PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

Escuela de Posgrado

Agrupamiento de textos basado en la generación de Embeddings

Tesis para obtener el grado académico de Magíster en Informática con mención en Ciencias de la Computación que presenta:

Anthony Wainer Cachay Guivin

Asesor:

Dr. Cesar Armando Beltran Castañon

Lima, 2022

Resumen

Actualmente, gracias a los avances tecnológicos, principalmente en el mundo de la informática se logra disponer de una gran cantidad de información, que en su mayoría son una composición de signos codificados a nivel computacional que forman una unidad de sentido, como son los textos. Debido a la variabilidad y alta volumetría de información navegable en internet hace que poder agrupar información veraz sea una tarea complicada. El avance computacional del lenguaje de procesamiento natural está creciendo cada día para solucionar estos problemas.

El presente trabajo de investigación estudia la forma como se agrupan los textos con la generación de Embeddings. En particular, se centra en usar diferentes métodos para aplicar modelos supervisados y no supervisados para que se puedan obtener resultados eficientes al momento de toparse con tareas de agrupamiento automático.

Se trabajó con cinco Datasets, y como resultado de la implementación de los modelos supervisados se pudo determinar que el mejor Embedding es FastText implementado con Gensim y aplicado en modelos basados en boosting. Para los modelos no supervisados el mejor Embedding es Glove aplicado en modelos de redes neuronales con AutoEncoder y capa K-means.

Palabras claves: Lenguaje de procesamiento natural, Análisis de modelos, Inteligencia artificial, Datasets con texto, Datasets en español, Agrupamiento, Embeddings, modelos supervisados y no supervisados.

Abstract

Nowadays, thanks to technological advances, mainly in the world of information technology, a large amount of information is available, most of which is a composition of signs encoded at a computational level that form a unit of meaning, such as texts. Due to the variability and high volume of navigable information on the Internet, grouping truthful information is a complicated task. The computational advance of natural language processing is growing every day to solve these problems.

The present research work studies the way texts are clustered with the generation of Embeddings. In particular, it focuses on using different methods to apply supervised and unsupervised models so that efficient results can be obtained when encountering automatic clustering tasks.

Five Datasets were worked with, and as a result of the implementation of the supervised models it was determined that the best Embedding is FastText implemented with Gensim and applied in models based on boosting. For the unsupervised models the best Embedding is Glove applied in neural network models with AutoEncoder and K-means layer.

Keywords: Natural processing language, Model analysis, Artificial intelligence, Datasets with text, Spanish datasets, Clustering, Embeddings, Supervised and Unsupervised models.

Agradecimientos

Estos agradecimientos van para la guía y apoyo de mi asesor. Así mismo, agradecer a mis familiares, amigos y compañeros de trabajo. Cada uno de ellos representó una gran motivación en la preparación de esta investigación.

Índice general pág.

Índice de tablas

pág.

Índice de figuras

pág.

Capítulo 1.

Generalidades.

1.1. Introducción.

La agrupación de textos es una acción de almacenar los datos conglomerados de forma consecutiva, de tal sentido que la forma de acceder a estos sea sencilla de interpretar y visualizar. Según Awasthi, P., & Zadeh, R. B. (2010, diciembre, p.1), nos mencionan que el Clustering o su traducción literal "agrupamiento" ha sido tradicionalmente una herramienta de aprendizaje no supervisado, a pesar de muchos usos en varios campos. Sin embargo, Thwe, Y. M., Ogawa, M., & Dung, P. N. (2019, noviembre, p.11). Defiende que el término Clustering debería ser aplicado con modelos no supervisados y que el término correcto para un modelo supervisado debe ser Clasificación. La tarea de agrupar textos sin etiquetas es un problema por la cual muchos estudios vienen atacando y hacen que el término Clustering tenga otro enfoque, esto lo mencionan los autores en su estudio Wang, J., Ma, Z., Nie, F., & Li, X. (2021, junio, p1.). Dado que la adquisición de información de etiquetas en el mundo real es molesta y laboriosa, a veces incluso imposible. Por ende, numerosas técnicas de validación usadas en modelos supervisados han sido ampliamente adoptadas para preprocesar los datos primarios, validar la suposición y predecir el modelo desconocido. Aún no existe una teoría bien establecida para describir el Clustering, por tanto, este estudio utiliza diversos modelos tanto supervisados como no supervisados para así cumplir con los objetivos.

Si planteamos un caso cotidiano consideremos el buzón de reclamos de algún banco, donde los correos llegan diariamente y estos deben agruparse en grupos, en la cual cada uno cual corresponde a un reclamo concreto. En este caso, el ojo humano tiene claro a qué grupo debe pertenecer cada documento, pero el gran número de correos hace que la agrupación manual sea muy dificultosa y esto hace que la empresa tenga mucho personal para atender todos estos correos. En este caso, un algoritmo puede interactuar con el trabajador para ayudarle a agrupar los documentos sin exigirle demasiado. Los enfoques tradicionales de Clustering optimizan alguna función objetivo, como el K-means o el K-median sobre el conjunto de puntos dado. Estos modelos trabajan bajo la suposición implícita de que minimizando una determinada función

objetivo se puede llegar a la verdad subyacente de la agrupación.

Cuando se trabaja con información de páginas web, correos, documentos, etc., no se sabe a certeza si es posible que no haya una manera de alcanzar el objetivo de agrupación que un personal tiene en mente sin interactuar con él. Por ejemplo, consideremos otro caso de información: Documentos que representan artículos de revistas científicas. Estos documentos podrían agruparse como (ciencia, tecnología, economía, etc.). Sin embargo, ésta es sólo una de las muchas agrupaciones posibles. La agrupación (ciencia + tecnología, economía, etc.) es igualmente probable apriori. O quizás el personal quiera que estos artículos se agrupen en (artículos de finanzas) frente a (artículos de economía). Estos diversos escenarios motivan a la necesidad de considerar el problema de Clustering bajo retroalimentación.

Recientemente, ha habido un interés en investigar diversos modelos y crear un marco teórico más formal para analizar los problemas y los algoritmos de Clustering. Por ejemplo, Thwe, Y. M., Ogawa, M., & Dung, P. N. (2019, noviembre, p.3) Nos hace énfasis al estudio de un modelo de auto entrenamiento basado en AutoEncoder en la cual nos muestra como una técnica eficaz para agrupar textos cortos. También los investigadores de Google Ontañón, S., Ainslie, J., Cvicek, V., & Fisher, Z. (2021). Nos demuestran como el uso de Transformers y AutoEncoder pueden ayudar a mejorar tareas complejas de NLP.

En el desarrollo de la investigación se utilizaron diversas técnicas de Embeddings, tales como Word2Vec (Le, Q., & Mikolov, T., 2014), GloVe (Pennington, J., Socher, R., & Manning, C. D., 2014), FastText (Zhou, M., Liu, D., Zheng, Y., Zhu, Q., & Guo, P., 2020)) Bert (Devlin, J., Chang, M. W., Lee, K., & Toutanova, K., 2018), todos estos Embeddings fueron extraídos con un corpus de español recopilado por la Universidad de Chile según los autores Utia Deza, J. V. (2020, agosto, p.10), estos Embeddings tienen bastante uso para el Procesamiento de Lenguaje Natural, debido a que estas son representaciones de palabras útiles, y dan mejor rendimiento en tareas diversas en el campo de NLP.

1.2. Definición del problema.

Hoy en día, existen diferentes técnicas de procesamiento de lenguaje natural (NLP), en su mayoría entrenados con vectores que presentan características de alta dimensionalidad, con representaciones muy variadas. Esto permitía que los modelos no tengan un desempeño adecuado al momento de agrupar textos. En campos recientes, la generación de Embeddings está permitiendo mejorar los desempeños de los modelos, debido a la asignación vectorial de símbolos en un espacio dimensionalmente bajos. Es decir, la distancia entre dos palabras se obtiene con la representación de los vectores. Dada esta premisa, en la agrupación de textos, plantea la necesidad de: ¿Cómo la generación de Embedding permite la agrupación de textos?

1.3. Justificación.

El desarrollo de esta investigación posibilita el estudio de las técnicas de lenguaje de procesamiento natural basados en Embedding, con la aplicación de algunos modelos supervisados y no supervisados. Por tanto, el presente trabajo permite ver como los Embedding se adaptan mejor para tareas de agrupación de textos. Con los resultados se logra incorporar al conocimiento científico y se contribuye a mejorar las técnicas de procesamiento natural de textos.

1.4. Alcance.

El presente proyecto parte de un grupo de Datasets, de los cuales algunos serán descargados de una ruta pública de Kaggle [1]y otros fueron creados utilizando técnicas de Web Scrapping. Los Datasets que estén en idioma inglés fueron traducidos a español, y se tomaron una muestra de la información que mejor han sido traducidos, para no tener mucho ruido o sesgo. Esto permitió tener un buen conjunto de datos, también se aplicaron diferentes técnicas de análisis y limpieza de datos.

Una vez obtenida los Datasets se utilizaron diferentes técnicas de Embeddings

[1] https://www.kaggle.com/

para desarrollar la investigación. El proyecto termina con la generación de reportes y análisis estadísticos basados en la investigación de estos Datasets.

1.5. Limitaciones.

El proceso de entrenar modelos de inteligencia artificial, hace que se necesite de un ambiente que tenga buenos recursos de CPU y GPU. Esta investigación tuvo la limitación de solo contar con recursos académicos gratuitos como es el uso de Google Colab[2], esto hizo que la investigación demore un poco en entrenar y extraer resultados. Por otro lado, conseguir Datasets en español tampoco fue una tarea sencilla debido a su poca investigación, lo que ocasionó que se tenga que conseguir Datasets con texto en inglés, donde se tuvo que utilizar algoritmos que consumen de la API de Google para traducción automática. Tener un especialista en el idioma que valide la traducción a mano representaría un costo y tiempo muy alto, debido a la gran cantidad de información de los textos dentro de los Datasets.

Otra limitación importante es el avance de las técnicas de lenguaje de procesamiento natural en el idioma español, algunas de ellas están en fase experimental o con un desempeño no tan favorable.

1.6. Viabilidad.

1.6.1. Viabilidad Técnica.

Para el desarrollo de este proyecto se tuvo libre acceso a los lenguajes de programación y a las diferentes técnicas para procesamiento de texto.

1.6.2. Viabilidad Económica.

En el desarrollo de este proyecto no se contó con limitaciones económicas debido a que las herramientas y datos a utilizar son Open Source, excepto algunos otros gastos administrativos que fueron coberturados por el investigador.

