sobre la inmigración en Colchane).

Figura 5.27: Evento de malabarista en Panguipulli

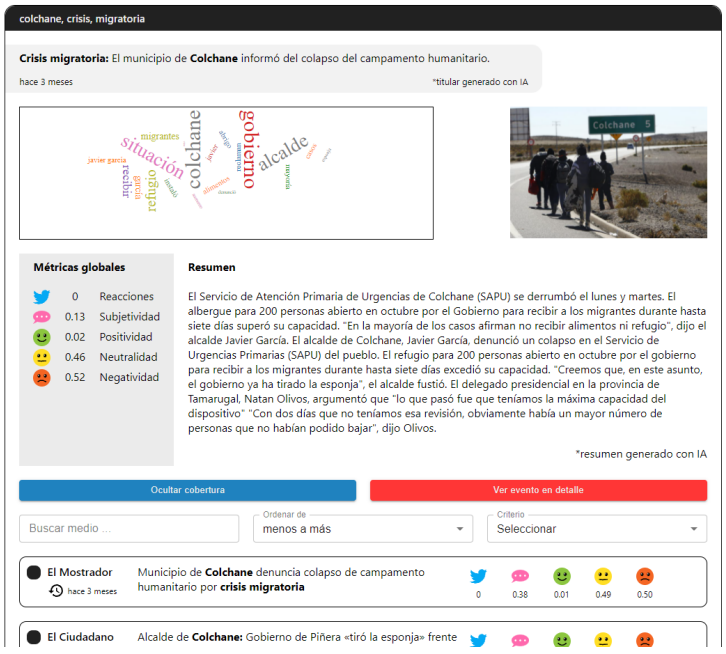

Figura 5.28: Evento de crisis migratoria en Colchane

En contraste a esto, las Figuras 5.29 y 5.30 ilustran eventos defectuosos, ya que nuevamente, por un alcance de nombre, se juntan artículos sobre “Juan Herrera”, “Juan Andrés Fontaine”, “Hugo herrera”, “Juan Abarca” y “San Juan Bautista”. Otro ejemplo de evento defectuoso se encuentra en la Figura 5.31, los “artículos” del medio La Red, son en realidad enunciados de la programación televisiva. De hecho, el resumen refleja que estos “artículos” no tienen cuerpo, ya que si uno entra a ver el contenido de estos “artículos”, se encuentra con videos. La pregunta es entonces ¿Deben estar los programas tipo video en el sistema?.

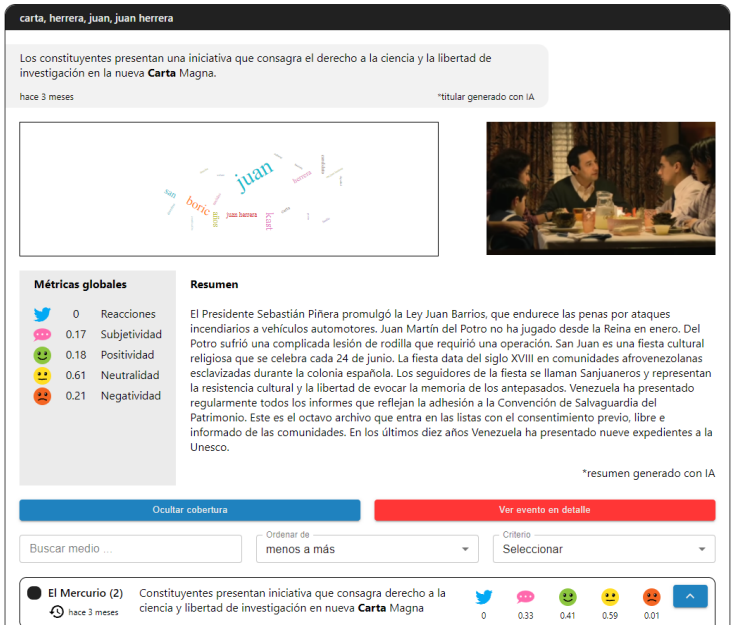

Figura 5.29: Evento de Carta de Juan Herrera: vista general

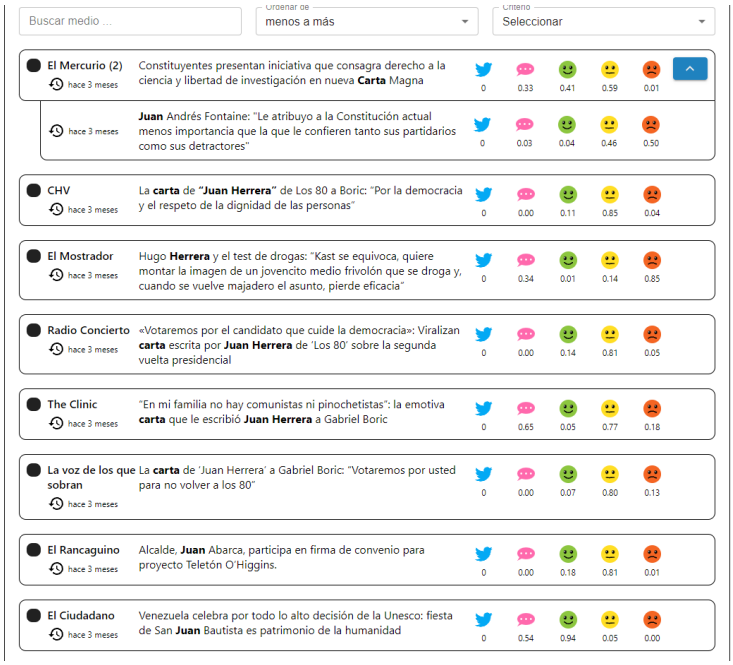

Figura 5.30: Evento de Carta de Juan Herrera: artículos

Figura 5.31: Evento de programas televisivos de La Red

Agrupación mediante *embeddings*, *Aglomerative clustering* y *Silhouette score*

Este método encuentra 195 *clusters* (eventos) sobre los 1.412 artículos del periodo.

Son 26 los eventos, como el expuesto en la Figura 5.32, que hablan de la muerte de Lucía Hiriart y de las reacciones generadas por esto. Al hacer un análisis de primera impresión, es posible notar que algunos de estos *clusters* hablan del funeral, otros de la familia de Lucía, otros de declaraciones de políticos al respecto, otros sobre las declaraciones del gobierno al respecto, etc. Es decir, todos estos *clusters* podrían interpretarse como sub-eventos del evento principal que es la muerte de esta persona.

En contraste con el caso de la heurística, acá si se encuentran estos sub-temas, pero el sistema sufre de lo que busca evitar: La Redundancia de la información. 26 *clusters* es una cifra alta, en comparación a 2 como el caso anterior. Sin embargo, es observable que las palabras claves de estos 26 *clusters* (que son los términos más utilizados en los cuerpos de los artículos en este caso) coinciden en tener a "Lucía", "Pinochet", "muerte" y "Hiriart". Por lo tanto, es posible aplicar el mismo Algoritmo 3 hecho para la heurística, pero en este caso como capa final de los resultados de agrupación por *embeddings*, para justamente agrupar *clusters* que son altamente similares bajo el criterio de palabras clave.

Por otro lado, el *cluster* fallido del caso anterior sobre Juan Herrera (Ver Figura 5.29) ahora si se agrupa bien, con cuatro de cinco artículos correctamente agrupados. La Figura 5.33 muestra la agrupación correcta hecha mediante este método de *clustering*.

Es posible observar, de manera general, que eventos más pequeños (con 3 o 4 artículos) y específicos son correctamente agrupados. Por ejemplo, algunos referentes al *show* de año nuevo de la Torre Entel, o del convenio entre municipalidades y la empresa Gasco para comprar gas más barato, o del transporte en la Región Metropolitana el día de las elecciones, etc.

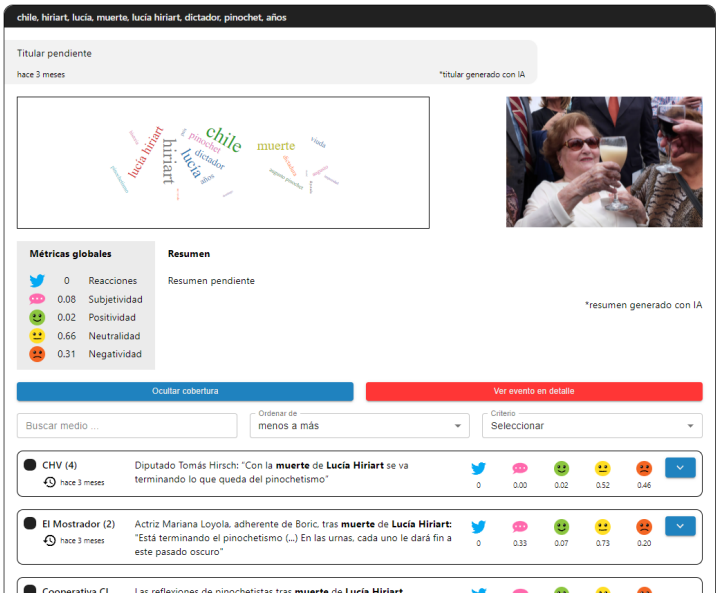

Figura 5.32: Evento de ejemplo sobre Lucía Hirirart, encontrado por *Agglomerative clustering*.

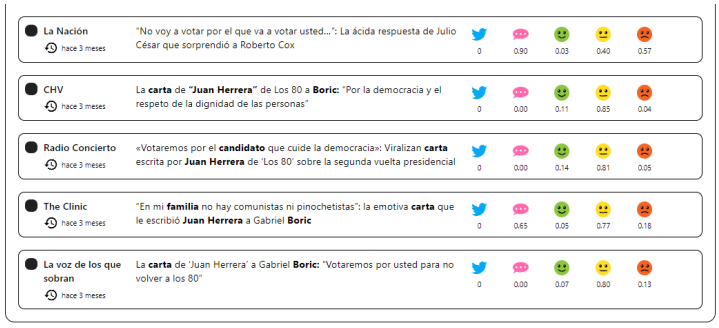

Figura 5.33: Evento de Carta de Juan Herrera: agrupación correcta

En el caso del cierre de las campañas presidenciales, existen más de 40 *clusters* al respecto. Al igual que con Lucía Hiriart, los *clusters* son sub-eventos del cierre de campaña. El algoritmo diferencia el evento de Sebastián Izquierdo llamando a sabotear las elecciones, del evento que compara la campaña de Kast con la del brasileño Jair Bolsonaro, del evento sobre el llamado de Boric para reunir apoderados de mesa para las elecciones, del evento de cierre de campaña de Boric en Parque Almagro, de la demanda del grupo Charros de Lumaco hacia José Antonio Kast, etc.

Acá nuevamente se encuentra un escenario donde es difícil distinguir si estas agrupaciones son correctas o no, porque ahora la información sobre los candidatos presidenciales está mucho más dispersa, pero más específica, en contraste al caso de la heurística. Quizás una buena alternativa, es utilizar la heurística de palabras para realizar un agrupamiento más general, y este algoritmo mediante *embeddings* para sub-agrupar eventos que contemplen más de 20 artículos, por ejemplo.

Destacar, que los eventos de Coldplay y Dua Lipa son agrupados correctamente en este caso, y muy bien diferenciados. (Ver Figuras 5.34 y 5.35) Por otro lado, hay cuatro eventos sobre el coronavirus, tres que son un poco dispersos (se componen de más de un tema en particular) pero un cuarto con eje en la variante ómicron.

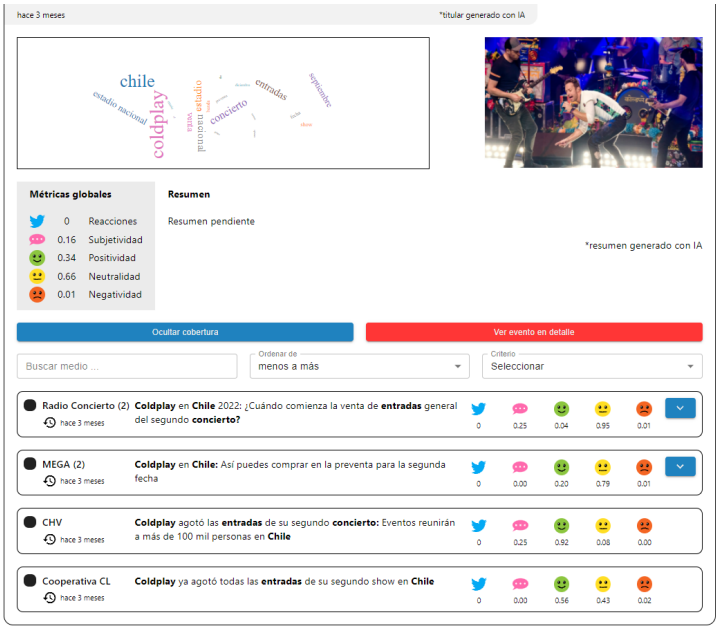

Figura 5.34: Evento de Coldplay: bien agrupado

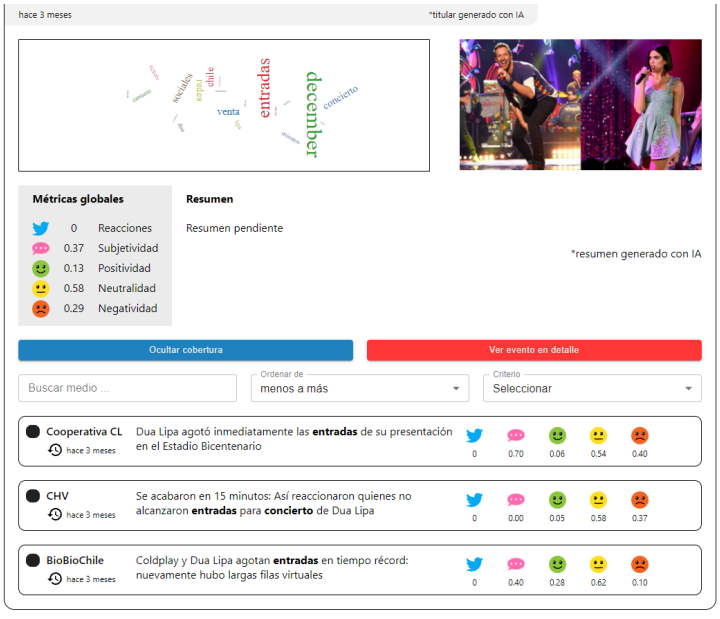

Figura 5.35: Evento de Dua Lipa: bien agrupado

En cuanto al evento de Plaza Baquedano (o *Italia*, o *Dignidad*) hay un evento sobre Plaza Italia y dos sobre Plaza Baquedano. El *cluster* sobre Plaza Dignidad encontrado con la heurística, acá se pierde en forma de ruido en otros *clusters*.

Existe un *cluster* relacionado a la presencia militar rusa en sus fronteras con Ucrania, y cómo la OTAN advierte de consecuencias hacia Rusia de invadir Ucrania. Este *cluster* en particular está muy bien capturado por el algoritmo, a pesar de que los titulares son totalmente distintos los unos del otro: (1) "La OTAN advirtió a Rusia de "consecuencias masivas" si ataca a Ucrania". (2) "Continúa la tensión: Senadores de EEUU proponen ayuda militar de unos $450 millones a Ucrania" y (3) Rusia propone tratados para limitar presencia militar occidental en sus fronteras. Esta agrupación evidencia la utilidad de los *embeddings* ajustados para la tarea STS, debido a que si bien, los titulares no tienen tantas palabras en común, su semántica es bastante similar.

Como última observación, existen muchos *clusters* pequeños que tienen ruido. Esto se debe a la naturaleza de *Agglomerative clustering*, que balancea la calidad de *clusters* grandes, al coste de generar ruido en *clusters* pequeños. Por otro lado, con esta metodología basada en *embeddings*, no se tiene la garantía de que todos los artículos de un evento contienen al menos dos o tres palabras en común, como si lo es con la heurística. Esto representa tanto una ventaja, como el caso del *cluster* sobre Ucrania, como una desventaja, al provocar más ruido. Además notar que, este algoritmo es incapaz de detectar ruido *per se*, como lo hace DBSCAN u OPTICS.

23 al 25 de febrero de 2022

Esta agrupación considera 990 artículos, y la idea es ver mediante el método de agrupación de heurística, si el evento del conflicto ruso-ucraniano queda en uno, o múltiples *clusters*. La ventana de tiempo es desde el 23 de febrero del 2022 a las 22:00 horas, hasta el 25 de febrero a las 00:00 horas (madrugada). La ejecución demoró 9 min. 10 seg. y se encontraron 55 eventos.

En la Figura 5.36 se muestra el evento más grande de la agrupación, con 264 artículos, referente al conflicto ruso-ucraniano. En Anexo D se adjuntan otros 4 *clusters* (eventos) referentes a este conflicto con más artículos. Además, existen además otros siete *clusters* con 3 o 4 artículos cada uno que no se adjuntan. Después del *cluster* más grande, le siguen otros con 51, 16 y dos con 13 artículos. En total, los doce *clusters* suman 376 artículos, que representan el 38 % de total de artículos, y reafirma al evento como *breaking news*.