[2] https://colab.research.google.com/

1.7. Objetivo General.

"Aplicar modelos supervisados y no supervisados basado en la generación de Embeddings para la agrupación de textos"

1.8. Objetivos Específicos.

- OE1: Analizar técnicas de NLP en Datasets de texto para la limpieza y normalización de datos.
- OE2: Generar Embeddings en Datasets de texto.
- OE3 Aplicar modelos supervisados y no supervisados utilizando los Embeddings
- OE4: Validar modelos supervisados y no supervisados utilizando los Embeddings

1.9. Planteamiento de las preguntas.

- ¿Cómo las técnicas de NLP nos permiten la limpieza y normalización de datos? Esta pregunta se responde con el OE1.
- ¿Cómo generar Embeddings en Datasets de texto? Esta pregunta se responde con el OE2
- ¿Cómo un modelo supervisado y no supervisado ayuda a la agrupación de textos basado en Embeddings? Esta pregunta se responde con el OE3 y OE4.

1.10. Datasets.

Para el presente trabajo se utilizaron cinco Datasets, estos se presentan a continuación:

- IMDB Dataset - Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011): Este conjunto de datos contiene el análisis del sentimiento de las críticas de películas. Con un total de 50 000 críticas etiquetadas de alta polaridad. Cada una de las opiniones etiquetadas tiene una etiqueta de sentimiento binaria, es

decir, "pos" para una crítica positiva o "neg" para las críticas negativas.

Tabla 1: Dataset IMDB.

Sentimiento	Cantidad
Negativo	1500
Positivo	1500

Fuente: Elaboración propia.

Figura 1: Muestra del Dataset IMDB.

	id	review_es	clean_review	sentimiento	words_len
0	47743	Mis pensamientos en la película, 9IT no fue bu...	destruirá mundo embargo intelectual juego huma...	negativo	207
1	14484	Anunciado por el canal siete en Australia como...	mundo titular créditos cegación mitología leva...	negativo	236
2	14531	Después de la fuerza de ataque espantosa, las ...	destruirá mundo escenario embargo gráficos arc...	negativo	278
3	14541	En la villa (2000) ** 1/2 1 / 2warning: puede ...	nominaciones embargo anciano siempre necesita ...	negativo	206
4	14702	Stifler, ha terminado de correr su milla desnu...	beta mundo odiamos casa blanco hembra pornogra...	negativo	275

'Hay 3000 observaciones con 5 características'

Fuente: Elaboración propia

La Tabla 1 contiene la cantidad de datos clasificados por sentimiento después de la limpieza de datos, en la Figura 1 podemos observar una muestra de datos.

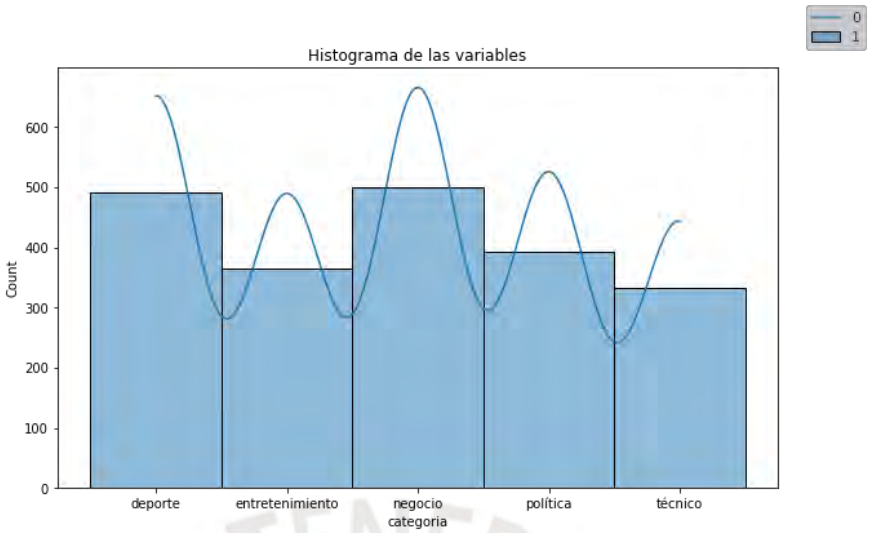
- BBC News - Archive - Greene, D., & Cunningham, P. (2006): Este conjunto de datos contiene noticias de la BBC, clasificados en cinco categorías (negocios, entretenimiento, política, deporte, tecnología). Con un total de 2225 datos.

Tabla 2: Dataset BBC News.

Categoría	Cantidad
Deporte	491
Entretenimiento	366
Negocio	500
Política	394
Técnico	334

Fuente: Elaboración propia.

Figura 2: Histograma de variables del Dataset BBC News.



Fuente: Elaboración propia.

Figura 3: Muestra del Dataset BBC News.

	categoría	título	contenido	clean_contenido	id
0	deporte	MIDO hace la tercera disculpa.	Ahmed 'Mido' Hossam ha hecho otra disculpa al...	mido expulsado egipto tiempo préstamo camerún ...	1412
1	deporte	Owen Set para el rol de Skipper	Gales Número de ocho Michael Owen dice que re...	dragones sencillo caer entrar tiempo reemplaza...	1660
2	deporte	Preguntas de Moore Capitanía	Brian Moore cree que el capitán de Inglaterra...	manejar lesiones campeones cree allí jonny joh...	1659
3	deporte	Inglaterra 'para lanzar refescas refrescas'	Inglaterra protestará por la Junta Internacio...	premiership pasar daily high kaplan tres infor...	1658
4	deporte	Wilkinson regresa 'poco probable'	Jonny Wilkinson se ve para perder la totalida...	premiership recuperarse lesiones campeones tie...	1657

Hay 2085 observaciones con 5 características

Fuente: Elaboración propia.

- Peruvian Food Reviews – Lázaro, Kaggle (2021): Este conjunto de datos en español contiene reseñas de comensales de 8791 restaurantes peruanos. Etiquetados con un total de 20 etiquetas. Con un total 1 160 666 reseñas de datos. En esta muestra, se aplicó una polaridad utilizando TextBlob[3].

Tabla 3: Dataset Peruvian Food Reviews.

Sentimiento	Cantidad
Negativo	430
Neutro	430
Positivo	430

Fuente: Elaboración propia.

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

Figura 4: Muestra del Dataset Peruvian Food Reviews.

	id_review	review	clean_review	sentimiento	words_len	Polarity
0	R129248	La comida más decepcionante de nuestra estanci...	tiempo apenas probamos amazonia alegro fuente ...	negative	81	-0.075104
1	R77336	El precio de los platos es caro para la cantid...	soles camareros carnes puedes pensando hecho b...	negative	74	-0.062895
2	R77558	Ayer por la noche fui a comer con mi esposo y ...	papas líneas cuarto ayer esperábamos orden exp...	negative	68	-0.212946
3	R77772	Fui con dos amigas adquiriendo una promoción d...	siacaso retiró orden bebidas atencion sesión m...	negative	60	-0.045833
4	R77885	Siendo la Costa Verde y teniendo el recuerdo d...	caliente último parecia ayer salad sandwiches ...	negative	122	-0.168929

'Hay 1290 observaciones con 6 características'

Fuente: Elaboración propia.

- Consumer Complaint – Consumer Financial Protection Bureau (2020): Este conjunto de datos contiene los reclamos de los consumidores sobre los productos y servicios financieros que enviaron a las empresas para que las respondan. Contiene 19 etiquetas y un total de 125 010 datos.

Tabla 4: Dataset Consumer Complaint.

Producto	Cantidad
Cobro de deudas	1000
Créditos de consumo	1000
Cuenta bancaria o servicio	1000
Cuenta de cheques o ahorros	1000
Hipoteca	1000
Informes de crédito	1000
Préstamo de día de pago	1000
Préstamo de vehículo o arrendamiento	1000
Préstamo estudiantil	1000
Tarjeta de crédito	1000
Tarjeta de prepago	1000
Transferencias de dinero	1000

Fuente: Elaboración propia.

Figura 5: Muestra del Dataset Consumer Complaint.

	producto	reclamos	Complaint ID	words_len	clean_reclamos
0	Cobro de deudas	Soy víctima de robo de identidad y en XXXX XXX...	1876097	66	attn escuche identidad menos después considera...
1	Cobro de deudas	Constar está intentando recolectar en un contr...	1648001	32	automático legal intentando información dijero...
2	Cobro de deudas	Recibí una llamada en XXXX para la recolección...	1322451	44	acumulara financial dijeron ahora total días c...
3	Cobro de deudas	La recuperación mejorada ha vuelto a una deuda...	1747969	34	sino señora recuperación después letra solo es...
4	Cobro de deudas	A. Tengo un sistema IC notificado dos veces ("...	1749969	184	requiere cobranza legales información identida...

Hay 12000 observaciones con 5 características

Fuente: Elaboración propia.

- Abstracts Scopus 2021: Este conjunto de datos es de elaboración propia extraída de la base de datos de Scopus la cual contiene el resumen de artículos científicos etiquetados en 8 niveles (agricultura, cultura, deporte, economía, salud, tecnología) con 2 754 datos.

Tabla 5: Dataset Abstracts Scopus.

Tipo	Cantidad
Agricultura	369
Cultura	500
Deporte	500
Economía	500
Salud	408
Tecnología	477

Fuente: Elaboración propia.

Figura 6: Muestra del Dataset Abstracts Scopus.

	resumen	tipo	clean_resumen	words_len
0	El tizón tardío, causado por Phytophthora infe...	agricultura	variedades universidad granjas agricultores ra...	178
1	Los recolectores de cazadores costeros de los ...	agricultura	nuevos universidad papel permanente recolector...	162
2	Antecedentes: Linaceae ha sido reconocido en t...	agricultura	ecológico desarrolla patrones madre importanci...	175
3	El Mayorazgo de Huasán, fundado a mediados del...	agricultura	fundado mayorazgo mediados memoria conflicto m...	111
4	Desde un modelo econométrico con datos del pan...	agricultura	consecutivos realizó especialmente diversidad ...	128

Hay 2754 observaciones con 4 características

Fuente: Elaboración propia.

Capítulo 2.

Marco Conceptual.

2.1. Word Embedding.

Los Embeddings son un conjunto de modelados en la cual son representados como vectores de números reales. Según, Rampisela, T. V., & Yulianti, E. (2020, junio, p.2). *"Los Embeddings son un método para representar datos de texto en forma de vector".* Asu ves, Alvarez, J. E., & Bast, H. (2017, octubre, p.15). Mencionan: *Los Embeddings son una representación vectorial del significado de las palabras con una noción difusa. En la práctica, esto suele significar que los Embeddings se colocan en un espacio similar o relacionados con los que están cerca unos de otros y los Embeddings diferentes están alejados unos de otros.*

2.2. Datasets.

Los conjuntos de datos son colecciones de datos de forma tabular comúnmente, estos datos se pueden encontrar en diferentes formatos como datos estructurados, semi estructurados y no estructurados. Según Renear, A. H., Sacchi, S., & Wickett, K. M. (2010, octubre, p.1) *El concepto de conjunto de datos es común a casi todas las disciplinas científicas en las que los datos constituyen la base empírica de las actividades de investigación.*

Es un campo de las ciencias de la computación que estudia diferentes métodos para procesar el lenguaje natural. El NLP nos da la posibilidad de interactuar con la IA, en cual se trabaja con algoritmos que obtienen y manipulan datos, esto de acuerdo a su arquitectura, de las que existen cinco niveles, Según Vásquez, A. C., Quispe, J. P., & Huayna, A. M. (2009, diciembre, p.48): *"Nivel Fonológico (...); Nivel Morfológico (...); Nivel Sintáctico (...); Nivel Semántico (...); Nivel Pragmático".*

2.3. Clustering o Agrupamiento.

Según Thwe, Y. M., Ogawa, M., & Dung, P. N. (2019, November, p.11). *El agrupamiento es una tarea que consiste en reunir los puntos de datos en una serie de grupos de datos en una serie de grupos basados en su similitud (...) Los grupos resultantes con puntos de datos similares se denominan clústeres. (...) El Clustering sólo requiere el conjunto de datos que tiene puntos de datos sin proporcionar las etiquetas (aprendizaje no supervisado).*

Capítulo 3.

Estado del Arte.

En este capítulo se presenta el estado de arte, para lo cual se decidió plantear preguntas de investigación que ayudaron a enfocar lo que quiere realizar, también se planteó cadenas de búsquedas para resolver las preguntas.

En base a las preguntas planteadas se determina ciertas palabras clave que ayudan a resolver las preguntas de investigación.

Query:

```
TITLE (
        ( "*text*" AND "*embedd*" ) AND
        ( "nlp" OR "natural process language" OR "*model*" ) AND
        ( "*cluster*" OR ( *class* ) )
      )
```

En el proceso de revisión bibliográfica se hizo la búsqueda de varios artículos científicos usando la consulta avanzada de las bases de datos de SCOPUS[4] e IEEEXplore[5]. Se procedió a revisar los trabajos relacionados con el tema, donde se revisó el resumen, Datasets utilizados, técnicas de Embeddings y resultados de la investigación. Del conjunto de datos encontrados en esta búsqueda se eliminaron algunos por duplicados o similares y por otras técnicas que no son relevantes a esta investigación.

[4] https://www.scopus.com/
[5] https://ieeexplore.ieee.org/

Tabla 6: Análisis de revisión de artículos.

Autor	¿Qué modelos basados en Embedding utilizan?	¿Qué técnicas de Embedding utilizan?	¿Qué Datasets utilizan?	¿Cuáles fueron los resultados?
He, B., Ahamad, M., & Kumar, S. (2021, August).	• Hierarchical Recurrent Neural Network (HRNN) • Temporal Interaction Embedding (TIES)	User Sequence Embedding vector (USEV)	• Wikipedia • Yelp	• Para el Dataset de Wikipedia, el mejor modelo que respondió fue el TIES con un F1 score de 0.578. • Para el Dataset de Yelp, el mejor modelo que respondió fue el TIES con un F1 score de 0.554.
Liu, N., Wang, Q., & Ren, J. (2021).	• Binary Relevance (BR) • Probabilistic Classifier Chain (PCC) • CNN • Sequence to Sequence RNN (seq2seq-RNN) • Sequence Generation Model (SGM) • set-RNN	• BERT • WordPiece embedding	• Arxiv Academic Paper Dataset (AAPD) • Slashdot, Reuters-21578 • Ren-CECps • COPD	• Para el Dataset de AAPD, el mejor modelo que respondió fue el set-RNN con un F1: 0.731 • Para el Dataset de Slashdot, el mejor modelo que respondió fue BERT con un F1: 0.583 • Para el Dataset de Reusters-21578, el mejor modelo que respondió fue LBA con un F1: 0.672 • Para el Dataset de COPD, el mejor modelo que respondió fue LBA con un F1: 0.978
El-Alami, F. Z., El Alaoui, S. O., & Nahnahi, N. E. (2021).	• CNN • LSTM • BiLSTM • MLP • SVM	• BERT • ULMFiT • ELMO • Word2vec • TFIDF	OSAC	Mejor Embedding BERT y mejor modelo SVM a un 94% de Accuracy
Shin, H. S., Kwon, H. Y., & Ryu, S. J. (2020).	• CNN • LSTM	• CSI Embedding	• CSI • Wikitext	• Para el Dataset de CSI, el mejor modelo que respondió fue CNN con un F1: 0.934 • Para el Dataset de WikiText, el mejor modelo que respondió fue LSTM con un F1: 0.925

Dai, Z., Li, K., Li, H., & Li, X. (2020, October)	• Auto-Encoder	• TF-IDF • SIF • STC • BERT	• Baidu Q&A • Today's Headlines • SINA	• Para el Dataset de Baidu Q&A, el mejor modelo que respondió fue uno propio AE con un ACC: 83.4 y NMI: 61.4 • Para el Dataset de **Today's** Headlines, el mejor modelo que respondió fue uno propio AE con un ACC: 67.9 y NMI: 58.4 • Para el Dataset de SINA, el mejor modelo que respondió fue uno propio AE con un ACC: 66.3 y NMI: 59.3
Li, Y., & Ye, M. (2020, April).	• CNN • LSTM • RELSTM	• LSTM	• ChnSentiCorp-Htl-unba-10000 • Sogou News Paper • Theme data	• Para el Dataset de ChnSentiCorpHtl-unba-10000, el mejor modelo que respondió fue RELSTM con un ACC: 0.89. • Para el Dataset de Sogou Paper, el mejor modelo que respondió fue RELSTM con un ACC: 0.975 • Para el Dataset de Theme, el mejor modelo que respondió fue RELSTM con un ACC: 0.977
Bounabi, M., El Moutaouakil, K., & Satori, K. (2020, April).	• SVM • FNN • RNN Model	• Word2vec • Doc2vec • Paragraph Vector-Distributed Memory (PV-DM)	BBCSport	• El mejor modelo para este estudio fue uno basado en redes neuronales con capas personalizadas llegando a un ACC de 99%, y el mejor Embedding fue Word2vec
Zhou, M., Liu, D., Zheng, Y., Zhu, Q., & Guo, P. (2020).	• SVM • RNN • CNN • DWE	• Word2vec • GloVe	ChnSentiCorp	• El mejor modelo fue uno propio basado en redes neuronales con capas personalizadas DWE con el Embedding de Glove, alcanzando un ACC de 94.8

Hadifar, A., Sterckx, L., Demeester, T., & Develder, C. (2019, August).	• CNN - Auto-Encoder	• TF • TF-IDF • Skip-Thought • SIF • STC • BERT	• SearchSnippets • Stackoverflow • Biomedical	• Para el Dataset de SearchSnippets, el mejor modelo que respondió fue uno propio CNN con un ACC: 77.1 y NMI: 56.7, ARI: • Para el Dataset de Stackoverflow, el mejor modelo que respondió fue uno propio CNN con un ACC: 59.8 y NMI: 54.8 • Para el Dataset de Biomedical, el mejor modelo que respondió fue uno propio CNN con un ACC: 54.8 y NMI: 47.1
Dong, Y., Liu, P., Zhu, Z., Wang, Q., & Zhang, Q. (2019).	• CNN • LSTM • Deep CNN • P-LSIAM	• Word2vec • BERT • FastText	• Yahoo • DBPedia • AGNews • Yelp P. • Yelp F.	• Para el Dataset de Yahoo, el mejor modelo que respondió fue uno propio P-LSIAM con un ACC 80.29 • Para el Dataset de DBPedia, el mejor modelo que respondió fue uno propio P-LSIAM con un ACC 99.39 • Para el Dataset de AGNew, el mejor modelo que respondió fue uno propio P-LSIAM con un ACC 93.19 • Para el Dataset de Yelp P., el mejor modelo que respondió fue uno propio P-LSIAM con un ACC 95.55 • Para el Dataset de Yelp F., el mejor modelo que respondió fue uno propio P-LSIAM con un ACC 66.43

Liu, W., Liu, P., Yang, Y., Yi, J., & Zhu, Z. (2019).	• SVM • TWE • GoW • BoE • PWE	Topical-word-embedding (TWE)	• 20NewGroup • Reuter21578	• Para el Dataset de 20NewGroup, el mejor modelo que respondió fue uno propio PWE con un ACC: 84.4 • Para el Dataset de Reuter21578, el mejor modelo que respondió fue uno propio PWE con un ACC: 90.2
Jinarat, S., Manaskasemsak, B., & Rungsawang, A. (2018, December).	• Tree Clustering algorithm (STC) • Word Semantic Graph (WSG)	Word2Vec	• Tweet Dataset • Web Snippet Dataset	• Para el Dataset de Tweet Dataset, el mejor modelo que respondió fue uno propio WSG con un F1: 0.51 • Para el Dataset de Web Snippet Dataset, el mejor modelo que respondió fue uno propio WSG con un F1: 0.54
Cha, M., Gwon, Y., & Kung, H. T. (2017, November).	• Brown clustering • K-means	• Bag-of-words • Word2vec • FastText • Doc2vec	Wiki-SimpleWiki Corpus	En este estudio el mayor desempeño fue el Embedding de FastText con el modelo no supervisado de K-means (Spearman: 0.825, Pearson: 0.822)

Fuente: Elaboración propia.

De la Tabla 6 podemos observar que los modelos supervisados basados en Embeddings que más uso tienen son: SVM, modelos basados en LSTM y modelos basados en RNN. En la cual, el que tiene mayor desempeño son los modelos basados en RNN. Los Embeddings que más han sido aplicados son: W2V, BERT y Embeddings propios, se puede también evidenciar que depende del estudio uno responde mejor que el otro.

Para los modelos no supervisados basado en Embeddings los que tienen mayor uso son los basados en RNN combinando algún modelo como K-Means o alguna técnica de extracción de N-Gramas o Similitud basado en Embeddings.

Capítulo 4.

Desarrollo y Resultados.

4.1. Técnicas de limpieza y normalización de datos.

4.1.1. Limpieza de datos.

Para la limpieza de los Datasets se usaron las siguientes técnicas:

a) DropNa: Es una técnica que permite eliminar los valores nulos que contenga un Dataset, de tal manera de no afectar el rendimiento del modelo.

b) DropDuplicates: Es una técnica que permite eliminar los valores duplicados en un Dataset, de tal manera de no afectar el rendimiento del modelo.

c) LowerCase: Es el proceso por el cual se le pasa los valores de una cadena para convertir en minúsculas. Este se usa con el fin de poder estandarizar las palabras.

d) StopWord. Por su traducción literal palabras vacías, son un conjunto de palabras de uso común en un idioma. Estos se almacenan en un conjunto de datos y luego usando la fórmula de conjuntos de diferencia se va eliminando los StopWords.

e) Punctuation Removal. Es una técnica que elimina los caracteres alfanuméricos, así como también los signos de puntuación.