Lo destacable de esta agrupación es justamente el *cluster* más grande, ya que la heurística fue capaz de capturar la mayoría de los artículos de este evento en uno solo. Si bien al interior de este *cluster* hay muchos matices, como en los ejemplos anteriores, esta agrupación logra separar esta *breaking news* de los demás eventos y mostrar la información de forma ordenada.

Para juntar los 11 *clusters* restantes a este gran *cluster*, se podría aplicar algo similar al Algoritmo 3, pero basado en el *top* 5 o *top* 10 de las palabras más repetidas en los cuerpos de las noticias. Porque todos los eventos contienen las palabras "Ucrania" y "Rusia" en sus nubes de palabras. Incluso podrían aplicarse técnicas más avanzadas de *topic modeling*[2] para encontrar conceptos claves en los cuerpos de las noticias, y aplicar heurísticas sobre estos términos.

[2]http://www.morethanbooks.eu/topic-modeling-introduccion/

Figura 5.36: Evento sobre conflicto ruso-ucraniano #1: 264 artículos

Una vez más, como existen tantos matices en un evento tan grande y de tanto impacto, lo idea sería agrupar los 376 artículos en sub-eventos. Por ejemplo, un sub-evento puede referirse a un ataque en específico a una ciudad, o las declaraciones del presidente Zelenski de Ucrania, o a las declaraciones del presidente Biden de Estados Unidos, o a las declaraciones de la OTAN, etc.

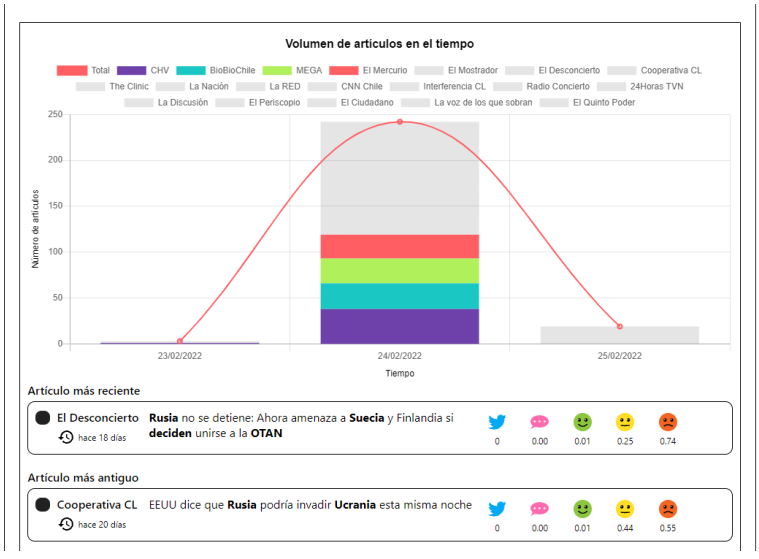

Figura 5.37: Gráfico de volumen de artículos para *cluster* más grande sobre el conflicto ruso-ucraniano

En la Figura 5.37 se muestra el volumen de artículos del *cluster* con 264 artículos, junto al primer y último evento para este rango de fechas. Es interesante, por ejemplo, ver que el primer medio en informar de esta situación fue Cooperativa CL con el titular premonitorio "EEUU dice que Rusia podría invadir Ucrania esta misma noche" publicado a las 22:38 la noche del 23.

De la Figura 5.38 se puede ver que los medios con más artículos respecto del tema fueron CHV, junto a BioBioChile, MEGA, El Mercurio, El Mostrador y el Desconcierto. Por otro lado, en la misma figura se aprecia que los conceptos más usados fueron "Ucrania", "Rusia", "país", "Putin" y "guerra".

Por último, la Figura 5.39 reafirma lo esperable: este evento es altamente negativo, con casi la mitad de los artículos (119) catalogados como "muy negativo" y el resto como "neutrales". En contraste, el titular más positivo se titula "Emocionante registro: Joven fue captado tocando piano en Ucrania mientras se acercaban las tropas rusas" publicado por CHV.

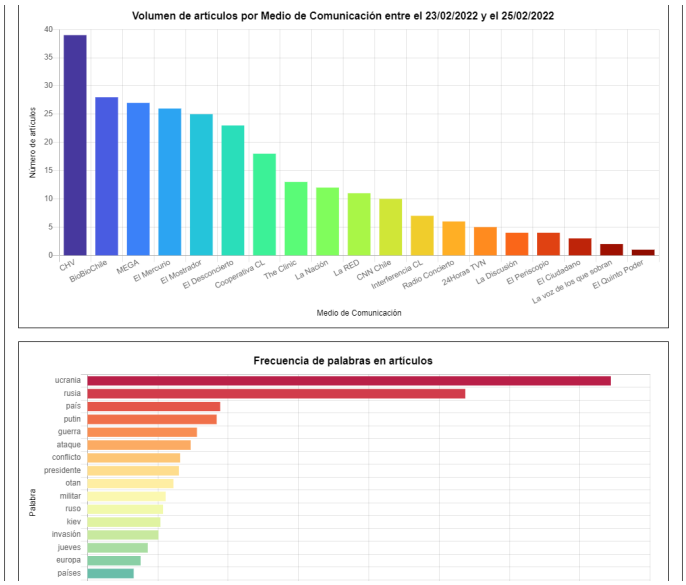

Figura 5.38: Gráfico de volumen de artículos por medio, y de términos más frecuentes para *cluster* más grande sobre el conflicto ruso-ucraniano

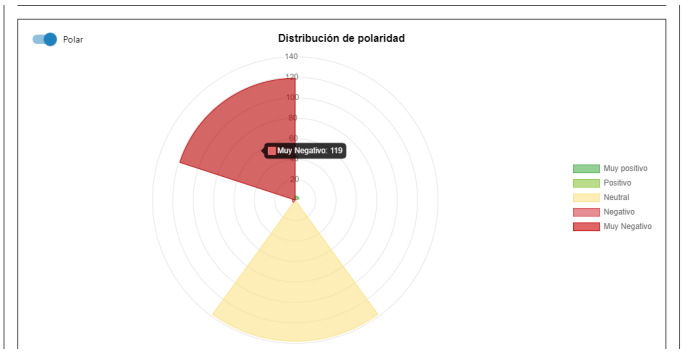

Figura 5.39: Gráfico polar de sentimientos para *cluster* más grande sobre el conflicto ruso-ucraniano

5.1.3. Análisis de polaridad y subjetividad de titulares generados

Como se menciona en la Sección 3.3.3, se desea averiguar si producir un titular a partir de los demás titulares mediante el modelo resumidor (en este caso *facebook/bart-large-cnn*[3]) produce un titular más neutro o más objetivo en comparación al promedio de métricas de los artículos que lo componen.

Para ello se hace un breve experimento. En primer lugar, se selecciona el modelo *RoBERTa Polarity* para comparar polaridades, destacando la neutralidad, y por otro lado, el modelo de *TextBlob* usado para medir la subjetividad. La elección del modelo de polaridad reside en que las métricas promedio de polaridad para cada evento se calcularon mediante este modelo, mientras que el titular se evaluó bajo los tres otros modelos. Notar que esto sucede por la limitación de los parámetros *accum_avg_polarity* y *accum_avg_subjectivity* en el *mapping* de *Eventos*, que obliga a escoger un modelo de polaridad y uno subjetividad para las métricas globales. Ver Sección 3.5.1.

En segundo lugar, se seleccionan todos los eventos procesados en el sistema y se comparan las métricas promedio con las del titular generado. En particular, se quiere saber el número de eventos donde el titular generado es más neutral que los que componen el artículo, cuando es menos subjetivo que los artículos, y cuando se cumplen ambas condiciones.

#	**Titular generado con IA**	**Positividad prom/titular**	**Neutralidad prom/titular**	**Negatividad prom/titular**	**Subjetividad prom/titular**
1	Conaf afirma haber visto pirómano durante los incendios forestales en el sector de Puerto Montt.	0.02/0.01	0.50/0.32	0.49/0.67	**0.16/0.00**
2	Mónica González: "Hemos fracasado como sociedad para no asumir el peso de esos 17 años" ...	0.10/0.01	0.42/0.18	0.49/0.81	0.10/0.30
3	Los pescadores de ENAP y Quintero llegan a un acuerdo de compensación de derrames de petr...	0.13/0.09	**0.68/0.83**	0.18/0.08	**0.22/0.00**
4	París dice que las cuarentenas para entrar en Chile siguen en vigor.	0.05/0.01	0.74/0.71	0.20/0.27	**0.28/0.00**
5	El Presidente de la Corte Suprema defiende la existencia del TC: "Sería un error suprimirlo".	0.05/0.10	0.84/0.78	0.10/0.12	0.00/0.00
6	Pfizer reporta menos eficacia de su vacuna en niños de 2 a 5 años de edad e intentará añadir ...	0.02/0.01	0.42/0.24	0.56/0.74	**0.34/0.23**
7	La Gendarmería denunció la fuga de un preso del Hospital Psiquiátrico. Están buscando a un ...	0.02/0.04	0.66/0.65	0.32/0.31	0.00/0.00
8	Plan paso a paso: Cartagena, Puchuncaví y otras 7 comunas avanzarán a la apertura inicial ...	0.09/0.09	**0.90/0.91**	0.01/0.00	**0.21/0.19**
9	Portavoces de la Comisión Nacional de Derechos Humanos se reunieron con asesores: Exig ...	0.03/0.06	**0.71/0.84**	0.26/0.10	**0.24/0.10**
10	Estados Unidos "respondirá con decisión" si Rusia invade Ucrania, dice Joe Biden.	0.11/0.06	**0.75/0.79**	0.14/0.15	0.00/0.00
11	Ronnie Fernández será el primer refuerzo de la Universidad de Chile en 2022.	0.38/0.07	**0.60/0.93**	0.01/0.01	0.30/0.33
12	Fuad Chahin y la crisis de la democracia cristiana: "El DC debe dejar atrás la ansiedad por ...	0.06/0.03	**0.70/0.77**	0.24/0.19	0.14/0.35
13	La ministra Paris habló de la partida de Daza para unirse a la campaña de Kast: "Para ella ...	0.09/0.03	0.77/0.73	0.14/0.24	**0.33/0.00**
14	La Comisión de Finanzas de la Casa aprueba impuestos a super ricos en financiación a PGU.	0.10/0.20	**0.71/0.78**	0.19/0.02	0.44/0.71
15	Palacios e idea de ampliar la prohibición de los recortes de servicios básicos: "Tenemos ...	0.21/0.05	**0.63/0.72**	0.16/0.23	**0.50/0.12**
16	La familia chilena asesinada por la policía en Estados Unidos cuenta con el apoyo de un ...	0.05/0.08	**0.56/0.72**	0.40/0.21	**0.13/0.00**
17	Regresar a Hong Kong un crucero con 2.500 personas para posibles casos covid.	0.04/0.03	**0.63/0.87**	0.33/0.10	0.36/1.00
18	Reino Unido rompe récord de infecciones covidias y advierte que "lo peor está por venir" ...	0.13/0.01	0.39/0.15	0.49/0.84	0.20/0.69
19	La salud ha bloqueado más de 700.000 pases de movilidad debido a la falta de la tercer ...	0.00/0.00	0.18/0.09	0.82/0.90	**0.43/0.29**
20	Alexis Sánchez comenzó y anotó un gol en triunfo del Inter de Milano.	0.22/0.56	0.66/0.44	0.12/0.00	**0.28/0.00**

Tabla 5.1: Muestra de 20 eventos donde se compara la neutralidad y subjetividad de los titulares generados con respecto al promedio de los artículos

De los 276 eventos registrados en el sistema, 146 (52.9 %) de los titulares generados son más neutrales, 141 (51.09 %) son menos subjetivos y 81 (29.35 %) cumplen ambas condiciones. Una muestra de 20 titulares extraídos del experimento se encuentran en la Tabla 5.1.

Mediante un breve análisis cualitativo de la muestra, es posible observar que, por ejemplo, los titulares 1, 3, 4, 13, 16 y 20 logran reducir la subjetividad a un valor de 0. El primer titular no contiene trazas de opinión ni adjetivos calificativos, el número 3 y 4 tampoco. Sin embargo, el

[3]https://huggingface.co/facebook/bart-large-cnn

número 13 contiene una cuña, una declaración, y, por lo tanto, corresponde a un falso negativo. El caso 16 y 20 también parecen ser casos correctos.

Por otro lado, el titular número tres logra aumentar la neutralidad al mismo tiempo que baja la positividad y la neutralidad. Mismo escenario en los titulares 12 y 17. Se tiene la hipótesis que esto sucede directamente por las palabras que usa el modelo para generar el titular, ya que reemplaza palabras generalmente asociadas a cosas positivas o negativas, por otras neutrales. Sin embargo, los titulares 14 y 16, aumentan la neutralidad, pero también la positividad. En particular, el titular 14 debió aumentar su positividad por el uso de las palabras "aprueba" y "ricos", mientras que el 16 por el uso de la palabra "apoyo".

De este breve experimento, es posible concluir que las hipótesis planteadas en la etapa de diseño son inciertas. Si bien, la mitad de los artículos fueron más neutrales o menos subjetivos, no es posible concluir que esto siempre va a ser así, mucho menos bajo el análisis de tan sólo 276 casos. Para validar estas hipótesis, se deberían analizar miles de casos, y bajo distintos modelos de subjetividad y polaridad.

Además, realizar esta evaluación solamente mediante *RoBERTa Polarity* teniendo otros dos modelos implementados es insuficiente, porque es sesgada a un sólo modelo. De todas formas, el hecho de tener dos alternativas para medir polaridad, valida que el sistema es flexible al momento de incluir nuevos modelos de ML.

A pesar de esto, se considera prometedor el resultado obtenido, al representar un primer paso para elaborar información lo más neutral y objetiva posible.

5.2. Validación con usuarios del *Proof of Concept*

5.2.1. Metodología

Se desea conocer si la idea detrás de la plataforma implementada es útil. No se quiere conocer si la implementación actual posee una buena usabilidad, o si tiene una interfaz atractiva. Se quiere validar el *proof of concept* del proyecto, y en específico, los distintos contextos donde esta herramienta puede ser usada.

Como se menciona en la Sección 3.8, el proceso de implementación del *frontend* en particular corresponde al punto más débil del presente trabajo. Por lo tanto, es altamente probable que la usabilidad del sistema sea baja. Lo cual no impide validar la prueba de concepto en términos de funcionalidad.

Se efectúan entrevistas a personas de diversos contextos. La estructura de las entrevistas consta de tres partes principales: (1) descripción del propósito de la herramienta, (2) mostrar las funcionalidades principales, y (3) observaciones generales.

La primera parte tiene por objetivo orientar al entrevistado al contexto de la plataforma, las problemáticas que busca resolver. Esta parte consta de una breve presentación donde se estipula que la plataforma busca:

1. Afrontar la sobrecarga de información, ordenando el flujo de noticias nacionales.
2. Afrontar la presencia de sesgos periodísticos, mediante las métricas de subjetividad y análisis de sentimientos.
3. Afrontar las burbujas mediáticas, catalizadas tanto por redes sociales como por la concentración de los medios en Chile, rompiendo el esquema y mostrando toda la gama periodística chilena.
4. Ser útil para la búsqueda de información periodística histórica del país. Disponible al público general, a modo de democratizar la información en Chile.

La segunda parte busca consolidar un diálogo con el entrevistado. Para ello se muestran las funcionalidades principales del sistema, y luego se responden varias preguntas con el objetivo de recoger los comentarios positivos y negativos del entrevistado. Cada pregunta tiene el propósito de validar los objetivos planteados inicialmente. El flujo de la entrevista es el siguiente:

En primer lugar, se informa al entrevistado sobre la existencia de tecnologías que permiten medir subjetividad y polaridad en texto. A modo de contexto, también se introducen modelos que clasifican un texto en discurso de odio, tendencia política, probabilidad de *fake news*, etc. Se explica que la idea es medir el sesgo periodístico de los artículos, pero que se debe proceder con responsabilidad.

En segundo lugar, se muestra la Vista buscador. Se muestran y describen los filtros presentes en el buscador. Se muestra un resultado de búsqueda: artículos entre el 16/12/2021 y 18/12/2021 con tres eventos principales: la muerte de Lucía Hiriart, el cierre de campañas presidenciales y una intervención en Plaza Baquedano. Y por último, se ordenan los resultados usando los distintos filtros disponibles.

Posteriormente, se procede a realizar las siguientes preguntas:

1. ¿Crees que es útil la búsqueda de información periodística, a modo general, pero además considerando en este las métricas de subjetividad y polaridad? ¿Crees que esto es útil en algún sentido?
2. Personalmente, ¿Para qué utilizarías un buscador de este estilo?
3. Actualmente ¿Qué herramienta usarías para cumplir esta labor?
4. ¿Crees que usarías esta herramienta para buscar noticias sobre eventos como la guerra en Ucrania, la inmigración o un evento relevante para ti?