Para usar estas técnicas de limpieza se usó la librería de NLTK[6] en español, seguido de las expresiones regulares con Python.

4.1.2. Normalización de datos.

Para la normalización de datos se usó la técnica de LabelEncoder[7] que codifica las

[6] https://www.nltk.org/
[7] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html

etiquetas de destino con un valor de 0 y 1. StandardScaler[8] que estandariza los datos eliminando la media y escalando a la varianza de la unidad.

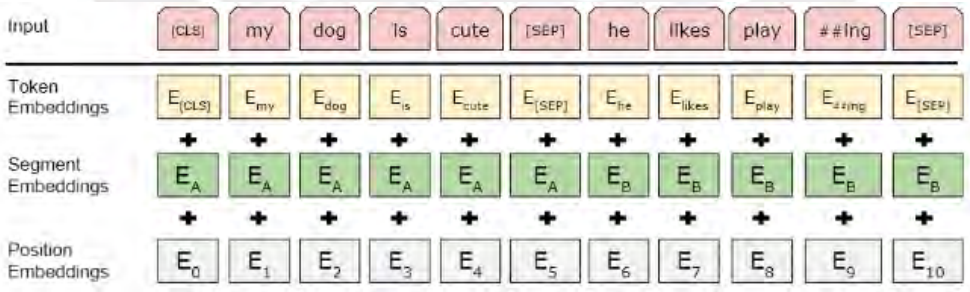
4.2. Generación Embeddings.

Para la generación de Embeddings se utilizaron las siguientes técnicas de lenguaje de procesamiento natural:

a) BERT.

La arquitectura del modelo BERT es un codificador transformador bidireccional no supervisado. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017), publicado en la biblioteca de tensor de los autores: Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Como versión en español de Bert, nace Beto, que fue entrenado con la técnica de enmascaramiento de palabras enteras. Por lo tanto, en la Figura 7 se muestra una representación del modelo BERT, donde se suman tanto las incrustaciones de tokens como la segmentación y la posición.

Figura 7: Representación de Bert.



Fuente: Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018).

Según Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). *Las incrustaciones de entrada son la suma de incrustaciones de tokens, incrustaciones de segmentación e incrustaciones de posición.*

[8] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

Para extraer los Embeddings preentrenados, se usó la librería de Tensorflow[9] con un tokenizador. El algoritmo[10] usa el tokenizador de Tensorflow para convertir los textos en formato array, esto sirve de entrada para el modelo de Beto. Después de esto se extrae la capa **"pooler_output"**, en la cual se encuentran los valores de los Embeddings.

b) Glove.

Este algoritmo se basa en las matrices de ocurrencia, que representan la frecuencia de ocurrencia de las palabras entre sí. Glove sigue una función para obtener las incrustaciones de palabras:

$$J_\theta = \sum_{i,j=1}^{v} f(x_{lj})\left(w_i^T \tilde{w}_j + b_j + \tilde{b}_j - \log x_{lj}\right)^2 \qquad \text{Ec. 01}$$

Donde el valor del elemento x_{lj} en la matriz de co-ocurrencia X corresponde al número de veces que la palabra l aparece junto a la palabra j. Esto contrasta con lo que menciona los autores: Swysen Cachaña, T. B. (2020). y Kilimci, Z. H., & **Akyokuş, S**. (2019)

El modelo Glove tiene 300 dimensiones y 855 380 vectores aplicados a un corpus español - Corpora[11]. Pérez Rojas, J. (2018).

c) FastText.

FastText utiliza caracteres de n-gramas para representar las palabras como el elemento más pequeño. Cada palabra se divide en caracteres de n-gramas donde: 3 ≤ n ≤ 6. Según **Kilimci, Z. H., & Akyokuş, S. (2019)**, *FastText aprende de las incrustaciones y también puede utilizarse para la clasificación de textos.*

[9] https://www.tensorflow.org/

[10] https://github.com/anthonywainer/embeddings_analysis_spanish/blob/master/src/main/embeddings_analysis_spanish/embeddings/bert_embedding.py

[11] https://github.com/josecannete/spanish-corpora

El modelo FastText tiene 300 dimensiones y 1 313 423 vectores aplicados a un corpus español. Fue desarrollado con el corpus Corpora con un peso de 3 mil millones de palabras. Pérez Rojas, J. (2018).

d) Word2Vec.

Word2vec es probablemente el modelo principal más integrado que inició una nueva tendencia en la semántica distribuida. El autor **Kilimci, Z. H., & Akyokuş, S.** (2019), nos menciona que el algoritmo de *Word2vec se basa en la aplicación de un modelo de red neuronal donde aprende asociaciones de palabras con información de corpus de texto*.

El modelo Word2Vec tiene 300 dimensiones y 1 000 653 vectores aplicados a un corpus español. Y fue desarrollado con el corpus Corpora con un peso de 1400 millones de palabras. Pérez Rojas, J. (2018).

e) GPT-2.

El GPT-2[12] fue entrenado sobre Wikipedia[13] en español utilizando técnicas de Transfer Learning y Fine-tuning. El modelo se ajustó a partir del modelo preentrenado en inglés utilizando las librerías Hugging Face[14] (Transformers y Tokenizers) envueltas en el framework de deep learning fastai v2 de los autores Berny Josúe & Obregon Carrer (2021). Esto según Lee, J. S., & Hsiang, J. (2020). *Los datos utilizados para este modelo provienen de Wikipedia en español, que contiene mucho contenido sin filtrar.*

Para extraer los Embeddings preentrenados de GPT2, se usó la librería de Tensorflow con un tokenizador. El algoritmo[15] usa el tokenizador de Tensorflow para convertir los textos en formato array, esto sirve de entrada para el modelo de GPT2. De la cual se realiza un "feature-extraction", que da como resultado los valores de los Embeddings.

[12] https://metatext.io/models/datificate-gpt2-small-spanish
[13] https://archive.org/details/eswiki-20150105
[14] https://huggingface.co/
[15] https://github.com/anthonywainer/embeddings_analysis_spanish/blob/master/src/main/embeddings_analysis_spanish/embeddings/gpt_embedding.py

Los modelos basados en Gensim se trabajan haciendo uso de un algoritmo[16] que busca los textos en formato array dentro del modelo de KeyedVectors[17] y luego extrae los Embeddings.

4.3. Métricas de evaluación.

Para validar los resultados de los modelos no supervisados se utilizaron las métricas de evaluación: Clustering Accuracy, NMI and ARI.

a) Clustering Accuracy

Para obtener el Accuracy en el Clustering se utiliza la siguiente fórmula.

$$ACC = \frac{\sum_{i=1}^{n} \delta(l_i = m(c_i))}{n} \qquad \text{Ec. 02}$$

Donde l_i es la etiqueta verdadera, c_i es el mapeo de clusters producido por el algoritmo, y m abarca todas las posibles asignaciones uno a uno entre clusters y etiquetas. Para mejorar la eficiencia, se debe utilizar el algoritmo húngaro. Wei, T., Lu, Y., Chang, H., Zhou, Q., & Bao, X. (2015).

b) Información mutua normalizada.

La información mutua normalizada (NMI) es una normalización de la puntuación de información mutua, en la que los resultados se escalan entre *0* (sin información mutua) y *1* (correlación perfecta).

$$NMI(\propto, \beta) = \frac{I(\propto, \beta)}{\sqrt{H(\propto)H(\beta)}} \qquad \text{Ec. 03}$$

Donde I es la información mutua y H es la entropía. Cuando el valor del NMI es mayor, más cerca están las dos divisiones. Sin embargo, una vez que el NMI alcanza

[16] https://github.com/anthonywainer/embeddings_analysis_spanish/blob/master/src/main/embeddings_analysis_spanish/embeddings/gensim_embedding.py

[17] https://radimrehurek.com/gensim/models/keyedvectors.html

el valor máximo, el número de clusters es el número óptimo para el modelo. Hotho, A., Staab, S., & Stumme, G. (2003)

c) Índice de Rand ajustado.

El índice de Rand computa una medida de similitud entre dos clústeres, considerando la cantidad y el total de pares de muestras. Donde se mapean en las mismas o diferentes agrupaciones en los clústeres predichos y verdaderos.

$$ARI = \frac{(RI - Expected\_RI)}{\max(RI) - Expected\_RI} \quad \text{Ec. 04}$$

Se garantiza que la puntuación ARI tiene un valor cercano a *0* para el etiquetado aleatorio, independientemente del número de clústeres y muestras, y exactamente *1* cuando los clústeres son idénticos hasta una permutación. Milligan, G. W., & Cooper, M. C. (1986).

4.4. Modelos no supervisados.

En este caso, se ha considerado trabajar con dos modelos y un tercer modelo que se basa en el modelo K-means.

a) K-means.

El algoritmo K-means fue introducido por James Macqueen hacia 1965 para minimizar la función objetivo partiendo de un conjunto de "n" observaciones en "z" grupos en los que cada observación pertenece al grupo cuyo valor medio es el más cercano MacQueen, J. B. (1965).

b) HDBSCAN.

También conocido como Clustering espacial jerárquico basado en la densidad de las aplicaciones ruidosas, basado en el algoritmo DBSCAN (Ester, M., Kriegel, H. P., Sander, J., & Xu, X., 1996), genera una jerarquía de Clustering completa de la que se puede extraer fácilmente una jerarquía simplificada compuesta sólo por los clústeres más significativos. Campello, R. J., Moulavi, D., & Sander, J. (2013)

c) AutoEncoder + K-means.

Este algoritmo es la aplicación del modelo de red neuronal AutoEncoder utilizado en el aprendizaje no supervisado, en el que primero codifica los datos en una dimensión más pequeña y luego los decodifica de nuevo en los datos originales.

Los datos de entrada para el autocodificador son las incrustaciones, estos datos se pasan al codificador y son procesados por la función de activación. El resultado de la codificación se introduce en el decodificador y, finalmente, se obtiene el vector resultante tras la reconstrucción.

i. Cálculo de la probabilidad asignada a cada punto central del clúster.

$$q_{ij} = \frac{(1 + ||z_i - u_j||)^{-1}}{\sum_{j'}(1 + ||z_i - u_{j'}||^2)^{-1}} \quad \text{Ec. 05}$$

Basándose en el algoritmo t-SNE (Van der Maaten, L., & Hinton, G., 2008) se mide la similitud entre los puntos incrustados z_i y los centroides u_j. El resultado q_{ij} es la asignación de incrustaciones a los centroides. (Hadifar, A., Sterckx, L., Demeester, T., & Develder, C, 2019)

ii. Cálculo de las probabilidades en la distribución.

$$p_{ij} = \frac{q_{ij}^2 / \sum_{i'} q_{i'j}}{\sum_{j'}(q_{ij}^2 / \sum_{i'} q_{i'j})} \quad \text{Ec. 06}$$

Este cálculo se realiza para evitar que los clústeres grandes sesguen el espacio de características ocultas. (Xie et al., 2016)

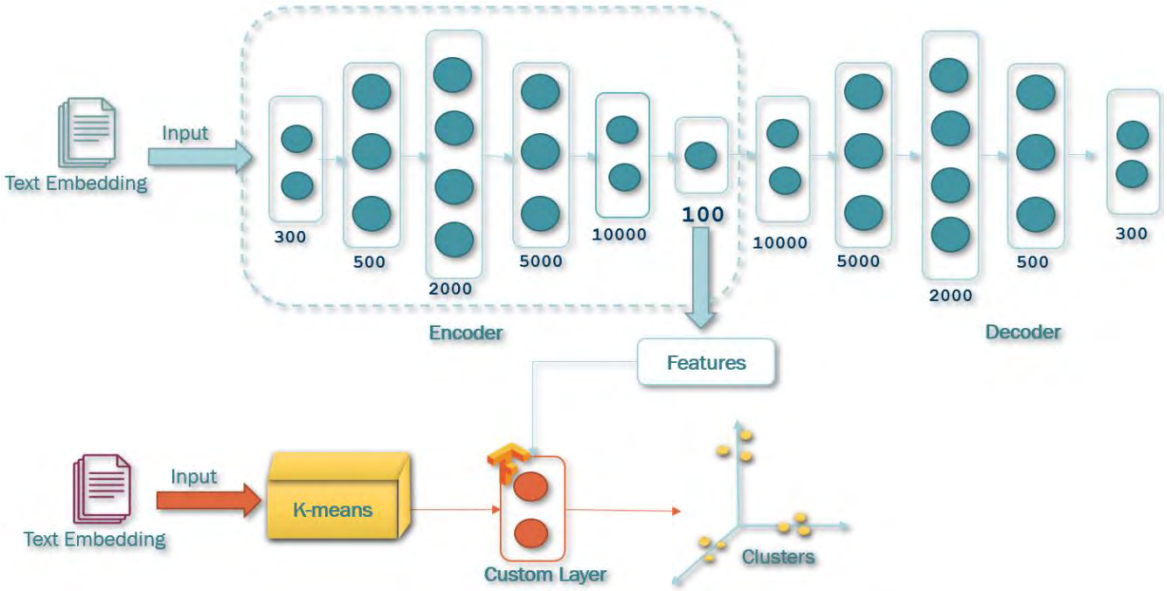
iii. Cálculo de la función de pérdida - Divergencia KL.

$$L = KL(P \,||\, Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad \text{Ec. 07}$$

La divergencia de Kullback-Leibler o puntaje de divergencia de KL, usada para calcular la función de pérdida, esta función calcula cuánto difieren las distribuciones de probabilidades una con respecto con la otra. Joyce, J. M. (2011)

Para la creación de la capa personalizada en TensorFlow se añaden las ecuaciones Ec. 05 y Ec. 07.

Figura 8: Autoencoder + K-means Clustering.



Fuente: Adaptado de Hadifar, A., Sterckx, L., Demeester, T., & Develder, C, 2019.

En la arquitectura del AutoEncoder Figura 8, se configuraron seis capas para el encoder y cinco para el decoder, la primera capa tiene un peso de 300 que representa la dimensión de los Embeddings, y las otras capas van aumentando de 500, 2000, 5000, 10000. A continuación, la última capa del codificador se extrae en 100 características. Los pesos de las capas del decoder son los inversos a los del encoder.

Para la agrupación de los textos, se utilizó el siguiente Algoritmo 1, cuya entrada son los Embeddings con sus respectivas dimensiones. En el primer y segundo paso, se entrena el modelo AutoEncoder con los Embeddings. (Figura 8). En el tercer paso, se extraen los vectores característicos de la capa del codificador. En el cuarto y quinto paso, se entrena el modelo de K-means con los Embeddings originales y luego se extraen los pesos asignados a cada centroide del clúster. En el sexto paso se configura la capa personalizada en TensorFlow, en la que se le pasan como entrada los vectores extraídos del Encoder, en el paso siete se asignan los pesos extraídos del resultado del entrenamiento de K-means. En el paso ocho se asignan las ecuaciones Ec. 05, y Ec. 07, en los pasos nueve a veintitrés se entrenan y predicen por lotes, aquí se establecen 100 iteraciones, un tamaño de lote de 512, un intervalo de actualización de 140 y un umbral

de tolerancia de 1e-3.

Finalmente, una vez realizado el entrenamiento por lotes, se validan los resultados utilizando las métricas comentadas en el apartado 4.3.

Algoritmo 1: Autoencoder + K-means Clustering.

Result: Enhancing clustering
Input: Embeddings with their vector dimensions
Pass the embeddings $\boldsymbol{E}$ to the autoencoder;
Train autoencoder model;
Extract result vectors from the encoder layer $\boldsymbol{RE}$;
Pass $\boldsymbol{E}$ to K-means model;
Extract the weights $\boldsymbol{W}$ from the K-means result $\boldsymbol{RK}$;
Configure $\boldsymbol{RE}$ as output in the custom layer;
Configure $\boldsymbol{W}$ in the custom layer;
Configure Eq. 05 and Eq. 07 in the custom layer;
Set the iterations number $\boldsymbol{I} = 100$;
Set batch size $\boldsymbol{BS} = 512$;
Set update interval $\boldsymbol{UI} = 140$;
Set the tolerance threshold to stop training $\boldsymbol{T} = 1e-3$;
For ***iter*** in range from elements of $\boldsymbol{E}$ **do**;
| **If** the rest between ***iter*** and $\boldsymbol{UI}$ is 0 **then;**
| | **Define** q with result on predict model with $\boldsymbol{E}$;
| | **Update** auxiliary target distribution with $\boldsymbol{q}$ and **Eq. 06**;
| | **Calculate** delta label from $\boldsymbol{sum}(\boldsymbol{q} != \boldsymbol{RK})/\boldsymbol{q}$;
| | **If** ***iter*** > 0 and delta label < $\boldsymbol{T}$ **then;**
| | | ***break;***
| | **end*;***
| **end*;***
| **Train** on batch assign ***BS;***
end*;*
Validate results with metrics;

Fuente: Adaptado de Hadifar, A., Sterckx, L., Demeester, T., & Develder, C, 2019.

4.5. Análisis de resultados.

DATASET 1: BBC News.

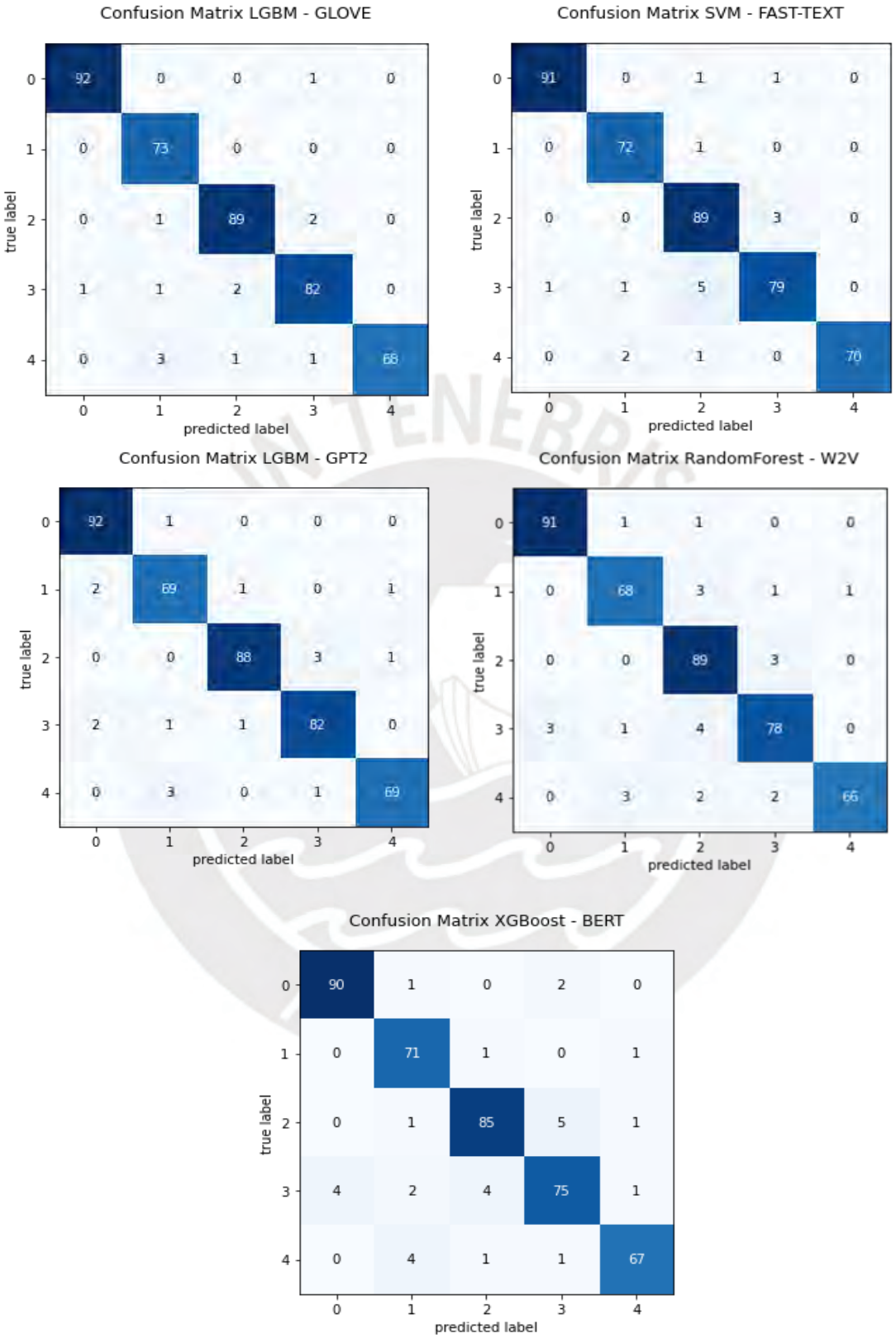
Tabla 7: Resultados de los modelos supervisados - BBC News.