Posteriormente, se muestra la Vista agrupación. Para ello se muestra una agrupación realizada entre las mismas fechas 16/12/2021 y 18/12/2021, para contrastar cómo la información es ordenada, en contraste con la Vista buscador. Al ingresar a la vista se describen los principales elementos de los eventos: título, resumen, palabras claves, fotografía, métricas y artículos. Se muestra un par de eventos a modo de ejemplo, y luego se realizan las siguientes preguntas:

1. ¿Te es fácil generar una idea general de este evento y encontrar la información importante para ti? ¿Qué elemento de la interfaz es lo que más te llama la atención?
2. Suele suceder que en redes sociales se ve la misma noticia una y otra vez, pero abordada por distintas fuentes de información. ¿Crees que esta forma de agrupar los artículos por eventos es una forma efectiva o amigable para ordenar la información del día a día?.
3. Entrando en el tema de burbujas informativas de redes sociales. ¿Le darías una oportunidad de lectura a medios que no conoces o que no son de tu preferencia, considerando las métricas de subjetividad y polaridad?
4. Como se dijo antes, existen modelos de inteligencia artificial que pueden predecir discurso de odio en texto, probabilidad de *fake news*, clasificar texto en tendencia política, etc. Sin embargo, el uso de estos filtros conlleva una responsabilidad mayor al calificar en forma más agresiva el trabajo de los periodistas. ¿Crees que estos filtros te servirían a ti para ordenar la información? Por ejemplo, de más a menos probable de que un titular sea discurso de odio.

Finalmente, en la tercera parte de la entrevista, se realizan dos preguntas generales:

1. Por último, si se desarrollara esta aplicación con una interfaz profesional, disponible para dispositivos móviles y de escritorio, que fuera de uso sencillo como redes sociales, donde la agrupación de eventos se muestre en forma de muro o*feed*, que es lo habitual, ¿Utilizarías esta aplicación para informarte? ¿Con cuánta frecuencia?
2. ¿Tienes algún comentario extra que quisieras mencionar?

El objetivo de responder estas 10 preguntas es obtener comentarios positivos y negativos del trabajo realizado, que ayuden a evaluar tanto la utilidad de la plataforma, como su uso en distintos contextos.

5.2.2. Resultados

Se entrevistaron veinticuatro personas cuyos perfiles se encuentran anexados en la Tabla E.1. Las entrevistas hechas duraron aproximadamente unos 30 minutos en total. Las ideas entregadas en detalle por cada uno de los entrevistados se adjuntan en Anexo E.2, E.3 y E.4.

Todos los entrevistados asociados al periodismo señalan que esta herramienta es más bien profesional y que les sería muy útil en su trabajo del día a día, ahorrándoles tiempo, pero también para realizar investigación o utilizarla en el ámbito académico y universitario. De hecho, la mayoría advierte que el uso de métricas de polaridad y subjetividad quizás no serían usadas por un público masivo.

En contraste, otro grupo de usuarios señala que haría uso de esta aplicación para informarse en el día a día, y que les sería bastante útil para contrastar posturas muchas veces antagónicas respecto a un mismo evento. Muchos utilizan el término "para ver las dos caras de la moneda".

Además añaden que utilizarían esta aplicación en sus dispositivos móviles de desarrollarse profesionalmente, y que generalmente, la usarían con la misma frecuencia que hoy en día utilizan otras aplicaciones para informarse, o bien, sólo para buscar temas que sean de su profundo interés.

Las entrevistadas de México y Argentina validan el uso de esta aplicación en otros países, ya que señalan la utilidad del mismo en otros contextos. Siendo México un caso extremo, ambas señalan que el problema de la concentración de los medios también se encuentra en sus países, y que esta plataforma les serviría para contrastar nuevas perspectivas.

Varios entrevistados también advierten el tema del sesgo en los modelos de *machine learning*, en una postura de desconfianza y/o también de curiosidad, por ejemplo, para contrastar los propios sesgos hacia los medios, con los que propone la plataforma. De hecho, por este mismo motivo, indican que la plataforma incentiva el ejercicio de contraste de información.

Tanto una profesora de lenguaje de educación media, como una profesora de la carrera de periodismo, comentan la gran utilidad que tendría una plataforma así en el contexto educacional. No tan sólo para preparar material docente, sino también como una herramienta útil para realizar trabajos de investigación por parte de los estudiantes.

Muchos entrevistados también señalan, que es una plataforma que beneficia a medios regionalistas o independientes, ya que ganan un espacio que les es difícil tener en plataformas como Google. Sirve también, para romper ciertos esquemas como "todas las noticias son trágicas" o para identificar medios que tengan cierto comportamiento al difundir discurso de odio o discriminar minorías.

Por otro lado, dos o tres entrevistados califican a la plataforma como una "biblioteca digital" o una "hemeroteca digital", útil para la búsqueda de información histórica. Una psicóloga también plantea el uso de esta herramienta para informarse, pero en un contexto de salud mental, donde se busca evitar noticias negativas.

Muchos usuarios encuentran la plataforma amigable e intuitiva en diseño, sobre todo del *layout* de eventos, pero al mismo tiempo, otros advierten que los colores están mal, o que las letras son muy chicas, que no hay una mayor explicación de los filtros, entre otras cosas. Generalmente, sugieren mejoras al sistema, y evidencian una baja usabilidad del mismo.

Finalmente, es claro evidenciar que los entrevistados proponen múltiples casos de usos distintos para la plataforma, ya sea en el plano académico, personal, educacional y profesional.

Capítulo 6

Conclusiones

6.1. Conclusiones generales

Se cumplen los objetivos específicos planteados al inicio de este trabajo. En primer lugar, se cumple la creación de un recolector de noticias para medios tradicionales e independientes, mediante el uso de 27 *scrappers*. En segundo lugar, se diseña e implementan dos metodologías para agrupar noticias, una basada en algoritmos de *clustering*, junto a *embeddings* generados con SBERT, y la otra utilizando una heurística de palabras clave.

En tercer lugar, se cumple el objetivo de diseñar e implementar una metodología para generar métricas de sesgo periodístico, mediante el cálculo de subjetividad y polaridad en titulares de noticias, usando modelos *estado del arte* de ML. En cuarto lugar, se crea una aplicación web que permite la búsqueda de noticias en el sistema, mediante el desarrollo de un *frontend* y de los microservicios pertinentes.

En quinto lugar, el sistema implementado es flexible y escalable, debido al énfasis puesto en el modelo de datos y la arquitectura del sistema. La incorporación de nuevas formas de agrupar noticias, o de nuevas maneras de medir sesgos periodísticos, es directa. Por último, se logra evaluar cualitativamente el sistema, mediante el análisis de casos de uso y el *feedback* recibido en las entrevistas a usuarios, provenientes de diversas áreas profesionales.

De esta manera, el objetivo general es también alcanzado, porque la sección de agrupaciones en el *frontend*, facilita el contraste de perspectivas y opiniones, al mostrar noticias de distintos medios respecto a un mismo hecho o tópico. Además, tanto las métricas de sesgos periodísticos, como el abanico de los medios implementados, logran mitigar el efecto de las burbujas informativas. Así lo declaran también, los usuarios entrevistados.

En cuento al sistema implementado, se valida que esta herramienta es factible técnicamente, pero que requiere de recursos para poder escalar horizontalmente. Al mismo tiempo, la mayoría de las partes del sistema, son perfectibles a nivel de *backend* y de *frontend*.

Respecto a los métodos de agrupación, la heurística de palabras es el método más rápido y que brinda buenos resultados, ya que logra agrupar noticias de un mismo tema en la mayoría

de los casos de estudio analizados. Además, da garantías de que cada titular perteneciente a un artículo tendrá al menos las palabras clave de ese evento. Si las agrupaciones son hechas en ventanas de tiempo pequeñas, es poco probable que el error sea grande. Por último, la optimización hecha para este algoritmo es fundamental para evitar *clusters* disjuntos, pero que tengan una misma temática.

Por otro lado, el método basado en *embeddings* es más complejo y costoso de ejecutar. Sin embargo, al utilizar *Agglomerative clustering* junto al criterio no supervisado *Silhouette*, se obtienen buenos resultados en poco tiempo (con base en los casos analizados). Sin embargo, sucede lo mismo que con el método de heurística sin la optimización final: existen *clusters* que son del mismo tópico, pero disjuntos. Esto se debe justamente a que los *clusters* formados son más densos y específicos. Por lo tanto, a este método le falta agregar la optimización elaborada para la heurística, para que los resultados puedan mejorar.

En ambos casos se encuentra un dilema de criterio: ¿Cuál es el grado de fineza con el cual se quieren agrupar los artículos?. Evidenciado en los casos de uso analizados, los eventos con más cobertura logran quedar en un sólo *cluster* de utilizar la heurística, pero en *clusters* separados de usar *embeddings*. Ambos resultados son correctos desde el punto de vista de la agrupación. Sin embargo, el primer caso, al reunir tantos artículos en un solo evento, provoca la pérdida de especificidad de los eventos, mientras que el segundo caso, genera un aumento en especificidad, pero se pierde en términos de sobrecarga de la información. Se estipula que la solución ideal abarca ambos criterios: se deberían agrupar eventos de alta cobertura en un solo evento, pero luego identificar sub-eventos en este evento, lo cual se plantea como desafío futuro.

Por otro lado, para mejorar los resultados de las agrupaciones con cualquiera de estas metodologías, se podrían desarrollar heurísticas de acuerdo al *top* 5 o *top* 10 conceptos presentes en los cuerpos de cada artículo, ya que son mucho más representativos de los artículos que el titular. Tanto para mejorar el resultado inicial de las agrupaciones, o bien, para juntar *clusters* que sean altamente similares.

Se realizan diversos experimentos para evaluar la subjetividad y polaridad en texto. Si bien existen falsos positivos y falsos negativos, sobre todo para la subjetividad, en la mayoría de los casos se obtiene el comportamiento esperado. Por ejemplo, en base a los casos analizados, los titulares con cuñas son identificados como subjetivos, aquellos con las palabras "muerte" o "división" como negativos, y aquellos con las palabras "emocionante" o "triunfo" como positivos. Es imperioso desarrollar un método supervisado de reevaluación de estas métricas, pero esto requiere de recursos.

La generación automática de titulares brinda resultados inciertos, pero al mismo tiempo representa un primer paso para elaborar información lo más neutra posible. Los experimentos hechos evidencian que el modelo produce resultados muchas veces menos subjetivos, con algunos textos incoherentes, pero también resultados que logran sintetizar un evento de forma coherente. Por otro lado, los casos de estudio analizados para la generación automática de resúmenes evidencian falencias en los modelos utilizados. Por ejemplo, añadiendo contenido que no existe en las noticias que resume o generando frases fuera de contexto (*context fragmentation*), sobre todo cuando el sujeto está implícito. En un futuro, el titular de los eventos se podría escoger aleatoriamente entre las noticias que lo componen (citando al medio), y por otro lado, se puede experimentar con modelos resumidores extractivos para la generación de resúmenes.

En cuanto a la validación de la utilidad del sistema, las 24 entrevistas hechas evidencian un interés generalizado por la plataforma y para distintos casos de uso. Los más destacables, proponen usarla para investigaciones periodísticas, en centros educaciones de enseñanza básica y media, la generación de *insights* para empresas y el más repetido de todos: para informarse en forma diaria.

Todos los entrevistados estuvieron interesados en usar la aplicación para buscar y contrastar información en el día a día, si esta se desarrollase profesionalmente. Por lo tanto, la oportunidad de desarrollo a futuro es prometedora, siempre y cuando se defina un nicho de usuarios en concreto, lo cual representa uno de los primeros desafíos a futuro.

Como era esperable, la interfaz de usuario y la usabilidad del sistema no es la mejor, pues entrevistados lo declararon así, encontrando detalles y sugiriendo mejoras. Lo destacable es, que aún teniendo en cuenta todas imperfecciones del sistema, los entrevistados consideran el prototipo expuesto, una herramienta útil.

Finalmente, todo el trabajo aquí presente se considera un buen acercamiento hacia una solución profesional, pero que requiere de más recursos para poder desarrollarse en profundidad. Representa una oportunidad de investigación, en el plano de ciencias de la computación con respecto al procesamiento del lenguaje natural, algoritmos y heurísticas, pero también en el plano de las ciencias sociales y humanidades.

6.2. Trabajo futuro

La tarea más importante a ejecutarse en un futuro es la definición del nicho de usuarios que usaría este sistema, dado que muchos entrevistados constatan que puede ser una herramienta de uso profesional, o una herramienta de uso masivo. Esta decisión no es sencilla, ya que implica cosas totalmente diferentes. Tal vez, en un punto se pueda crear un tipo de producto para cada tipo de usuarios, pero esto compromete, al mismo tiempo, contar con grandes cantidades de recursos.

Por otro lado, este trabajo demuestra la necesidad de contar con un equipo de trabajo más amplio y profesional, para poder llevar esta aplicación a un producto final. La arquitectura basada en microservicios permite involucrar a más desarrolladores de *software*, sin comprometer un coste tan alto en temas logísticos. Por esto, comenzar un emprendimiento con este trabajo no es una realidad imposible, siempre y cuando se cuente con algún fondo de financiamiento.

En cuanto a aspectos técnicos, la mejora más grande a incorporar, debe ser abordar la detección de eventos en forma incremental y sin repetición. Esto es, utilizar el mejor algoritmo de *clustering*, y solapar resultados en ventanas de tiempo secuenciales, detectando eventos en común y juntar los artículos sin repetición. Este es un problema no fácil de abordar, si bien es similar a un *clustering online*, en este caso presenta el desafío de contar con datos entrantes y salientes constantemente de manera masiva. Además, este es un problema complejo de evaluar. Sin embargo, esta mejora permite evitar datos redundantes en la base de datos y mejorar la experiencia a nivel de usuario.

Una mejora referente a los modelos PLN, es que si bien se usan modelos del *estado del arte* para procesar texto, que han sido evaluados en *datasets* ampliamente usados en investigación, sería interesante poder generar nuevos *datasets*, pero en español. Con los artículos procesados, se podrían reevaluar los distintos modelos y *pipelines* del sistema. Sin embargo, para esto se necesita de un proceso de *user annotation* que es bastante costoso de realizar. Reevaluar los modelos de subjetividad, podría dar métricas más exactas de la confiabilidad de los modelos utilizados. Lo mismo es aplicable para la polaridad. Quizás el modelo más complejo de reevaluar es el de *summarization*, donde se debería efectuar un análisis más profundo.

Otra mejora notable al sistema, es incorporar nuevos modelos de procesamiento de texto, aprovechando que el sistema es flexible y escalable. Se pueden incluir nuevos modelos para las tareas PLN ya existentes, o bien, incorporar nuevas tareas PLN. Muchos entrevistados señalaron que un filtro de discurso de odio, o de probabilidad de *fake news* les sería bastante útil.

En cuanto a conexiones con Twitter, se pueden extraer *tweets* que contengan las palabras claves de un evento noticioso, para incorporar reacciones de personas sobre un evento, bajo una ventana de tiempo fija. Además, se podrían utilizar los mismos modelos de polaridad y subjetividad para identificar tendencias, pero esta vez, en las personas.

En el plano del *backend*, existen problemas de escalabilidad horizontal que deben ser abordados si se espera continuar con este trabajo. Las consultas al sistema comprenden un alto volumen de datos. Se debe desarrollar todo un sistema de mantenimiento de la base de datos, tener múltiples servidores para soportar cientos y miles de solicitudes por minuto, y en general, velar por la seguridad de todos los procesos en cada paso.

Por otro lado, sería útil saber si las noticias recolectadas son verídicas, no verificadas, o falsas. De esta forma se pueden identificar los medios que desinforman al público general. Para esto es necesario apoyarse en organizaciones que verifican noticias como FastCheck[1], Fact Checking UC[2], Factchecking de La Tercera[3] o Mala Espina Check[4]. También en oráculos de periodistas u organizaciones especializadas profesionales. Idea que también sugieren algunos entrevistados.

En cuento a la concentración de los medios, se podría ingresar al sistema la información de propiedad de cada medio, y comparar la similitud de contenido entre los medios que pertenecen a una misma línea periodística. Se podrían hacer estudios de comportamiento histórico de medios, e incluso analizar si un autor cambia su línea periodística al cambiar de medio.

Con un sistema más robusto y completo, se podría permitir que ciertas entidades, periodistas o usuarios comunes puedan realizar denuncias, a través de la plataforma, sobre ciertos medios por faltas de ética, malas prácticas periodistas o divulgación de *fake news*. De esta forma, incorporar de manera histórica estos hechos y que en conjunto al análisis de texto de un medio, den una idea más clara del comportamiento de cada medio. Tener esta información ordenada e histórica, permitiría, de cierto modo, auditar a los medios nacionales en forma más transparente, no tan sólo en su línea periodística, pero también teniendo presente hechos graves que no pueden ser detectados con el análisis de texto.