Embedding	Model	Accuracy	Precision	Recall	F1 Score
GLOVE	LGBM	0.969	0.970	0.969	0.969
	XGBoost	0.957	0.957	0.957	0.957
	SVM	0.954	0.956	0.954	0.954
	RandomForest	0.950	0.950	0.950	0.950
FAST-TEXT	SVM	0.962	0.962	0.962	0.962
	LGBM	0.959	0.959	0.959	0.959
	RandomForest	0.952	0.953	0.952	0.952
	XGBoost	0.950	0.950	0.950	0.950
GPT2	LGBM	0.959	0.959	0.959	0.959
	XGBoost	0.947	0.947	0.947	0.947
	RandomForest	0.947	0.947	0.947	0.947
	SVM	0.590	0.851	0.590	0.589
W2V	SVM	0.954	0.955	0.954	0.955
	LGBM	0.947	0.948	0.947	0.947
	XGBoost	0.940	0.940	0.940	0.940
	RandomForest	0.940	0.941	0.940	0.940
BERT	XGBoost	0.930	0.931	0.930	0.930
	RandomForest	0.923	0.924	0.923	0.923
	LGBM	0.923	0.924	0.923	0.923
	SVM	0.470	0.841	0.470	0.449

Fuente: Elaboración propia.

En la Tabla 7, Glove Embedding es ligeramente más exitoso cuando se aplica a un modelo supervisado basado en Boosting LGBM. Sin embargo, SVM con FAST-TEXT también logra una predicción muy alta. El Embedding de Bert es el que menos Accuracy muestra en su resultado, pero no se puede descartar como un mal modelo porque la predicción es muy alta en modelos basados en boosting.

Figura 9: Matriz de confusión - BBC News.



Fuente: Elaboración propia.

Tabla 8: Resultados de los modelos no supervisados - BBC News.

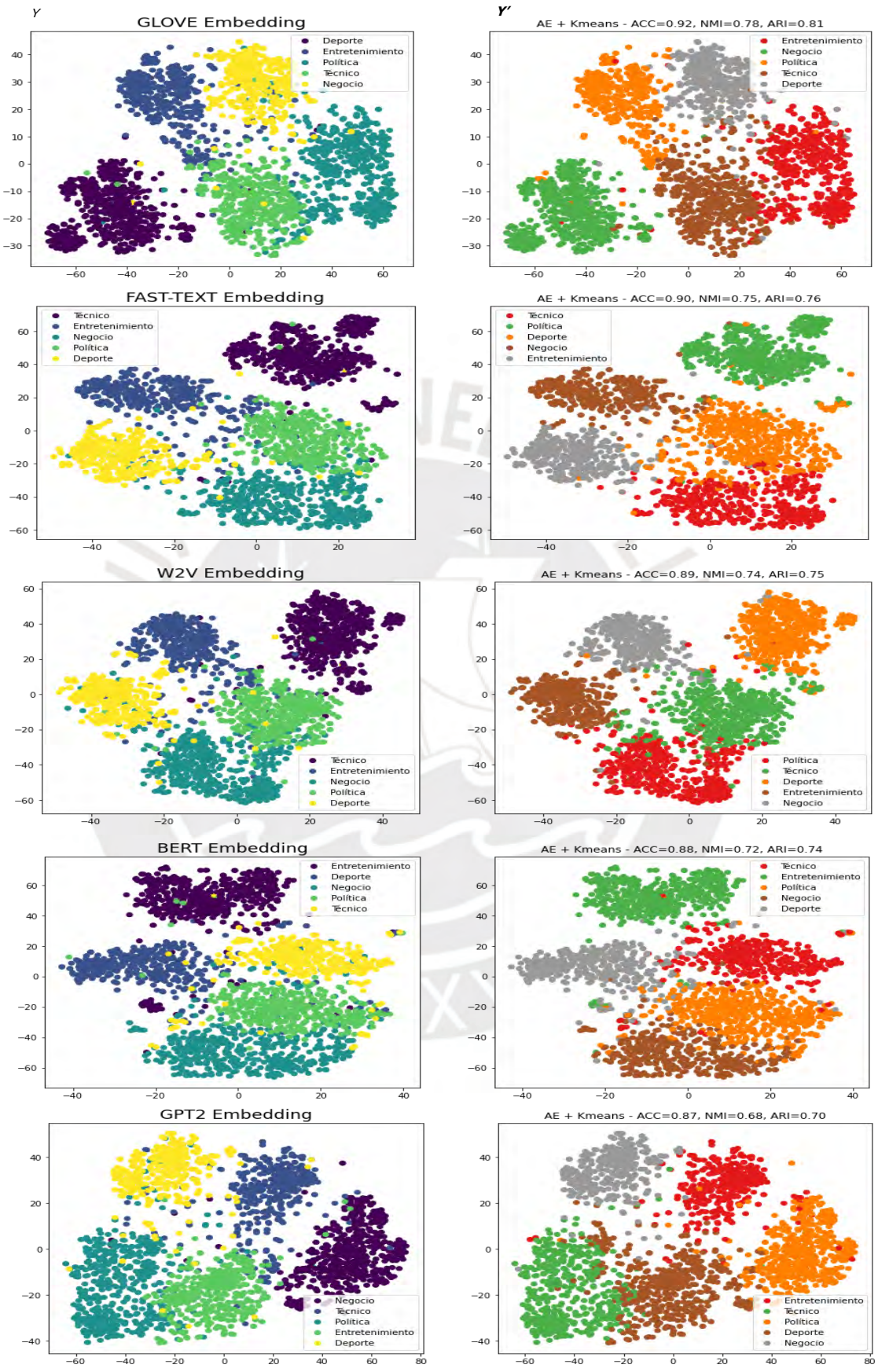
MODEL	EMBEDDING	ACC	NMI	ARI
AE K-MEANS	Glove	0.92	0.79	0.81
	Fast-Text	0.90	0.76	0.77
	W2V	0.90	0.74	0.76
	Bert	0.89	0.73	0.74
	Gpt2	0.87	0.69	0.70
K-MEANS	Gpt2	0.88	0.71	0.71
	Glove	0.87	0.70	0.71
	Fast-Text	0.87	0.69	0.70
	Bert	0.85	0.67	0.67
	W2v	0.85	0.67	0.66
HDBSCAN	Glove	0.33	0.15	0.04
	Fast-Text	0.30	0.07	0.03
	Bert	0.28	0.07	0.03
	Gpt2	0.28	0.05	0.01
	W2v	0.27	0.05	0.02

Fuente: Elaboración propia.

En la Tabla 8, Glove Embedding es ligeramente más exitoso cuando se aplica al modelo Autoencoder. Sin embargo, K-means con GPT2 también logra una predicción adecuada, un poco mejor que Glove.

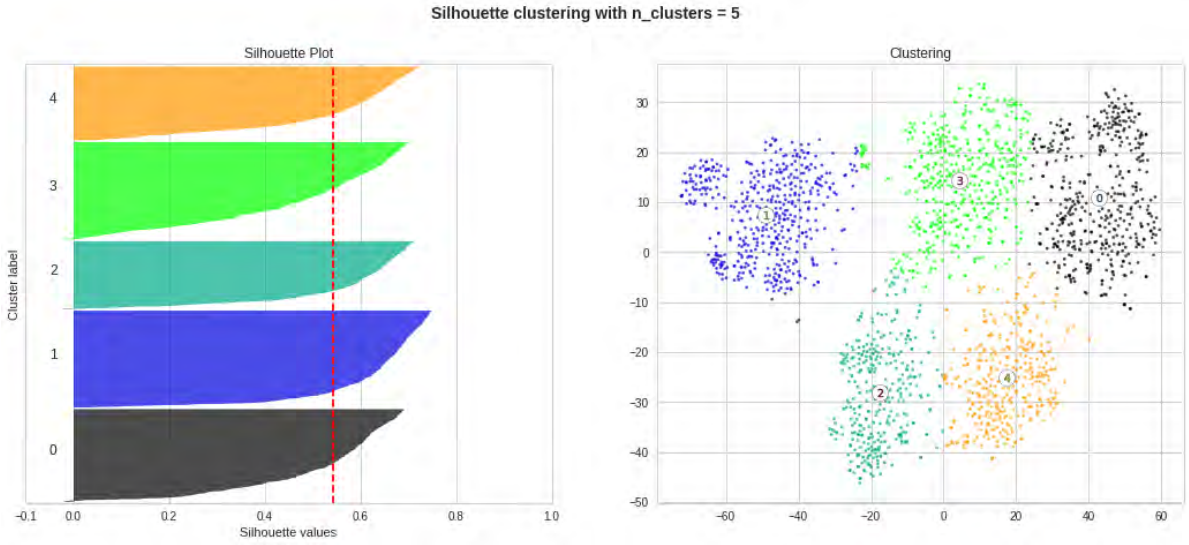
Según la distribución de la Figura 10, la variable Y está muy correlacionada con la variable Y', esto nos indica un buen nivel de predicción.

Figura 10: Clustering - BBC News.



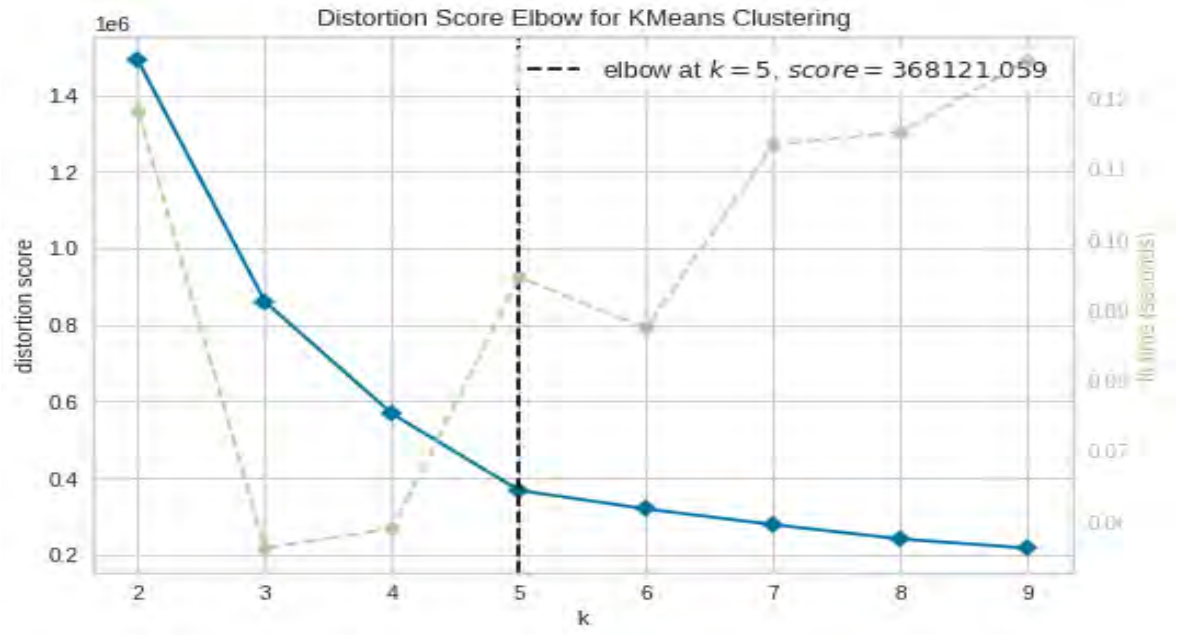
Fuente: Elaboración propia.