[1]https://www.fastcheck.cl/
[2]https://factchecking.cl/
[3]https://www.latercera.com/factchecking/
[4]https://www.malaespinacheck.cl/

Se podría generar una ponderación y *ranking* de los medios de comunicación, de acuerdo a su grado ética, veracidad, interpretación y opinión. Y con este *ranking*, se podría ordenar la información de más objetiva a menos objetiva para cada noticia en el sistema, en forma dinámica.

En definitiva, las mejoras al sistema requieren de recursos y financiamiento para poder ejecutarse. Además, algunas de ellas requieren proceder con mayor responsabilidad, y otras, de validaciones costosas. Se espera que, de surgir un emprendimiento con este trabajo, puedan llevarse a cabo.

También es muy importante contar en un futuro con una asesoría legal en temas de propiedad intelectual, para esclarecer los límites de un emprendiendo basado en este trabajo. Si bien parece no haber problemas con la legislación actual, es pertinente realizar una investigación más a fondo, y estar al tanto de las modificaciones a la ley.

Por último, en temas prácticos, se requiere de un análisis de mercado y/o evaluación de proyecto más profundo para evaluar si el comienzo de una *startup* con esta herramienta es sustentable económicamente, ya que ideas como añadir publicidad o recibir donaciones parecen ser insuficientes. Sin embargo, de acuerdo al *feedback* de los entrevistados, es prácticamente seguro que un producto final de esta plataforma sería usado por muchas personas, bajo múltiples contextos.

Bibliografía

[1] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.

[2] Mehdi Allahyari, Seyed Amin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys J. Kochut. Text summarization techniques: A brief survey. *CoRR*, abs/1707.02268, 2017. URL: `http://arxiv.org/abs/1707.02268`, `arXiv:1707.02268`.

[3] María José Anguita Ramírez, Pedro Labrador Blanes. Pluralismo y libre competencia en el mercado de la televisión y radiodifusión: el caso chileno. *Revista de Comunicación*, 18, 01 2012. URL: `https://revistadecomunicacion.com/article/view/1017`.

[4] Aymé Arango, Jorge Pérez, and Barbara Poblete. Hate speech detection is not as easy as you may think: A closer look at model validation. SIGIR'19, page 45–54, New York, NY, USA, 2019. Association for Computing Machinery. `doi:10.1145/3331184.3331262`.

[5] David P Baron. Persistent media bias. *Journal of Public Economics*, 90(1-2):1–36, 2006.

[6] David Bawden and Lyn Robinson. Information overload: An overview. *Oxford encyclopedia of political decision making*, 2020.

[7] Kent Beck. *Test-driven development: by example*. Addison-Wesley Professional, 2003.

[8] Hila Becker, Dan Iter, Mor Naaman, and Luis Gravano. Identifying content for planned events across social media sites. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, page 533–542, New York, NY, USA, 2012. Association for Computing Machinery. `doi:10.1145/2124295.2124360`.

[9] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020. URL: `https://arxiv.org/abs/2004.05150`, `arXiv:2004.05150`.

[10] Shai Ben-David, Dávid Pál, and Hans Ulrich Simon. Stability of k-means clustering. In Nader H. Bshouty and Claudio Gentile, editors, *Learning Theory*, pages 20–34, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[11] James Briggs. Bert for next sentence prediction: The other half to pretraining bert. URL: `https://towardsdatascience.com/bert-for-next-sentence-prediction-466b67f8226f`.

[12] James Briggs. Masked-language modeling with bert: Fine-tune your models on any dataset. URL: https://towardsdatascience.com/masked-language-modelling-with-bert-7d49793e5d2c.

[13] Costin Busioc, Stefan Ruseti, and Mihai Dascalu. A literature review of nlp approaches to fake news detection and their applicability to romanianlanguage news analysis. *Revista Transilvania*, (10), 2020.

[14] Cadem. El futuro de los medios, 2020. URL: https://www.cadem.cl/encuestas/el-futuro-de-los-medios/.

[15] T. Caliński and J Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1):1–27, 1974. URL: https://www.tandfonline.com/doi/abs/10.1080/03610927408827101, arXiv:https://www.tandfonline.com/doi/pdf/10.1080/03610927408827101, doi:10.1080/03610927408827101.

[16] José Cañete, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez. Spanish pre-trained bert model and evaluation data. 2020.

[17] KAICIID Dialogue Center. Los expertos constatan que el aumento del discurso del odio es consecuencia de la pandemia, 2021. URL: https://www.kaiciid.org/es/noticias-eventos/noticias/los-expertos-constatan-que-el-aumento-del-discurso-del-odio-es.

[18] Universidad Central. Director de fast check cl advirtió sobre la evolución de las fake news hacia el desorden de información, 2021. URL: https://www.ucentral.cl/noticias/fac-economia-gobierno-y-comunicaciones/esc-gobierno-y-comunicaciones/director-de-fast-check-cl-advirtio-sobre-la-evolucion-de-las-fake-news.

[19] Reuters Fact Check. Fact check-la prueba del "brazo magnético"no es evidencia que la vacuna de covid-19 contenga un rastreador o un chip, 2021. URL: https://www.reuters.com/article/factcheck-vacuna-magnetismo-idUSL2N2N62BT.

[20] CodeEmporium. Bert neural network - explained! URL: https://www.youtube.com/watch?v=xI0HHN5XKDo.

[21] Huggingface Course. How do transformers work? URL: https://huggingface.co/course/chapter1/4.

[22] Huggingface Course. Transformers: summary. URL: https://huggingface.co/course/chapter1/9.

[23] Brad J Cox. *Object oriented programming: an evolutionary approach*. Addison-Wesley Longman Publishing Co., Inc., 1986.

[24] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *CoRR*, abs/1901.02860, 2019. URL: http://arxiv.org/abs/1901.02860, arXiv:1901.02860.

[25] Biblioteca Nacional de Chile. Registro de nuevos medios escritos. URL: https://tinyurl.com/yydef68f.

[26] Franciso de Lara y Carolina Urrejola. La posverdad desde el punto de vista filosófico de francisco de lara, 2022. URL: https://www.tele13radio.cl/podcast/nativos/la-posverdad-desde-el-punto-de-vista-filosofico-de-francisco-de-lara.

[27] Universidad de Playa Ancha. Decano denuncia problemas éticos y criminalización por parte de los medios de prensa chilenos, 2019. URL: https://tinyurl.com/5xm7a2yh.

[28] William de Vazelhes, CJ Carey, Yuan Tang, Nathalie Vauquier, and Aurélien Bellet. metric-learn: Metric Learning Algorithms in Python. *Journal of Machine Learning Research*, 21(138):1–6, 2020.

[29] Jacob Devlin and Ming-Wei Chang. Open sourcing bert: State-of-the-art pre-training for natural language processing. *Google AI Blog*, 2018. URL: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html.

[30] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL: http://arxiv.org/abs/1810.04805, arXiv:1810.04805.

[31] DW. La pandemia generó un aumento en el discurso de odio contra personas lgbti, 2021. URL: https://www.dw.com/es/la-pandemia-gener%C3%B3-un-aumento-en-el-discurso-de-odio-contra-personas-lgbti/a-56592772.

[32] Jacob Eisenstein. *Introduction to Natural Language Processing*. 2019.

[33] Elastic. Data in: documents and indices, 2020. URL: https://www.elastic.co/guide/en/elasticsearch/reference/current/documents-indices.html.

[34] France24. Chile: los medios de comunicación tradicionales no se escapan del descontento, 2020. URL: https://www.france24.com/es/20200131-protestas-chile-desconfianza-medios-tradicionales-plataformas.

[35] Andrea Galassi, Marco Lippi, and Paolo Torroni. Attention in natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(10):4291–4308, 2020.

[36] Rama Rohit Reddy Gangula, Suma Reddy Duggenpudi, and Radhika Mamidi. Detecting political bias in news articles using headline attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 77–84, 2019.

[37] Bolin Gao and Lacra Pavel. On the properties of the softmax function with application in game theory and reinforcement learning. *arXiv preprint arXiv:1704.00805*, 2017.

[38] Matt W Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

[39] Tim Groseclose and Jeffrey Milyo. A measure of media bias. *The Quarterly Journal of Economics*, 120(4):1191–1237, 2005.

[40] Felix Hamborg, Karsten Donnay, and Bela Gipp. Automated identification of media bias in news articles: an interdisciplinary literature review. *International Journal on Digital Libraries*, 20(4):391–415, 2019.

[41] John B. Horrigan. Information overload, 2016. URL: http://assets.pewresearch.org/wp-content/uploads/sites/14/2016/12/05162316/PI_2016.12.07_Information-Overload_FINAL.pdf.

[42] Huggingface. Translation. URL: https://huggingface.co/tasks/translation.

[43] Marina Danchovsky Ibrishimova and Kin Fun Li. A machine learning approach to fake news detection using knowledge verification and natural language processing. In *International Conference on Intelligent Networking and Collaborative Systems*, pages 223–234. Springer, 2019.

[44] Ley 17336. Propiedad Itelectual. Título III. Limitaciones y excepciones al derecho de autor y a los derechos conexos. *Biblioteca del Congreso Nacional de Chile*, 2017. URL: https://www.bcn.cl/leychile/navegar?idNorma=28933&idParte=8917017.

[45] Amnistía Internacional. Cómo hacer frente a la sobrecarga informativa durante la crisis de covid-19, 2020. URL: https://www.amnesty.org/es/latest/news/2020/04/how-to-deal-with-news-overload-during-the-covid-19-crisis/.

[46] Bahamonde J., Bollen J., Elejalde E., Ferres L., and Poblete B. Power structure in chilean news media. *PLoS ONE*, 2018. URL: https://doi.org/10.1371/journal.pone.0197150.

[47] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. SMART: robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *CoRR*, abs/1911.03437, 2019. URL: http://arxiv.org/abs/1911.03437, arXiv:1911.03437.

[48] Jonah Engel Bromwich Jocey Fortin. Sinclair made dozens of local news anchors recite the same script. *The New York Times*, 04 2018. URL: https://www.nytimes.com/2018/04/02/business/media/sinclair-news-anchors-script.html.

[49] Sergio Toro Maureira y Sebastián Valenzuela Juan Pablo Luna. El ruidoso silencio de los medios tradicionales, 2021. URL: https://www.ciperchile.cl/2021/03/23/el-ruidoso-silencio-de-los-medios-tradicionales/.

[50] Janani Kalyanam, Mauricio Quezada, Barbara Poblete, and Gert Lanckriet. Prediction and characterization of high-activity events in social media triggered by real-world news. *PLOS ONE*, 11(12):1–13, 12 2016. doi:10.1371/journal.pone.0166694.

[51] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *CoRR*, abs/2001.04451, 2020. URL: https://arxiv.org/abs/2001.04451, arXiv:2001.04451.

[52] Vijay Kumar, Jitender Kumar Chhabra, and Dinesh Kumar. Performance evaluation of distance metrics in the clustering algorithms. *INFOCOMP Journal of Computer Science*, 13(1):38–52, 2014.

[53] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019. URL: http://arxiv.org/abs/1909.11942, arXiv:1909.11942.

[54] José Rafael Lantigua. El sesgo mediático: los colores del periodismo, 2019. URL: https://www.diariolibre.com/opinion/lecturas/el-sesgo-mediatico-los-colores-del-periodismo-II14940686.

[55] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019. URL: http://arxiv.org/abs/1910.13461, arXiv:1910.13461.

[56] John Light. How media consolidation threatens democracy: 857 channels (and nothing on). *Moyers and Company*, 05 2017. URL: https://billmoyers.com/story/media-consolidation-should-anyone-care/.

[57] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. pages 74–81, July 2004. URL: https://www.aclweb.org/anthology/W04-1013.

[58] Yang Liu. Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*, 2019.

[59] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[60] T Soni Madhulatha. An overview on clustering methods. *arXiv preprint arXiv:1205.1117*, 2012.

[61] Goutam Majumder, Partha Pakray, Alexander Gelbukh, and David Pinto. Semantic textual similarity methods, tools, and applications: A survey. *Computación y Sistemas*, 20(4):647–665, 2016.

[62] Inderjeet Mani. *Automatic summarization*, volume 3. John Benjamins Publishing, 2001.

[63] Noortje Marres and Esther Weltevrede. Scraping the social? *Journal of Cultural Economy*, 6(3):313–335, 2013. arXiv:https://doi.org/10.1080/17530350.2013.772070, doi:10.1080/17530350.2013.772070.

[64] Robert C Martin, James Grenning, Simon Brown, Kevlin Henney, and Jason Gorman. *Clean architecture: a craftsman's guide to software structure and design*. Number s 31. Prentice Hall, 2018.

[65] El Martutino. [video] por falsa manifestación en valparaíso: Denuncian falta de ética de mega y canal anuncia despidos, 2019. URL: https://tinyurl.com/2xyvm629.

[66] Diana Maynard and Adam Funk. Automatic detection of political opinions in tweets. In *Extended semantic web conference*, pages 88–99. Springer, 2011.

[67] Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. Efficient estimation of word representations in vector space. pages 1–12, 01 2013.

[68] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.

[69] BBC News Mundo. Coronavirus en brasil: las frases del presidente jair bolsonaro que han marcado el impacto de la pandemia en brasil, el país más golpeado de américa latina, 2020. URL: https://www.bbc.com/mundo/noticias-52652662.

[70] Yasna Mussa. Académicos evidencian racismo y sesgo en la prensa chilena al informar sobre migrantes, 2016. URL: https://tinyurl.com/yxqsqoez.

[71] Eli Parisier. Beware online "filter bubbles". URL: https://www.ted.com/talks/eli_pariser_beware_online_filter_bubbles/.

[72] Eli Parisier. What obligation do social media platforms have to the greater good? URL: https://tinyurl.com/2wm9k9xf.

[73] Federico Pascual. Topic modeling: An introduction. *MonekyLearn Blog*, 2019. URL: https://monkeylearn.com/blog/introduction-to-topic-modeling/.

[74] Vanessa Peña-Araya, Mauricio Quezada, Barbara Poblete, and Denis Parra. Gaining historical and international relations insights from social media: spatio-temporal real-world news analysis using twitter. *EPJ Data Science*, 6(1):25, Oct 2017. doi:10.1140/epjds/s13688-017-0122-8.

[75] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL: http://www.aclweb.org/anthology/D14-1162.

[76] D T Pham, S S Dimov, and C D Nguyen. Selection of k in k-means clustering. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 219(1):103–119, 2005. arXiv:https://doi.org/10.1243/095440605X8298, doi:10.1243/095440605X8298.

[77] The A.I. Hacker Michael Phi. Illustrated guide to transformers neural network: A step by step explanation. URL: https://www.youtube.com/watch?v=4Bdc55j80l8.

[78] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[79] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL: http://jmlr.org/papers/v21/20-074.html.

[80] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084, 2019. URL: http://arxiv.org/abs/1908.10084, arXiv:1908.10084.

[81] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL: https://aclanthology.org/D07-1043.

[82] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. URL: https://www.sciencedirect.com/science/article/pii/0377042787901257, doi:https://doi.org/10.1016/0377-0427(87)90125-7.

[83] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL: http://arxiv.org/abs/1910.01108, arXiv:1910.01108.

[84] Francisca Delgado. Pedro Santander:. "en las campaÑas presidenciales nos enfrentamos a un discurso de odio y violencia", 2021. URL: https://diariousach.cl/programas-radio-usach/razones-editoriales/pedro-santander-en-las-campanas-presidenciales-nos-enfrentamos-a-un.

[85] Tina Schuh. Feature selection in aggressive comment detection. 2017.

[86] Amit Singhal et al. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.

[87] Nick Statt. Google personalizes search results even when you're logged out, new study claims. URL: https://www.theverge.com/2018/12/4/18124718/google-search-results-personalized-unique-duckduckgo-filter-bubble.

[88] Teresa Sádaba-Garraza. Origen, aplicación y límites de la "teoría del encuadre" (framing) en comunicación. 2001.

[89] teaLeef. Why is euclidean distance not a good metric in high dimensions. URL: https://stats.stackexchange.com/questions/99171/why-is-euclidean-distance-not-a-good-metric-in-high-dimensions.

[90] Tippaya Thinsungnoena, Nuntawut Kaoungkub, Pongsakorn Durongdumronchaib, Kittisak Kerdprasopb, and Nittaya Kerdprasopb. The clustering validity with silhouette and sum of squared errors. *learning*, 3(7), 2015.

[91] Ciclos UDP. Confianza en medios de comunicación, 2019. URL: https://ciclos.udp.cl/2020/10/21/cofianza-en-medios-de-comunicacion/.

[92] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL: http://arxiv.org/abs/1706.03762, arXiv:1706.03762.

[93] Minh Vu. Political news bias detection using machine learning. *URL: https://pdfs. semanticscholar. org/8445/2eb068bdfe7d5809734a5da8f5c7d10bebfa. pdf*, 2017.

[94] William Yang Wang. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*, 2017.

[95] Wikipedia. News aggregator. URL: `https://en.wikipedia.org/wiki/News_aggregator`.

[96] Roberto Aparici y David García-Marín. La posverdad de la burbuja informativa, 2019. URL: `https://theconversation.com/la-posverdad-de-la-burbuja-informativa-118657`.

[97] Selim Firat Yilmaz, E. Batuhan Kaynak, Aykut Koç, Hamdi Dibeklioglu, and Suleyman Serdar Kozat. Multi-label sentiment analysis on 100 languages with dynamic weighting for label imbalance. *CoRR*, abs/2008.11573, 2020. URL: `https://arxiv.org/abs/2008.11573`, `arXiv:2008.11573`.

[98] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021.

[99] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

Anexos

Anexo A

Software similar

A.1. Event Registry

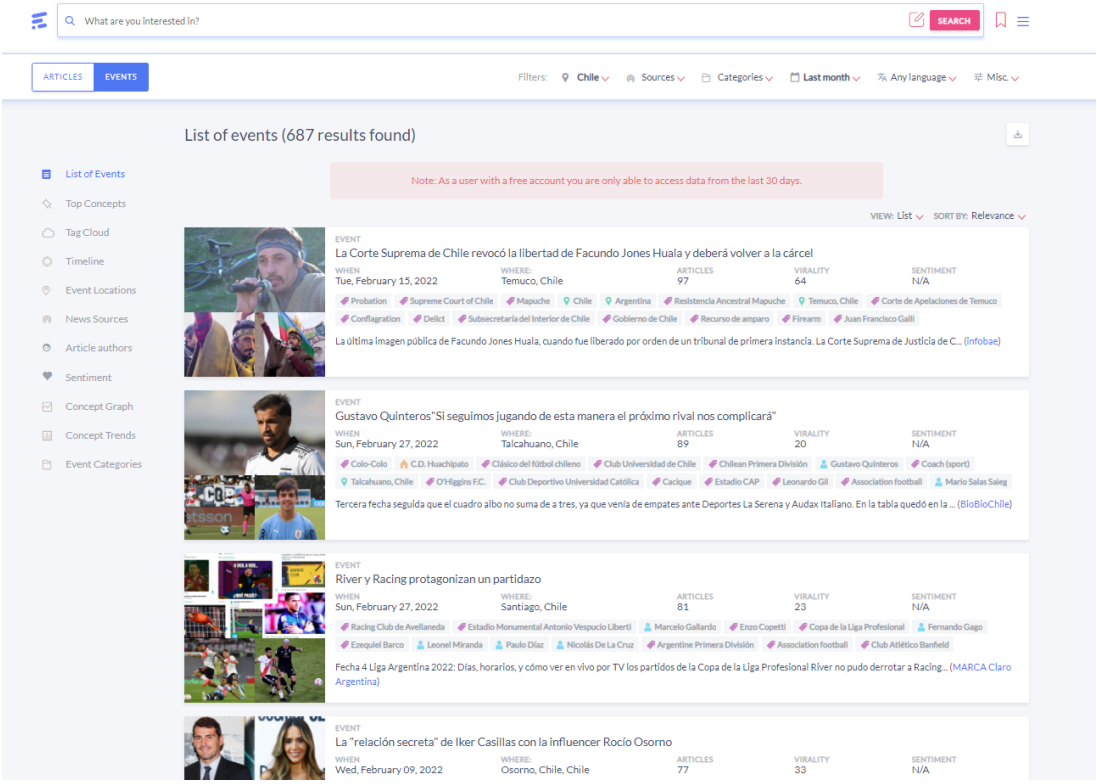

Figura A.1: Event Registry: Vista agrupación

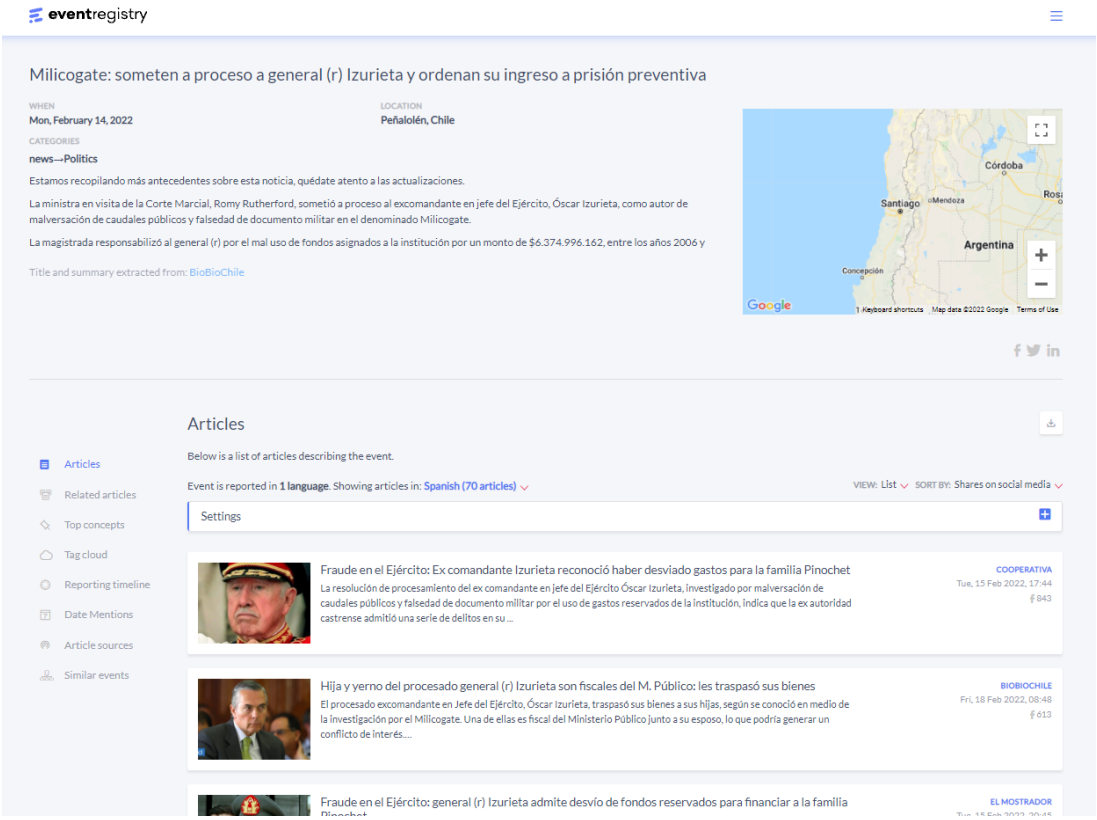

Figura A.2: Event Registry: Vista detalle de evento

A.2. Google News

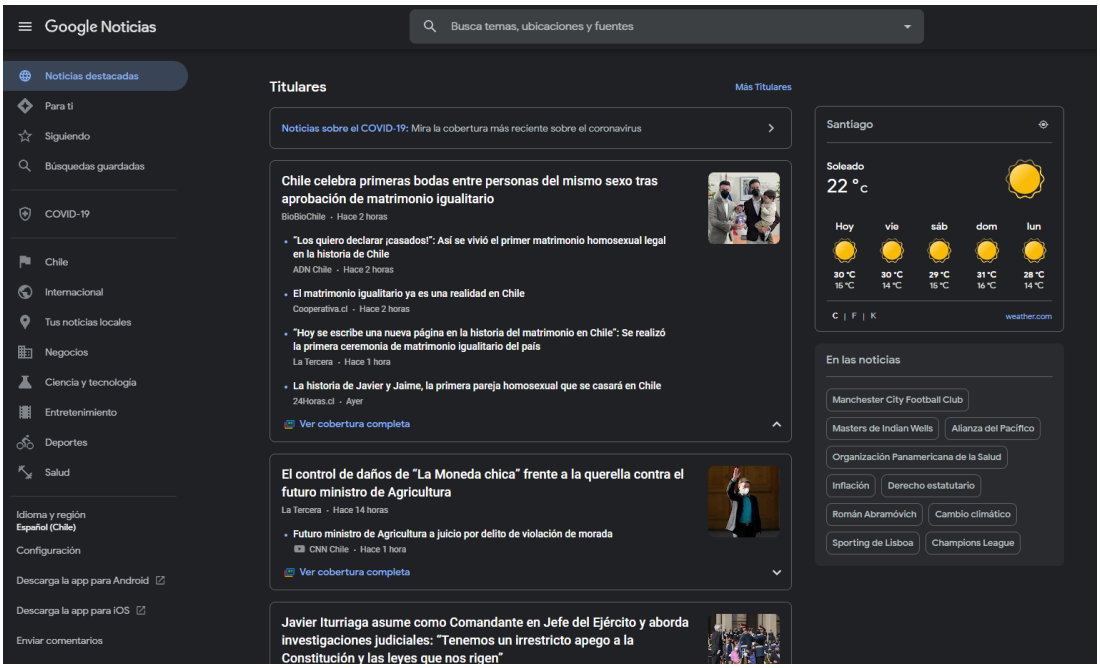

Figura A.3: Google News: Vista agrupación

Anexo B

Mappings para Elasticsearch

B.1. *Mapping* de artículos

```
"media_id" : { "type" : "keyword" },
"date_scrapping" : {
    "type": "date",
    "format": "date_optional_time"
},
"date_publication" : {
    "type": "date",
    "format": "date_optional_time"
},
"date_modification" : {
    "type": "date",
    "format": "date_optional_time"
},
"authors" : {"type" : "keyword"},
"sources" : {"type" : "keyword"},
"categories" : {"type" : "keyword"},
"tags" : {"type" : "keyword"},
"title" : { "type" : "text" },
"drop" : { "type" : "text" },
"body" : { "type" : "text" },
"url_article" : { "type" : "keyword" },
```

Índice de Códigos Fuente B.1: Elementos básicos del *mapping* de artículos.

```
"image" : {
    "properties" : {
        "url": { "type" : "text" },
        "authorship" : { "type" : "text" },
        "description": { "type" : "text" }
    }
}
"has_tweet": { "type" : "boolean"},
"tweet_id" : { "type" : "keyword" },
"tweet_metrics" : {
    "type": "nested",
    "properties": {
        "retweets" : { "type": "integer" },
        "likes" : { "type": "integer" },
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        }
    }
}
```

Índice de Códigos Fuente B.2: Elementos *image* y *tweet_metrics* del *mapping* de artículos

```
"tokenization_es": {
    "type": "nested",
    "properties": {
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "text_source": { "type": "text" },
        "model": { "type": "text" },
        "words": { "type": "text" },
        "word_count": {
            "type": "nested",
            "properties": {
                "word": { "type": "text" },
                "count": { "type" : "integer"}
            }
        }
    }
}
```

Índice de Códigos Fuente B.3: Elemento *tokenization_es* del *mapping* de artículos

```
"en": {
    "type": "nested",
    "properties": {
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "text_source": { "type": "text" },
        "model": { "type": "text" },
        "text": { "type": "text" },
        "tokenization_en": {
            "type": "nested",
            "properties": {
                "model": { "type": "text" },
                "words": { "type": "text" },
                "word_count": {
                    "type": "nested",
                    "properties": {
                        "word": { "type": "text" },
                        "count": { "type" : "integer"}
                    }
                }
            }
        }
    }
}
```

Índice de Códigos Fuente B.4: Elemento *en* del *mapping* de artículos

```
"embeddings": {
    "type": "nested",
    "properties": {
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "text_source": { "type": "text"},
        "lang": { "type": "text"},
        "model": { "type": "text" },
        "embedding": { "type": "float" }
    }
}
```

Índice de Códigos Fuente B.5: Elemento *embeddings* del *mapping* de artículos

```
"knn" : {
    "type": "nested",
    "properties": {
        "article_ids" : { "type": "integer" },
        "text_source": { "type": "text"},
        "lang": { "type": "text"},
        "model" : { "type": "text" },
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "date_since" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "date_last" :  {
            "type": "date",
            "format": "date_optional_time"
        }
    }
}
```

Índice de Códigos Fuente B.6: Elemento *knn* del *mapping* de artículos

```
"polarity": {
    "type": "nested",
    "properties": {
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "text_source": { "type": "text" },
        "lang": { "type" : "text"},
        "model": { "type": "text" },
        "positive": { "type": "float" },
        "negative": { "type": "float" },
        "neutral": { "type": "float" },
        "compound": { "type": "float" }
    }
}
```

Índice de Códigos Fuente B.7: Elemento *polarity* del *mapping* de artículos

```
"subjectivity": {
    "type": "nested",
    "properties": {
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "text_source": { "type": "text" },
        "lang": { "type" : "text"},
        "model": { "type": "text" },
        "score": { "type": "float" }
    }
}
```

Índice de Códigos Fuente B.8: Elemento *subjectivity* del *mapping* de artículos

```
"preprocessed" : { "type" : "boolean" },
"event_id" : { "type" : "keyword" },
"has_knn": { "type" : "boolean"},
```

Índice de Códigos Fuente B.9: Elementos *flags* y *event_id* del *mapping* de artículos

B.2. *Mapping* de eventos

```
"snapshots": {
    "type": "nested",
    "properties": {
        "timestamp" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "date_since" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "date_last" :  {
            "type": "date",
            "format": "date_optional_time"
        },
        "clustering_model": {"type" : "text"},
        "new_keywords": {"type": "text"},
        "new_articles_id": {"type": "text"},
    }
}
```

Índice de Códigos Fuente B.10: Elementos básicos del *mapping* de eventos

```
"snapshots": {
    "type": "nested",
    "properties": {
        "accum_twitter" : {
            "properties": {
                "retweets" : {"type": "integer"},
                "likes" : {"type": "integer"}
            }
        },
        "accum_avg_polarity": {
            "properties": {
                "text_source": { "type": "text" },
                "lang": { "type" : "text"},
                "model": {"type": "text"},
                "positive": {"type": "float"},
                "negative": {"type": "float"},
                "neutral": {"type": "float"},
                "compound": {"type": "float"}
            }
        },
        "accum_avg_subjectivity": {
            "properties": {
                "text_source": { "type": "text" },
                "lang": { "type" : "text"},
                "model": {"type": "text"},
                "score": {"type": "float"}
            }
        },
        "accum_top20_word_count_drop_and_body": {
            "type": "nested",
            "properties": {
                "word": { "type": "text" },
                "count": { "type" : "integer"}
            }
        },
    }
}
```

Índice de Códigos Fuente B.11: Elementos acumulativos del *mapping* de eventos

```
"snapshots": {
    "type": "nested",
    "properties": {
        "title": {
            "properties": {
                "model": { "type": "text" },
                "lang": { "type" : "text"},
                "text": {"type": "text"},
                "polarity": {
                    "properties": {
                        "text_source": { "type": "text" },
                        "lang": { "type" : "text"},
                        "model": {"type": "text"},
                        "positive": {"type": "float"},
                        "negative": {"type": "float"},
                        "neutral": {"type": "float"},
                        "compound": {"type": "float"}
                    }
                },
                "subjectivity": {
                    "properties": {
                        "text_source": { "type": "text" },
                        "lang": { "type" : "text"},
                        "model": {"type": "text"},
                        "score": {"type": "float"}
                    }
                }
            }
        },
    }
}
```

Índice de Códigos Fuente B.12: Elemento *title* del *mapping* de eventos

```
"snapshots": {
    "type": "nested",
    "properties": {
        "summary": {
            "properties": {
                "model": { "type": "text" },
                "lang": { "type" : "text"},
                "text": {"type": "text"},
                "polarity": {
                    "properties": {
                        "text_source": { "type": "text" },
                        "lang": { "type" : "text"},
                        "model": {"type": "text"},
                        "positive": {"type": "float"},
                        "negative": {"type": "float"},
                        "neutral": {"type": "float"},
                        "compound": {"type": "float"}
                    }
                },
                "subjectivity": {
                    "properties": {
                        "text_source": { "type": "text" },
                        "lang": { "type" : "text"},
                        "model": {"type": "text"},
                        "score": {"type": "float"}
                    }
                }
            }
        }
    }
}
```

Índice de Códigos Fuente B.13: Elemento *summary* del *mapping* de eventos

```
"date_first_article": {
    "type": "date",
    "format": "date_optional_time"
},
"snapshots": {
    "type": "nested",
    "properties": {
        [...]
    }
}
```

Índice de Códigos Fuente B.14: Elemento *date_first_article* del *mapping* de eventos

B.3. *Mapping* de agrupaciones

```
"date_execution" :  {
    "type": "date",
    "format": "date_optional_time"
},
"date_since" :  {
    "type": "date",
    "format": "date_optional_time"
},
"date_last" :  {
    "type": "date",
    "format": "date_optional_time"
},
"events_ids": {"type" : "text"},
"n_articles": {"type" : "integer"},
"execution_time_in_sec" : {"type" : "float"},
"algorithm": { "type" : "text"},
"algorithm_simple": { "type" : "text"},
"algorithm_params": {
    "type" : "nested",
    "properties" : {
        "name" : "text",
        "value": "text"
    }
},
"algorithm_metrics": {
    "type" : "nested",
    "properties" : {
        "name" : "text",
        "value": "text"
    }
}
```