Figura 11: Silhouette Clustering - BBC News.



Fuente: Elaboración propia.

Figura 12: Elbow - BBC News.



Fuente: Elaboración propia.

Tanto como en la Figura 11 y la Figura 12, se puede apreciar que el método de la Silhouette[18] y Elbow[19] coinciden en la cantidad de clústeres, demostrando que el Dataset con 5 categorías está claramente bien etiquetado y que permite validar los modelos usando los Embeddings de GLOVE.

[18] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html
[19] https://www.scikit-yb.org/en/latest/api/cluster/elbow.html

DATASET 2: IBDB Reviews.

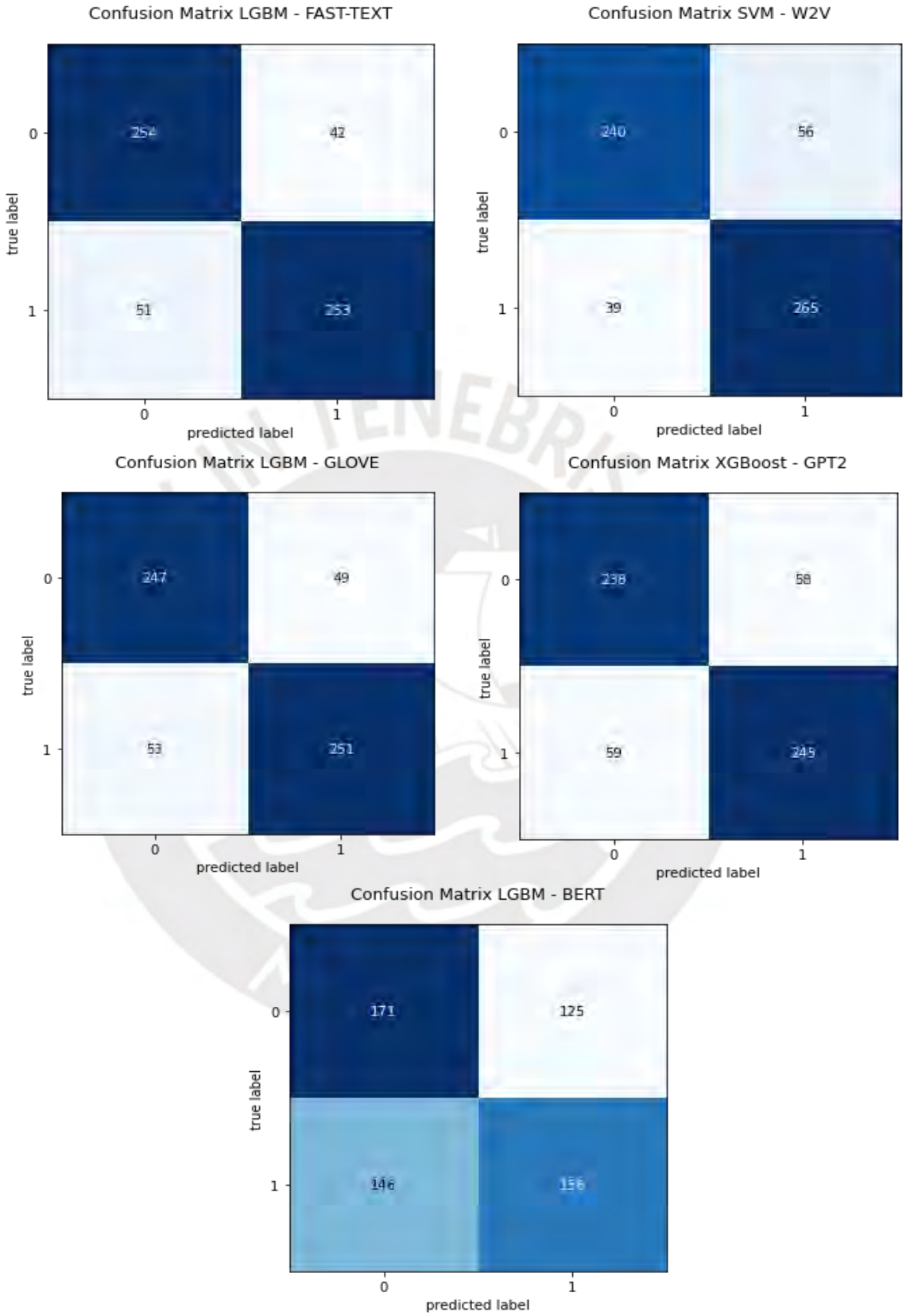
Tabla 9: Resultados de los modelos supervisados - IBDB Reviews.

Embedding	Model	Accuracy	Precision	Recall	F1 Score
FAST-TEXT	LGBM	0.845	0.845	0.845	0.845
	SVM	0.843	0.845	0.843	0.843
	XGBoost	0.828	0.829	0.828	0.828
	RandomForest	0.807	0.807	0.807	0.807
W2V	SVM	0.842	0.843	0.842	0.841
	XGBoost	0.812	0.812	0.812	0.812
	RandomForest	0.807	0.807	0.807	0.807
	LGBM	0.803	0.803	0.803	0.803
GLOVE	LGBM	0.830	0.830	0.830	0.830
	SVM	0.827	0.831	0.827	0.826
	XGBoost	0.823	0.823	0.823	0.823
	RandomForest	0.793	0.793	0.793	0.793
GPT2	XGBoost	0.805	0.805	0.805	0.805
	LGBM	0.803	0.803	0.803	0.803
	RandomForest	0.780	0.781	0.780	0.780
	SVM	0.715	0.740	0.715	0.706
BERT	LGBM	0.548	0.549	0.548	0.548
	XGBoost	0.510	0.510	0.510	0.510
	RandomForest	0.510	0.510	0.510	0.510
	SVM	0.505	0.506	0.505	0.503

Fuente: Elaboración propia.

En la Tabla 9, FAST-TEXT Embedding es más exitoso cuando se aplica a un modelo supervisado basado en Boosting LGBM. Sin embargo, SVM con W2V también logra una predicción muy alta. El Embedding de Bert es el que menos Accuracy muestra en su resultado, pero este podría descartar como modelo de bajo rendimiento porque la predicción es muy baja.

Figura 13: Matriz de confusión - IBDB Reviews.



Fuente: Elaboración propia.

Tabla 10: Resultados de los modelos no supervisados - IBDB Reviews.

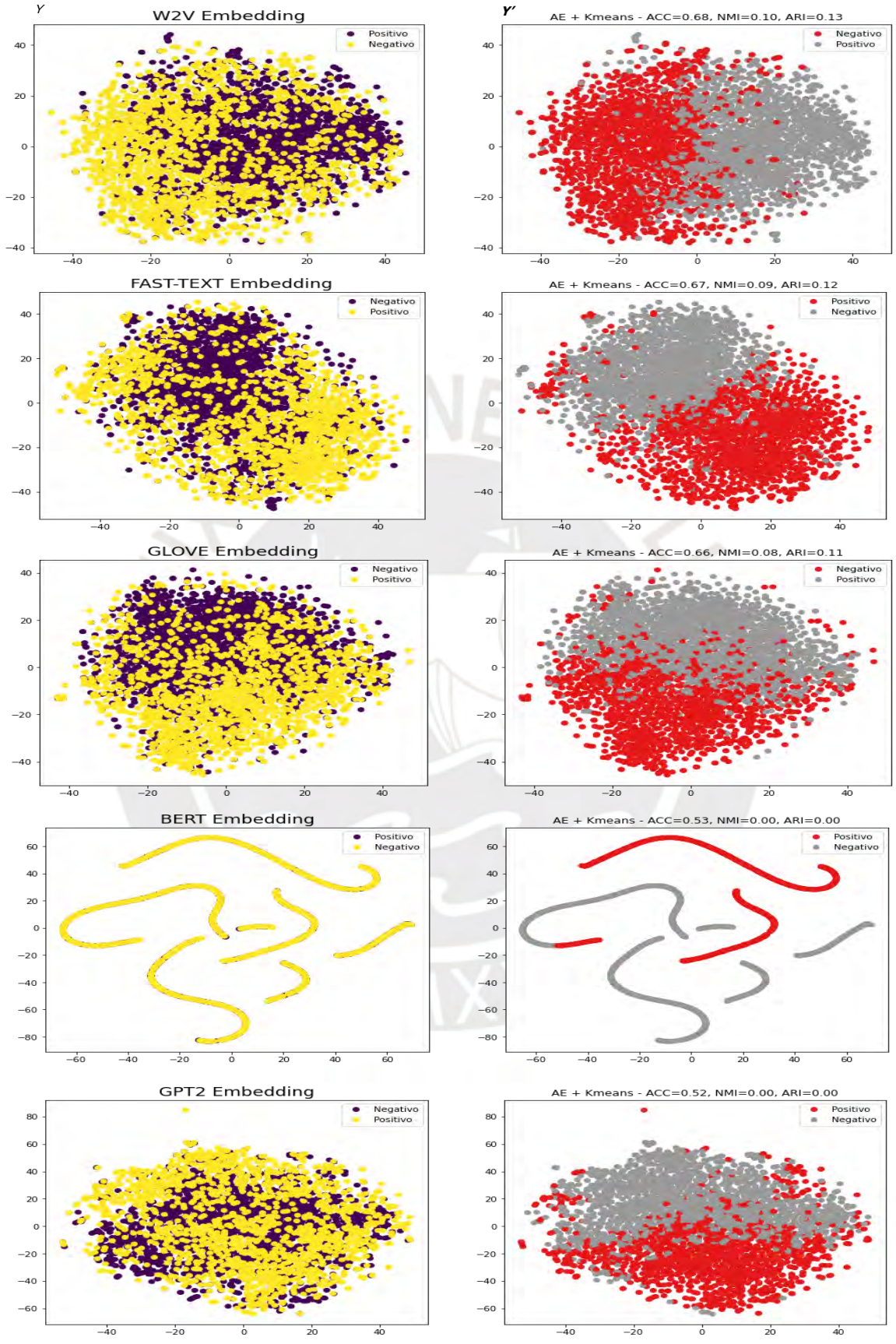
MODEL	EMBEDDING	ACC	NMI	ARI
AE K-MEANS	W2V	0.68	0.10	0.13
	Fast-Text	0.67	0.09	0.12
	Glove	0.66	0.08	0.11
	Bert	0.53	0.00	0.00
	Gpt2	0.52	0.00	0.00
K-MEANS	W2v	0.68	0.10	0.13
	Glove	0.65	0.07	0.10
	Fast-Text	0.65	0.07	0.09
	Bert	0.54	0.00	0.00
	Gpt2	0.51	0.00	0.00
HDBSCAN	Fast-Text	0.54	0.02	0.01
	Gpt2	0.51	0.01	0.00
	Glove	0.50	0.00	0.00
	W2v	0.50	0.00	0.00
	Bert	0.50	0.00	0.00

Fuente: Elaboración propia.