Índice de Códigos Fuente B.15: *Mapping* de agrupaciones

Anexo C

Experimentos previos

C.1. Experimentación previa con heurística

Configuración	Labels predicted
True labels	[0, 0, 0, 0, 0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4, 5, 5, 6, 6, 7, 7, 7, 7, 7, 7, 7, 8]
threshold=1 individual_tokens=False join_similar=True	[0, 0, 0, 0, 0, 0, 0, 0, 10, 11, 5, 4, 4, 1, 4, 4, 3, 3, 9, 2, 7, 9, 3, 7, 4, 2]
threshold=1 individual_tokens=True join_similar=True	[0, 0, 0, 0, 0, 0, 0, 0, 10, 11, 5, 1, 1, 1, 1, 4, 3, 3, 9, 2, 7, 9, 3, 7, 1, 2]
threshold=2 individual_tokens=False join_similar=True	**[0, 0, 0, 0, 0, 0, 0, 0, 8, 9, 10, 1, 1, 1, 1, 1, 4, 4, 11, 12, 7, 13, 4, 7, 1, 14]**
threshold=2 individual_tokens=False join_similar=False	[0, 0, 2, 0, 0, 2, 0, 0, 8, 9, 10, 1, 1, 3, 1, 1, 4, 4, 11, 12, 7, 13, 6, 7, 1, 14]
threshold=2 individual_tokens=True join_similar=True	**[0, 0, 0, 0, 0, 0, 0, 0, 8, 9, 10, 1, 1, 1, 1, 11, 4, 4, 12, 13, 7, 14, 4, 7, 1, 15]**
threshold=2 individual_tokens=True join_similar=False	**[0, 0, 2, 0, 0, 2, 0, 0, 8, 9, 10, 1, 1, 3, 1, 11, 4, 4, 12, 13, 7, 14, 6, 7, 1, 15]**
threshold=3 individual_tokens=False join_similar=False	[0, 0, 0, 0, 0, 2, 0, 0, 4, 5, 6, 1, 1, 7, 1, 1, 8, 9, 10, 11, 12, 13, 14, 15, 1, 16]
threshold=3 individual_tokens=False join_similar=True	[0, 0, 0, 0, 0, 0, 0, 0, 4, 5, 6, 1, 1, 7, 1, 1, 8, 9, 10, 11, 12, 13, 14, 15, 1, 16]
threshold=4 individual_tokens=False join_similar=False	[2, 3, 4, 5, 1, 1, 1, 6, 7, 8, 9, 0, 0, 10, 0, 11, 12, 13, 14, 15, 16, 17, 18, 19, 0, 20]
threshold=4 individual_tokens=False join_similar=True	[2, 3, 4, 5, 1, 1, 1, 6, 7, 8, 9, 0, 0, 10, 0, 11, 12, 13, 14, 15, 16, 17, 18, 19, 0, 20]
threshold=2 individual_tokens=False join_similar=True *Sin filtro de stopwords	[1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0]
threshold=5 individual_tokens=False join_similar=True *Sin filtro de stopwords	[0, 0, 0, 0, 0, 0, 0, 0, 6, 7, 8, 0, 0, 0, 0, 9, 0, 0, 10, 11, 12, 13, 0, 14, 0, 15]
threshold=7 individual_tokens=False join_similar=True *Sin filtro de stopwords	[3, 4, 5, 0, 0, 6, 7, 8, 9, 10, 11, 0, 0, 0, 0, 0, 12, 13, 14, 15, 16, 17, 18, 19, 0, 20]

Tabla C.1: Resultados de heurística para muestra de 27 artículos, bajo distintas configuraciones

C.2. Experimentación con modelos de traducción

C.2.1. Experimento con texto sobre inauguración de Convención Constitucional

Texto original: extracción de oraciones de varias fuentes

Los convencionales del pueblo Mapuche, la Lista del Pueblo, el PS y el FA **llamaron** a sus adherentes a caminar con ellos desde distintos puntos de la capital hasta el ex Congreso Nacional. El ministro del Interior indicó que a los constituyentes **"los pueden acompañar algunas personas**, siempre y cuando tengan su Pase de Movilidad o su permiso en **Comisaría Virtual**". **De cara a** la instalación de la Convención Constitucional, **a realizarse este domingo 4 de julio** en el ex Congreso Nacional, **al menos cuatro convocatorias están confirmadas** para los minutos previos a la ceremonia. Las **distintas marchas** están convocadas por el Partido Socialista (PS), el Frente Amplio (FA), la Lista del Pueblo y **los convencionales pertenecientes al pueblo Mapuche**. En el primer caso, los socialistas invitaron a **sus adherentes** a reunirse a las **9 de la mañana** en la estatua del ex presidente Salvador Allende, en la Plaza de la Constitución. Desde ese lugar **pretenden caminar** hasta el edificio del ex Congreso Nacional, ubicado en Compañía 1131.

El FA instó a sus **militantes y a la ciudadanía** a reunirse a una cicletada y caminata cultural. **Las personas se juntarán a las 08:30** en la Plaza Yungay y el destino será el ex Congreso. **En tanto, desde las 08:15 de la mañana**, la Lista del Pueblo marchará **desde Plaza Italia hasta Compañía**, mientras que los convencionales Mapuche tendrán una ceremonia en el Cerro Huelén (Santa Lucía).

Al respecto, el ministro del Interior, Rodrigo Delgado, indicó que los organizadores de las manifestaciones tuvieron reuniones de coordinación con las **autoridades correspondientes**, **como el intendente de Santiago**, Felipe Guevara.

Según el jefe de la cartera de Interior, "en esa reunión le **pedimos** información acerca de cuál era el propósito de esa caminata, de esa marcha, y en los cuatro casos (. . .) **ellos tienen convencionales constituyentes** que **de un punto determinado van a caminar** y se van a **aproximar** hacia el ex Congreso, tal como lo van a hacer el resto de los 155 convencionales".

Asimismo, el ministro Delgado indicó que "lo que tenemos **acá** son la aproximación de convencionales electos que van al ex Congreso. **Los pueden acompañar algunas personas**, siempre y cuando tengan su Pase de Movilidad o su permiso en **Comisaría Virtual**".

Texto final traducido nuevamente a español con modelo *opus-mt-en-es*

Los convencionales del pueblo mapuche, la Lista del Pueblo, el PS y la FA **convocaron** a sus adherentes a caminar con ellos desde diferentes puntos de la capital hasta el ex Congreso Nacional. El Ministro del Interior indicó que los constituyentes "**pueden estar acompañados por algunas personas**, siempre que tengan su Pase de Movilidad o permiso en la Estación Virtual **de Policía**". **Ante la instalación** de la Convención Constitucional, **que se celebrará el domingo 4 de julio** en el ex Congreso Nacional, **se confirman al menos cuatro convocatorias** para los minutos previos a la ceremonia. Las **diversas marchas** son convocadas por el Partido Socialista (PS), el Frente Amplio (FA), la Lista del Pueblo y **las convencionales del pueblo mapuche**. En el primer caso, los socialistas invitaron a **sus adeptos** a reunirse a **las 9.00 horas** en la estatua del ex Presidente Salvador Allende, en la Plaza de la Constitución. Desde ese lugar **tienen la intención de caminar** hasta el edificio del ex Congreso Nacional, ubicado en Compañía 1131.

FA instó a sus **militantes y ciudadanos** a reunirse para una bicicletada y una caminata cultural. **La gente se reunirá a las 08:30** en la Plaza Yungay y el destino será el ex Congreso. **Mientras tanto, a partir de las 08:15 horas**, la Lista Popular marchará **de Plaza Italia a Compañía**, mientras que los convencionales mapuches tendrán una ceremonia en Cerro Huelen (Santa Lucía).

En este sentido, el Ministro del Interior Rodrigo Delgado indicó que los organizadores de las manifestaciones tuvieron reuniones de coordinación con las **autoridades pertinentes**, **como el alcalde de Santiago**, Felipe Guevara.

Según el jefe de la cartera del Interior, "en esa reunión le **preguntamos** sobre el propósito de esa caminata, de esa marcha, y en los cuatro casos (...) **tienen constituyentes convencionales** que **desde cierto punto caminarán** y se **acercarán** al ex Congreso, así como lo harán el resto de los 155 convencionales".

Asimismo, el Ministro Delgado indicó que "lo que tenemos **aquí** es la aproximación de los convencionales electos que van al ex Congreso. **Pueden ser acompañados por algunas personas**, siempre y cuando tengan su Pase de Movilidad o su permiso en la **Estación Virtual de Policía**".

Traducción del modelo *opus-mt-es-en* del texto

The Mapuche people's conventionals, the People's List, the PS and the FA called on their adherents to walk with them from different points of the capital to the former National Congress. The Minister of the Interior indicated that the constituents "may be accompanied by some people, provided they have their Mobility Pass or permission in Virtual Police Station." In view of the installation of the Constitutional Convention, to be held on Sunday, July 4, at the former National Congress, at least four summonses are confirmed for the minutes prior to the ceremony. The various marches are called by the Socialist Party (PS), the Broad Front (FA), the People's List and the Mapuche people's conventional ones. In the first case, the socialists invited their adherents to meet at 9 a.m. in the statue of former President Salvador Allende, in the Plaza de la Constitución. From that place they intend to walk to the building of the former National Congress, located in Compañía 1131.

FA urged its militants and citizens to gather for a cycletada and cultural walk. People will gather at 08:30 in Yungay Square and the destination will be the former Congress. Meanwhile, from 08:15 a.m., the People's List will march from Plaza Italia to Compañía, while the Mapuche conventionals will have a ceremony at Cerro Huelen (Saint Lucia).

In this regard, Interior Minister Rodrigo Delgado indicated that the organizers of the demonstrations had coordination meetings with the relevant authorities, such as the mayor of Santiago, Felipe Guevara. According to the head of the Interior portfolio, "in that meeting we asked him about the purpose of that walk, of that march, and in the four cases (...) they have conventional constituents that from a certain point they will walk and approach the former Congress, just as they will do the rest of the 155 conventional ones."

Likewise, the Minister Delgado indicated that "what we have here are the approximation of elected conventionals that go to the former Congress. They can be accompanied by some people, as long as they have their Mobility Pass or their permission in Virtual Police Station." Four political parties in Chile have called on their followers to take part in a series of marches in Santiago on Sunday, July 4.

C.2.2. Experimento de noticia de CNN sobre inauguración de Convención Constitucional

Texto original

Durante la tarde de este lunes, en la previa de la sesión de la Convención Constitucional, convencionales electos por la Lista del Pueblo, Rodrigo Rojas Vade, Alejandra Pérez y Giovanna Grandón, participaron en una manifestación por **la liberación de los y las detenidas durante el 18 de octubre.** La movilización se realizó en la Plaza de Armas, donde familiares de los presos de la revuelta social llegaron con pancartas para visibilizar sus demandas. "Esta movilización está organizada por los grupos de apoyo, de familiares y amigos de presos políticos, a quienes siempre hemos acompañado, siempre estamos **acá**", explicó Rojas. **En la misma línea,** agregó que "son parte vital para nosotros, **porque alguno de los y las jóvenes que están presos pude ser yo**, muchas de ellas **están sin causa**, están sin un debido proceso. Para mí son parte importante, porque son de la manifestación de donde yo vengo, de la calle, **del pueblo que ha sido herido y torturado.** Nosotros estamos con ellos y los vamos a seguir apoyando".

Lee también: Autor del proyecto de indulto: "Es una deuda pendiente, **hay que buscar** una salida política".

Junto a esto, **detalló que** "es un acto simbólico **por una causa que ha sido invisibilizada**, hay mucha gente que ha intentando normalizar mediante los medios de comunicación la violencia sistemática a los Derechos Humanos y eso no puede pasar en un país que quiere comenzar con una democracia sana". Se espera que **cerca de las 15:00 horas comience la sesión de** la Convención Constitucional en el ex Congreso en Santiago, la que será **presidida** por Elisa Loncón. **Los y las constituyentes de Chile Vamos** se han expresado en contra del proyecto **y que se discuta en el órgano**. En la instancia, según informó el vicepresidente del organismo, Jaime Bassa, se discutirá la posibilidad de emitir una declaración conjunta sobre los presos durante el estallido social y del **proyecto de indulto que se tramita** en el Congreso[1].

[1]https://www.cnnchile.com/pais/rojas-vade-detenidos-18-o-apoyo_20210705/

Texto final traducido nuevamente a español con modelo *opus-mt-en-es*

Durante la tarde de este lunes, antes de la sesión de la Convención Constitucional, los órganos de tratados elegidos por la Lista Popular, Rodrigo Rojas Vade, Alejandra Pérez y Giovanna Grandón, participaron en una manifestación para **la liberación de las mujeres y los detenidos durante el 18 de octubre.** La movilización tuvo lugar en la Plaza de Armas, donde familiares de los presos de la revuelta social llegaron con pancartas para hacer visibles sus demandas. "Esta movilización está organizada por grupos de apoyo, familias y amigos de presos políticos, a quienes siempre hemos acompañado, siempre estamos **aquí**", explicó Rojas. **En el mismo sentido**, agregó que "son una parte vital de nosotros, **porque una de las jóvenes y mujeres que están encarceladas podría ser yo**, muchas de ellas **no tienen causa**, no tienen el debido proceso; para mí son una parte importante, porque son de la manifestación de donde vengo, de la calle, **de las personas que han sido heridas y torturadas;** estamos con ellas y seguiremos apoyándolas.

Lee también: Autor del proyecto de indulto: "Es una deuda pendiente, **tenemos que buscar** una salida política"

Junto con esto, **explicó que** "es un acto simbólico **por una causa que ha sido invisible**, hay muchas personas que han tratado de normalizar la violencia sistemática a los derechos humanos a través de los medios de comunicación y que no puede suceder en un país que quiere comenzar con una democracia sana". Se espera que **alrededor de las 3 p.m. la sesión de** la Convención Constitucional **comience** en el ex Congreso de Santiago, que será **presidido** por Elisa Loncon. **Los electores de Chile Vamos** se han expresado en contra del proyecto **y que será discutido en el órgano**. En la instancia, según el vicepresidente de la agencia, Jaime Bassa, se discutirá la posibilidad de emitir una declaración conjunta sobre los presos durante la explosión social y el **proyecto de ley de indulto que se está procesando** en el Congreso...

Texto intermedio traducido con modelo *opus-mt-es-en*

During the afternoon of this Monday, prior to the session of the Constitutional Convention, treaty bodies elected by the People's List, Rodrigo Rojas Vade, Alejandra Pérez and Giovanna Grandón, participated in a demonstration for the release of those and women detained during October 18. The mobilization took place in the Plaza de Armas, where relatives of the prisoners of the social revolt arrived with banners to make their demands visible. "This mobilization is organized by support groups, families and friends of political prisoners, whom we have always accompanied, we are always here," explained Rojas. In the same vein, he added that "they are a vital part of us, because one of the young women and women who are imprisoned could be me, many of them are without cause, they are without due process. For me they are an important part, because they are from the demonstration where I come from, from the street, from the people who have been wounded and tortured. We are with them and we will continue to support them."

Also read: Author of the pardon project: "It is a pending debt, we have to look for a political way out"

Along with this, he explained that "it is a symbolic act for a cause that has been invisible, there are many people who have tried to normalize systematic violence to human rights through the media and that cannot happen in a country that wants to start with a healthy democracy." It is expected that around 3 p.m. the session of the Constitutional Convention will begin at the former Congress in Santiago, which will be presided over by Elisa Loncon. The constituents of Chile Vamos have expressed themselves against the project and that it will be discussed in the organ. In the instance, according to the vice president of the agency, Jaime Bassa, the possibility of issuing a joint statement on the prisoners during the social explosion and the pardon bill that is being processed in Congress will be discussed..

C.3. Experimentación con resumidores

C.3.1. Generación resúmenes por evento de inauguración de Convención Constitucional

Texto otiginal: extracción de oraciones de varias fuentes

Los convencionales del pueblo Mapuche, la Lista del Pueblo, el PS y el FA **llamaron** a sus adherentes a caminar con ellos desde distintos puntos de la capital hasta el ex Congreso Nacional. El ministro del Interior indicó que a los constituyentes **"los pueden acompañar algunas personas**, siempre y cuando tengan su Pase de Movilidad o su permiso en **Comisaría Virtual**". **De cara a** la instalación de la Convención Constitucional, **a realizarse este domingo 4 de julio** en el ex Congreso Nacional, **al menos cuatro convocatorias están confirmadas** para los minutos previos a la ceremonia. Las **distintas marchas** están convocadas por el Partido Socialista (PS), el Frente Amplio (FA), la Lista del Pueblo y **los convencionales pertenecientes al pueblo Mapuche**. En el primer caso, los socialistas invitaron a **sus adherentes** a reunirse a las **9 de la mañana** en la estatua del ex presidente Salvador Allende, en la Plaza de la Constitución. Desde ese lugar **pretenden caminar** hasta el edificio del ex Congreso Nacional, ubicado en Compañía 1131. El FA instó a sus **militantes y a la ciudadanía** a reunirse a una cicletada y caminata cultural. **Las personas se juntarán a las 08:30** en la Plaza Yungay y el destino será el ex Congreso. **En tanto, desde las 08:15 de la mañana**, la Lista del Pueblo marchará **desde Plaza Italia hasta Compañía**, mientras que los convencionales Mapuche tendrán una ceremonia en el Cerro Huelén (Santa Lucía). **Al respecto, el ministro del Interior, Rodrigo Delgado**, indicó que los organizadores de las manifestaciones tuvieron reuniones de coordinación con las **autoridades correspondientes**, **como el intendente de Santiago**, Felipe Guevara. Según el jefe de la cartera de Interior, "en esa reunión le **pedimos** información acerca de cuál era el propósito de esa caminata, de esa marcha, y en los cuatro casos (. . .) **ellos tienen convencionales constituyentes** que **de un punto determinado van a caminar** y se van a **aproximar** hacia el ex Congreso, tal como lo van a hacer el resto de los 155 convencionales". Asimismo, el ministro Delgado indicó que "lo que tenemos **acá** son la aproximación de convencionales electos que van al ex Congreso. **Los pueden acompañar algunas personas**, siempre y cuando tengan su Pase de Movilidad o su permiso en **Comisaría Virtual**".

Output* de *bert2bert_ shared-spanish-finetuned-summarizatio

El ministro del Interior dice que a los constituyentes " los pueden acompañar algunas personas, siempre y cuando tengan su pase de Movilidad o su permiso en Comisaría Virtual "

Output* de *bert2bert_ shared-spanish-finetuned-muchocine-review-summarization

Una declaración de principios, pero más bien, una declaración de intenciones...

Output* de *mt5-small-spanish-summarization

los convencionales del pueblo mapuche, el pp y el gobierno llama

Output* de modelo *google/pegasus-xsum

Cuatro partidos políticos en Chile han llamado a sus seguidores a participar en una serie de marchas en Santiago el domingo 4 de julio.

Output* de modelo *facebook/bart-large-cnn

Los convencionales del pueblo mapuche, la Lista del Pueblo, el PS y la FA llamaron a sus adherentes a caminar con ellos desde diferentes puntos de la capital. El Ministro del Interior indicó que los constituyentes "pueden estar acompañados por algunas personas, siempre que tengan su Pase de Movilidad o permiso en la Estación Virtual de Policía".

Anexo D

Resultados de agrupación del 23 al 25 de febrero de 2022 mediante heurística

D.1. Eventos sobre el conflicto ruso-ucraniano

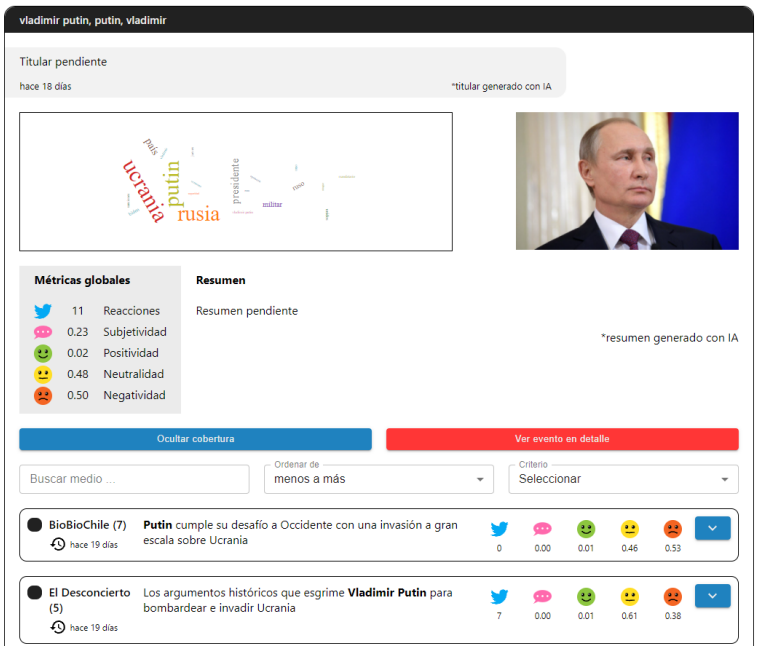

Figura D.1: Evento sobre conflicto ruso-ucraniano #2: 51 artículos

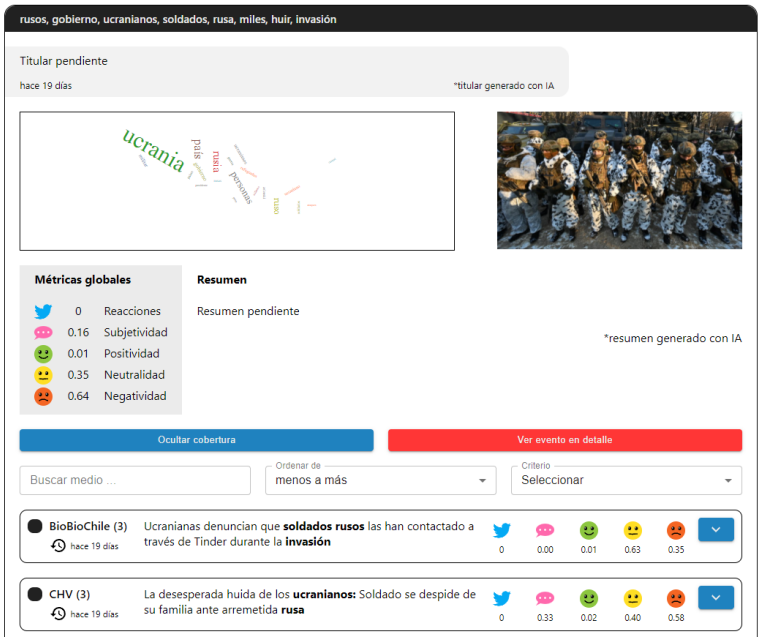

Figura D.2: Evento sobre conflicto ruso-ucraniano #3: 13 artículos

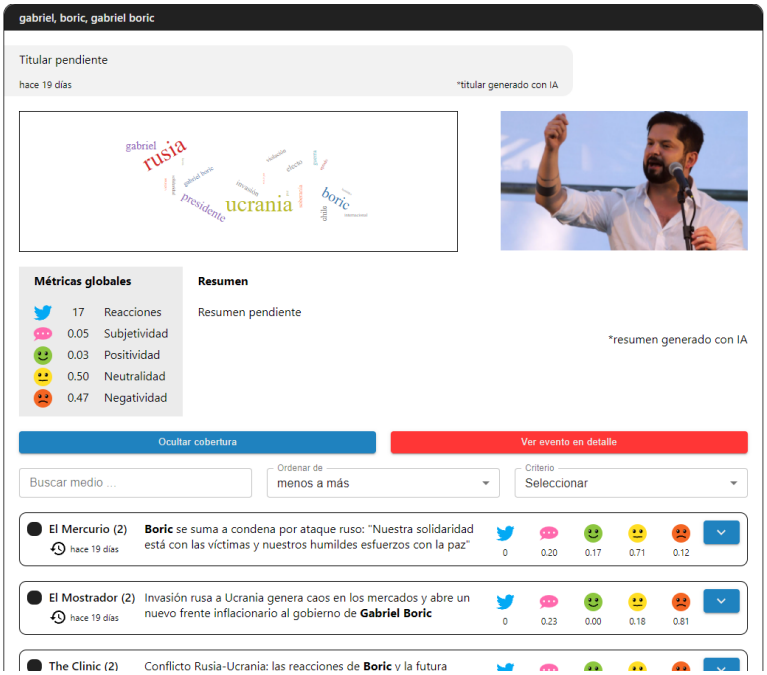

Figura D.3: Evento sobre conflicto ruso-ucraniano #4: 16 artículos

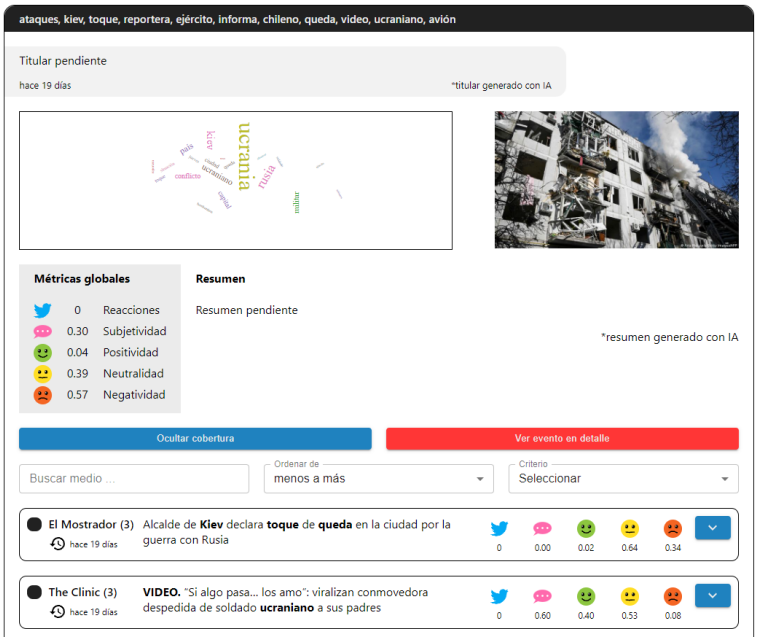

Figura D.4: Evento sobre conflicto ruso-ucraniano #5: 13 artículos

Anexo E

Entrevistas a usuarios

E.1. Perfiles de los entrevistados

#	**Nombre**	**Nacionalidad**	**Profesión u oficio**
1	Mariana Vargas	Argentina	PhD in Computer Science at Université de Lille, 2021. Master's degree in Computer Science, UNC, 2013.
2	Gustavo Donat	Chilena	Periodista de CNN Chile y Radio María. Trabajo esporádico en La voz de los que sobran.
3	Anónimo	Anónimo	Pedagogía en Español como lengua extranjera. Grado Superior en Filología. Asesoramiento Educativo.
4	Francisco Guerra	Chilena	Periodista y Licenciado en Comunicación Social de la Universidad Diego Portales, 2021.
5	Bryan Bizarro	Chilena	Ingeniero Civil Eléctrico, Universidad de Chile, 2021.
6	Daniel Bustos	Chilena	Abogado, Universidad Católica del Norte, 2016.
7	Paulina Bravo	Chilena	Pedagogía en Lenguaje, Universidad Católica del Maule, 2020.
8	Andrea Rodríguez	Chilena	Magíster en Historia de América, Universidad de Chile, 2004. Psicología, Universidad de Chile, 2010.
9	Catalina Domínguez	Chilena	Estudiante de Ingeniería Civil Eléctrica, Universidad de Chile.
10	Cristian Sánchez	Chilena	Estudiante de Ingeniería Civil en Computación, Universidad de Chile.
11	Cristobal Muñoz	Chilena	Periodista y Licenciado en Comunicación Social de la Universidad Diego Portales, 2021.
12	Dante Mardones	Chilena	Estudiante de Ingeniería Civil en Computación, Universidad de Chile.
13	Eva Ugartechea	Mexicana	Ingeniería Industrial y de sistemas, Tecnológico de Monterrey, 2019.
14	Francys Pinto	Chilena	Estudiante de Ingeniería Civil Eléctrica, U. de Chile.
15	Gabriela Bravo	Chilena	Educadora de párvulos mención en matemáticas, Universidad Católica del Maule, 2018.
16	Isidora Frías	Chilena	Trabajo Social, UTEM, 2021.
17	Johnny Bolados	Chilena	Backend engineer, Cornershop by Uber.
18	Jorge Garrido	Chilena	Ingeniero Civil Mecánico, Universidad de Chile, 2021.
19	José Pacheco	Chilena	Estudiante de Ingeniería en Computación, Universidad de Chile.
20	Valeria Hermosilla	Chilena	Psicóloga. Universidad Autónoma de Chile, 2021.
21	Vicente Rojas	Chilena	Estudiante de Ingeniería Civil en Computación, Universidad de Chile.
22	Magdalena Saldaña	Chilena	Assistant Professor, Universidad Católica de Chile. Researcher, Instituto Milenio Fundamento de los Datos.
23	Francisca Canseco	Chilena	Periodista en rol de producción, Radio Cooperativa.
24	Daniel Saavedra	Chilena	Psicólogo y académico de la Universidad de Chile.

Tabla E.1: Perfiles de personas entrevistadas

E.2. Entrevistas de la 1 a la 10

La entrevistada #1, PhD en Ciencias de la Computación, reconoce que plataformas como Google le muestra resultados que es probable que lea, y no necesariamente, posturas antagónicas a la propia. Señala que la plataforma abre perspectivas, da la posibilidad de conocer otras posturas y que es amigable, fácil de usar e intuitiva, a pesar de sugerir algunas mejoras en la visualización. Encuentra el trabajo "buenísimo y ambicioso". Añade que es una herramienta que invita a la curiosidad por informarse y contrastar los sesgos propios con los del sistema. Por último, indica que usaría una plataforma así para informarse sobre el día a día en Argentina.

El entrevistado #2, periodista de CNN Chile, resalta que la aplicación es muy útil para medios de comunicación, más que para un usuario común. Que sirve para analizar la competencia, el contenido propio y buscar información que quizá no está en Google, o que es limitada por sus filtros. Señala que la plataforma sirve como biblioteca digital, que sirve para elaborar pautas y notas. Advierte una baja usabilidad y que la interfaz podría ser más inclusiva para personas con discapacidades. Resalta bastante que no utilizaría un medio independiente para informarse primero sobre una noticia, porque fallan mucho en verificar fuentes y carecen de rigurosidad. Pero que sí los usaría para construir una versión de un hecho distinta y constatar versiones. Por último, abre puertas para usar un producto como este en áreas diferentes como por ejemplo, artículos académicos o publicaciones científicas.

El entrevistado #3, filólogo, indica un interés profundo por la información. Dicho esto, señala no conocer ninguna herramienta como esta, con tanto nivel de detalle y tantos filtros. Le parecía una herramienta más profesional, para periodistas o instituciones, que para usuarios masivos. Añade que quizás, el público general, no tiene interés por contrastar información, o bien, que carece de educación sobre los conceptos de neutralidad o subjetividad. En esa línea, dice que haría uso de la aplicación todos los días, pero que quizás el público general no. Al mismo tiempo ayudaría a las personas a tener una visión clara de un evento y a elaborar conclusiones propias sobre un hecho, o bien, a enseñar estos conceptos por medio de la aplicación.

Por otro lado, señala que la herramienta es una muy buena aproximación en lo que pretende hacer, que funciona súper bien, con un *layout* amigable, va en la dirección correcta: es innovadora e interesante. Dice que los periodistas están expuestos al escrutinio público y deberían estar sujetos a una ética ciudadana. Admite que confiaría más en sus propios sesgos, que los sesgos del sistema, pero que le daría oportunidad de lectura a medios que no lee usualmente. Por último, da énfasis en que es difícil enseñar los conceptos de subjetividad y neutralidad a los humanos, y que en consecuencia, es aún más difícil enseñar esto a una máquina.

El entrevistado #4, periodista, destaca que es una herramienta útil para personas que trabajan en la prensa o en agencias, ya que les permite hacer análisis del contenido propio y de la competencia, pero que también es útil para estudiantes universitarios de periodismo. En forma personal, señala que esta herramienta le ahorraría mucho tiempo al momento de buscar información, porque el método manual que tiene actualmente utilizando Google es bastante demoroso. Destaca que las noticias aparezcan agrupadas por medio, que el *layout* de eventos es fácil de utilizar y que evite información redundante. Valora que haya medios alternativos que, generalmente, ignora, y da hincapié en que este proyecto podría aliarse con entidades de *fact checking* para alertar a usuarios cuando la información es sospechosa.

El entrevistado #5, ingeniero eléctrico, critica que redes sociales como Twitter sólo le muestran cosas "bonitas" y que le gustaría ver cómo piensa el resto de las personas. Como usuario habitual de Google News, dice que emplearía esta herramienta como complemento, para primero centrarse en los hechos mediante los filtros disponibles y luego para buscar artículos con más o menos subjetividad para formar su propia opinión; Que los filtros le ayudan a ver "las dos caras de la moneda" respecto a un evento. Cataloga a la aplicación como profesional, intuitiva, y no difícil de usar, pero al mismo tiempo, se muestra preocupado respecto de los sesgos que adquieren los modelos de ML a través de los datos de entrenamiento. Por último, cree que las personas usarían la aplicación para ver información de las últimas 24 horas, y que otro tipo de personas la usarían para buscar información histórica.

El entrevistado #6, abogado, señala que esta aplicación le ahorraría tiempo en su trabajo, al estructurar la información de manera más organizada y con más filtros que herramientas Google, o El Mercurio legal. Dice reiteradas veces que este sistema es útil, interesante, completo y efectivo. Señala que actualmente tiene información limitada por canales como la televisión, la radio o historias de Instagram, y que esta aplicación "abre información o conocimiento que podría no haber accedido". Le interesa conocer medios que no conoce actualmente, y medios que difunden discurso de odio para reprocharlos. Destaca que esta herramienta le permite acceder a más información para elaborar una opinión más seria. Finalmente, destaca las métricas globales de los eventos porque es la primera vez que ve que una noticia se puede ordenar en base a su neutralidad, negatividad, positividad y reacciones.

La entrevistada #7, profesora de lenguaje de enseñanza básica y media, plantea reiteradas veces el uso de esta herramienta en el plano educativo. Para el uso de docentes que quieren buscar textos no literarios, discernir información, y luego trabajar en clases, o bien para realizar investigación. La idea es usarla también para enseñar los conceptos de subjetividad, objetividad y separar hechos de opiniones, que bajo un lenguaje complejo, a veces se hace difícil, pero evitando titulares escandalosos: la plataforma facilita esta labor. También, sirve para contrastar información proveniente de medios, con la información proveniente de redes sociales como Twitter.

A modo personal, señala que esta herramienta le sirve para conocer nuevos medios, y de visualizarlos en forma ordenada, ya que menciona que plataformas como Google le muestran miles de resultados y con sub-temas de lo que busca. El filtro de discurso de odio le sería útil porque menciona que trasciende la tendencia política.

Indica que hay dos tipos de lectura: global y analítica. La plataforma facilita tanto la lectura global, mediante la agrupación de eventos, y la lectura analítica, al facilitar los *links* de cada artículo. Finaliza "Que por favor salga la plataforma y que esto lo lleves directamente a mi colegio e instruyan a los profes' porque está buenísima. Está muy buena".

La entrevistada #8, historiadora y psicóloga, comenta que una herramienta así le hubiera servido muchísimo cuando hizo su tesis de historia, ya que el proceso que tuvo que realizar fue bastante engorroso y demoroso. Señala que una plataforma así le serviría para generar material para sus clases de historia y analizar eventos como el conflicto mapuche, de manera histórica. Critica el tamaño de la letra en la interfaz y que no es amigable para usuarios con alguna discapacidad. Valora mucho las imágenes en la Vista de agrupaciones, e indica que este *layout* le permitiría encontrar información de su interés rápidamente.

Figura E.1: Captura de la herramienta digital que le permitió a Andrea Rodríguez contrastar medios televisivos para un caso particular.

Comenta que hace uso de una herramienta digital para contrastar medios de televisión en vivo[1] (Ver Figura E.1). También señala, que desde el área de las ciencias sociales, es probable que el sólo término de "inteligencia artificial" no tenga mucha confiabilidad. Indica que usaría una aplicación así todos los días para informarse, y valora que, en contraste con Twitter, acá no se mezclan comentarios de personas naturales con medios de comunicación. Finalmente, cataloga la aplicación como "un aporte social y cultural muy bueno".

La entrevistada #9, estudiante de ingeniería eléctrica, señala que usaría la plataforma como complemento de plataformas de *fact-checking*, para validar noticias en el plano nacional. El resumen y el titular le llaman la atención del *layout* de eventos, pero con claras críticas a la interfaz también. Plantea un escenario interesante: cuando un evento es abordado por medios de un sólo sector, por ejemplo, plantea el tema de "La tierra es plana". ¿Cómo sería la elaboración del resumen y titular en ese caso?. A lo cual se responde indicando que si el *input* de texto es sesgado, el resumen muy probablemente también lo sea. Finaliza dando algunas ideas de personalización de la aplicación y planteando el desafío de agregar medios que informen por Instagram o TikTok[2].

El entrevistado #10, estudiante de ingeniería en computación, encuentra bastantes detalles en términos de interfaz y usabilidad del sistema. A pesar de esto, señala que une herramienta como esta le sirve para no perder tiempo viendo la misma información muchas veces, ya que en

[1]https://www.pslabs.cl/tele.html
[2]https://www.tiktok.com/about

sus palabras "Usar Google es exhaustivo y toma mucho tiempo". Añade que le ayuda a distinguir la polaridad de los medios, y distinguir cuando estos tienen subjetividad hacia un tema que no deberían. Por último, está abierto a ver medios que no conoce y recalca que "es una muy buena plataforma para ver noticias".

De los primeros diez entrevistados hay varias ideas destacables y que es prudente enumerar:

1. Google muestra resultados sesgados, y muchas veces no de posturas antagónicas a la propia.
2. La aplicación se podría utilizar en el contexto periodístico de Argentina.
3. La plataforma abre perspectivas, mediante su gran abanico de medios que muchos no conocían.
4. Existen muchos detalles en la interfaz a nivel visual y de experiencia de usuario, es decir, la usabilidad del sistema es mala.
5. La plataforma invita a informarse y contrastar los sesgos propios con los del sistema.
6. Muchos la usarían para informarse en el día a día.
7. Existe un caso de uso útil para los medios de comunicación, para analizar la competencia, generar pautas y notas.
8. Sirve como biblioteca digital.
9. Abre las posibilidades a usarlo para artículos académicos o publicaciones científicas.
10. Varios usuarios señalan que parece una herramienta más profesional que de uso masivo.
11. La plataforma ayuda a que los usuarios elaboren conclusiones propias sobre un hecho.
12. El *layout* de eventos es amigable.
13. Todos los entrevistados señalan que no conocían una herramienta como esta, que lo que más se acerca son los *trending topics* de Twitter o Google News.
14. Advierten sobre los sesgos de los modelos de inteligencia artificial, y de cómo estos podrían ser reprochados por los usuarios.
15. Sirve para ver "las dos caras de la moneda" respecto a un hecho.
16. Creen que serviría más para informar sobre hechos ocurridos recientemente, para usuarios masivos.
17. Señalan que una herramienta así sirve para ahorrar tiempo al buscar información más precisa, en contextos de investigación o al buscar temas de interés.
18. Plantean su uso en educación.
19. Sirve para estar informado en forma global sobre un evento, pero también para realizar un análisis más específico.
20. Utilizarían esta aplicación en complemento a sus propios métodos para informarse.
21. Sirve para buscar información más objetiva primero y luego abrirse a otras perspectivas.

E.3. Entrevistas de la 11 a la 20

Al entrevistado #11, periodista, le parece que la plataforma es como una hemeroteca capaz de cumplir intereses para la investigación periodística académica, para las marcas y agencias en cuanto a comunicación estratégica, y también para personas que quieran informarse. Señala que una herramienta así sólo se puede obtener en privado y que es importante para estudios de periodismo o de comunicación. Además, dice que herramientas como Google posee hartos sesgos debido a su algoritmo de recomendación y que siempre salen los medios más famosos, señala "Acá los medios más chicos reciben un espacio".

Por otro lado, enfatiza en que las métricas de polaridad y subjetividad tienen un fin más académico que masivo, y duda que personas naturales las usen para ordenar información. Además, que muchos elementos, entre ellos las métricas globales, estarían destinados más para investigación, que para informar. A pesar de esto, señala que estás métricas pueden servir para romper el esquema de "todas las noticias de la tele son negativas", y que medir la subjetividad es un avance muy valioso, pero que debería ser bien explicado en el sitio, ya que para él es súper importante la transparencia. Finaliza diciendo que utilizaría esta aplicación para informarse en la mañana y en la noche.

El entrevistado #12, estudiante de ingeniería en computación, señala que para eventos como el estallido social de Chile "se llena de información de un solo lado", y que mediante esta aplicación se pueden contrastar varias fuentes, que es lo que necesita para informarse. Dice además que generalmente se informa por Google de eventos que le interesan, pero que no llega a medios mas chicos porque no pasa de la primera página.

La entrevistada #13, ingeniera comercial mexicana, indica que una plataforma así serviría bastante en el contexto mexicano, ya que ese país es el primero en los *rankings* de periodistas asesinados. Además, señala que Televisa, el medio de televisión más grande, concentra una gran parte del poder informativo del país y es difícil ver otras perspectivas. Además, admite que las redes sociales sólo le muestran contenido de su preferencia.

Le da bastante importancia al filtro de subjetividad para encontrar medios que quizás desconoce y añade que un filtro de *fake news* le sería bastante útil, porque "hoy en día hay mucha noticia falsa". Usaría la aplicación sólo para buscar eventos que sean de su real interés y no a diario.

La entrevistada #14, estudiante de ingeniería eléctrica, cree que la mayoría de las noticias actuales son trágicas y que los filtros del sistema le sirven para variar el estilo de noticias, ya que se siente "agobiada" por tantas noticias malas. También, señala que generalmente se informa desde los medios más grandes y le queda la duda si se está informando en forma sesgada o no. Dice que la herramienta sería útil para personas que les gusta tener distintas perspectivas. Un filtro de tendencia política lo encuentra innecesario porque ya tiene un criterio propio, pero uno de probabilidad de *fake news*, o de predicción de discurso de odio lo encuentra útil.

La entrevistada #15, profesora de párvulos, señala que utilizaría la herramienta para investigar de cosas que le interesen más, y añade que sería más fácil y rápido usando esta herramienta debido a la gran cantidad de filtros que existen. Dice que es una aplicación bien completa y que hay un gran abanico de medios que no conocía.

La entrevistada #16, egresada de trabajo social, cree que al informarse sobre una noticia se dejan muchos medios de lado, y cree que esto a veces es una estrategia o bien simplemente sucede. Dice que, usar los filtros para sólo ver noticias positivas o negativas esta bien por salud mental, pero al mismo tiempo uno queda ajeno de las situaciones de la vida real. Dice que hacer investigación por Google es una "lata" porque da un rango de tiempo limitado. Agrega que hay software privado para encontrar información más fácil, pero desconoce el nombre.

Utilizaría la herramienta para centrarse en lo neutral y luego en lo independiente. Le daría una oportunidad a medios que desconoce porque dice que se están esforzando para ser conocidos. Agrega "Es importante dar espacio y cabida a prensa independiente". Por último, describe al *layout* de eventos como amigable y ordenado, y que usaría una aplicación todos los días, porque le gusta estar informada.

El entrevistado #17, *software engineer*, resume su postura en la siguiente frase "Tienes que plantear mejor el público objetivo al que quieres llegar, y mejorar la interfaz también, pero me parece una iniciativa súper buena". Apunta a los medios de comunicación o las personas naturales como público objetivo. Además, señala que es una aplicación que puede aplicarse a nivel mundial también, y que sirve para contrastar posturas opuestas en tiempos de campañas políticas. Critica bastante las métricas globales, porque dice que sería difícil que un público masivo las entienda sin mayor explicación.

El entrevistado #18, ingeniero mecánico, cree que es una herramienta muy útil, pero tiene sus dudas si el ciudadano común vaya a buscar las noticias con los filtros, porque sólo quiere informarse de lo último que está pasando. Dice que puede servir para periodistas o personas que hagan investigación. Además, señala que usaría más el filtro de subjetividad que los de polaridad, y que sacar conclusiones de subjetividad y polaridad a través de Google sería más lento. Dice que buscar artículos más objetivos sería más rápido con esta plataforma.

Un filtro de *fake news* le sería muy útil porque dice que hay gente que cae en estas noticias sin darse cuenta, pero un filtro de clasificación de tendencia política "ensuciaría el sistema" porque se imagina que lo que se trata de hacer es llevar a una persona, independiente de su posición, todas las opiniones posibles. Con ese filtro se volvería al mismo sesgo que se busca evitar. Finalmente, señala que la interfaz podría mejorar bastante.

El entrevistado #19, estudiante de ingeniería en computación, dice que la herramienta le ayuda a tener una visión más general de la noticia y tener más puntos de vista, y medios que no conocía. Y señala como punto importante, que un filtro de tendencia política serviría para que los usuarios se centren en ciertos medios únicamente y que el sesgo propio se reitere.

La entrevistada #20, psicóloga, señala que es una propuesta súper interesante desde el punto de vista de salud mental, ya que a veces hay personas que sólo se quieren informar, evitando artículos morbosos, y que con los filtros se puede llegar de mejor forma a artículos más neutros o menos negativos. Además, señala que la tendría como primera plataforma para informarse porque lo encuentra interesante. Por último, añade que existan medios regionales en la plataforma es un beneficio para ellos, porque generalmente son desconocidos.

De estos entrevistados, hay varias ideas nuevas referentes a la plataforma:

1. Sirve para la investigación periodística académica.
2. Es útil para las marcas y agencias en cuanto a la comunicación estratégica.
3. Varios usuarios usarían la aplicación, pero sólo para informarse sobre temas que sean de su interés.
4. Otros usuarios lo utilizarían como "un medio de comunicación más".
5. Es útil para contrastar varias fuentes en tiempos de crisis.
6. Se podría utilizar en el contexto preiodístico de México.
7. Un filtro de tendencia política podría ser contraproducente.
8. Sirve para romper el esquema de "todas las noticias son trágicas".
9. Un filtro de *fake news* sería bastante útil, porque las personas no se dan cuenta cuando caen en ellas.
10. Existe una aplicación directa en el contexto de salud mental, para evitar noticias extremadamente negativas.

E.4. Entrevistas de la 21 a la 24

El entrevistado #21, estudiante de ingeniería en computación, usaría la aplicación como un medio de comunicación más. Destaca que la aplicación le ayuda a distinguir entre "campaña del terror, o todo esta bien", en sus palabras "cuáles son los medios más subjetivos". Dice que el *layout* de eventos le ayuda a tener una idea general de las noticias, y al mismo tiempo, buscar de forma específica artículos que sean de su interés. Añade que los filtros de subjetividad y negatividad le ayudan a "atacar lo que son las *fake news*". Por último, da varias sugerencias para mejorar la interfaz y la experiencia de usuario.

La entrevistada #22, periodista, e investigadora del IMFD, indaga bastante en cuáles son los modelos utilizados para medir la subjetividad y la polaridad en texto, y de las funcionalidades generales del sistema. Encuentra la plataforma una muy buena herramienta para investigación periodística, e incluso menciona que sirve para ejemplificar la "teoría del encuadre" (*framming* en inglés) [88]. Le gustaría mostrar este tipo de herramientas en clases, y también advierte que es muy posible que existan casos donde el sesgo no es capturable ni por palabras positivas o negativas, ni por una métrica de subjetividad, sino más bien, por el mensaje mismo que se transmite.

La entrevistada #23, periodista del área de producción de Radio Cooperativa, comienza preguntando si es que la aplicación tiene algún nicho específico, porque dice que hay bastantes personas que no se manejan mucho con la tecnología y que quizás no usarían los filtros del sistema. Que para este tipo de personas les es más fácil informarse por redes sociales, porque al final terminan siendo fuentes de información.

Señala que para el periodismo se puede volver muy útil, para preparar entrevistas, buscar información reciente e histórica y enfocado en periodistas que trabajan en la web, porque sirve para profundizar en contenido. Lo que más le llama la atención del *layout* de eventos es el resumen pero desconfía bastante, ya que debería validar cada hecho que señalan, y le produce aún más desconfianza que sean producidos por una inteligencia artificial.

Destaca bastante el tema de la descentralización de la información y cómo ella le da espacios en su día a día a medios regionales, pero que hay un largo camino por delante. Y por esa razón sí daría oportunidad de lectura a medios regionales en la plataforma. Por último, menciona que desde el lado profesional, un filtro de tendencia política no le sería útil porque ya tiene un criterio propio.

Finalmente, el entrevistado #24, psicólogo, señala que "esta herramienta pudiera ser súper útil ante el evidente sesgo comunicacional está mediado por factores políticos y económicos. Para saltarse el sesgo político-económico de los de los grandes medios de comunicación, que son los que más nos llegan". Advierte que en el país existe una concentración del poder y que la información de los medios muchas veces cumple los intereses de las personas más adineradas.

No cree que Google le permita cumplir esta labor, porque señala que Google tiene sus propios sesgos. Dice, "Yo creo que es bien útil no sólo para organizar la información del día a día, sino también para uno poder ir estableciendo ciertos filtros a raíz de ciertos sesgos informativos". Por ejemplo, menciona que esta plataforma le permitiría identificar los medios que dan bastante hincapié en la delincuencia y brindan mayor sensación de inseguridad al informar.

Por último, señala "Hay que pensar que la burbuja informativa que estaba mencionando va muy de la mano con un monopolio pensando en la desigualdad de las fuerzas, de los medios de información". Y que haría uso una aplicación así para esclarecer conflictos donde hay muchos interlocutores, o bien, utilizaría los filtros para identificar medios que emiten juicios de odio hacia minorías, porque justamente él trabaja con inmigrantes, minorías y disidencias sexuales.

De esta forma, a partir de estos últimos entrevistados, se añaden las siguientes ideas centrales:

1. Los usuarios creen que esta plataforma les ayuda a verificar la veracidad de una noticia.
2. Sirve además, en contextos de clases universitarias de periodismo.
3. Advierten que no es claro el nicho de usuarios objetivo.
4. Advierten que quizás es una plataforma compleja para un público masivo.
5. Existe una desconfianza hacia el modelo resumidor de eventos.
6. Esta herramienta es un aporte en el tema de descentralización de la información.
7. Es una herramienta útil ante los sesgos comunicacionales político-económicos del país.
8. Sirve para desarrollar filtros propios al momento de informarse.
9. Sirve para identificar medios que discriminen a minorías.