En la Tabla 10, W2V Embedding tiene una mejor precisión, aunque las puntuaciones NMI y ARI fueron muy bajas. El modelo de AutoEncoder no pudo encontrar ninguna mejora sobre el K-means base. El modelo HDBSCAN basado en el ruido también tuvo un bajo rendimiento.

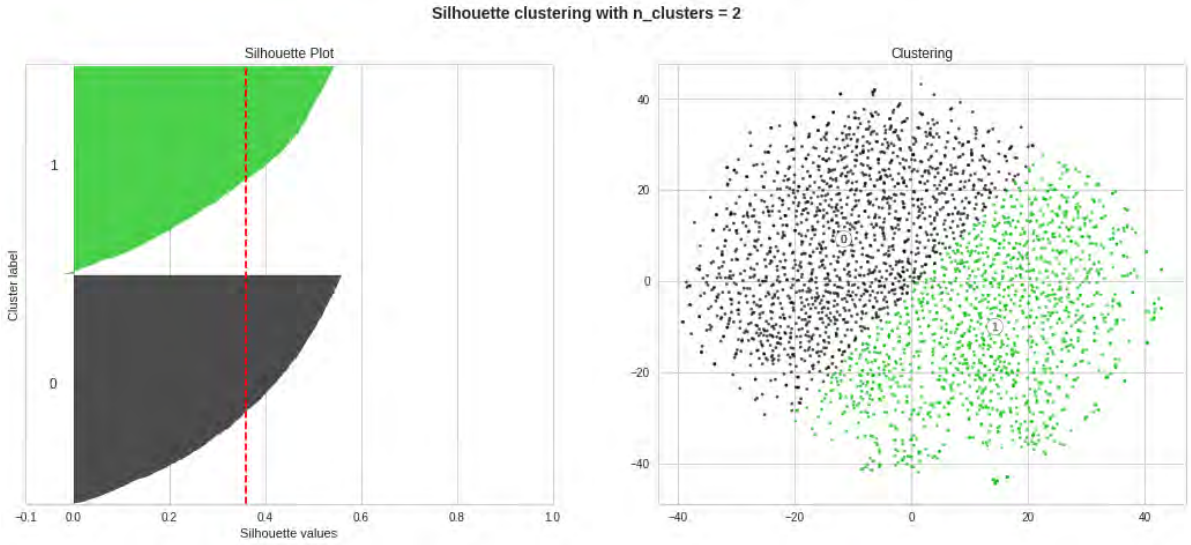
La distribución de Y en la Figura 14 muestra que la información no está claramente agrupada, lo que también podría ser un indicador para comprobar cómo se comporta el algoritmo AutoEncoder en Y'.

Figura 14: Clustering IBDB Reviews.



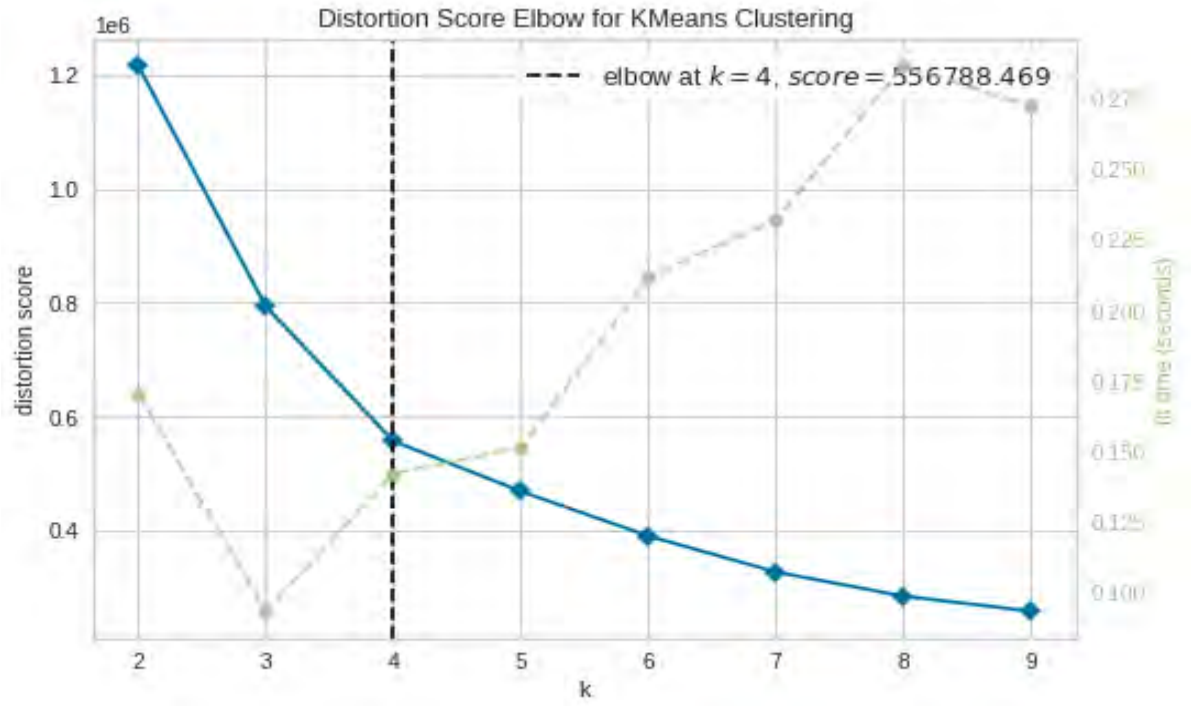
Fuente: Elaboración propia.

Figura 15: Silhouette Clustering - IBDB Reviews.



Fuente: Elaboración propia.

Figura 16: Elbow - IBDB Reviews.



Fuente: Elaboración propia.

En la Figura 15, se visualiza la distribución del método Silhouette con dos clústeres, comparando con la Figura 16 al aplicar el método Elbow, se aprecia que la cantidad de clústeres difiere un poco, esto podría ser debido a que la data etiquetada presenta datos que con ruido que confunde al algoritmo de clusterización.

DATASET 3: Peruvian Food Reviews.

Tabla 11: Resultados de los modelos supervisados - Peruvian Food Reviews

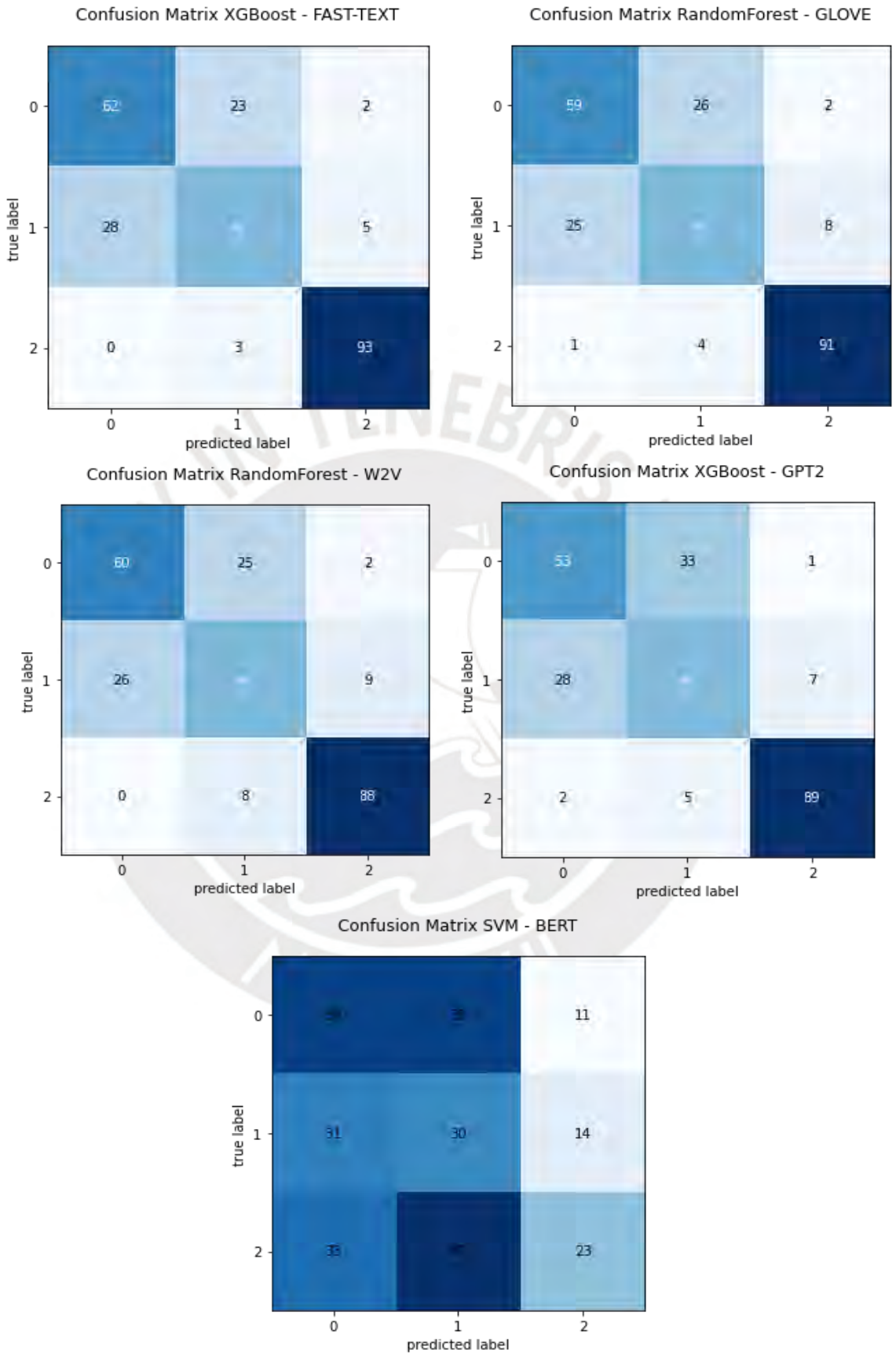
Embedding	Model	Accuracy	Precision	Recall	F1 Score
FAST-TEXT	XGBoost	0.764	0.758	0.764	0.760
	RandomForest	0.756	0.749	0.756	0.752
	SVM	0.752	0.749	0.752	0.750
	LGBM	0.748	0.739	0.748	0.742
GLOVE	RandomForest	0.744	0.739	0.744	0.741
	XGBoost	0.733	0.736	0.733	0.734
	SVM	0.733	0.722	0.733	0.724
	LGBM	0.709	0.707	0.709	0.708
W2V	RandomForest	0.729	0.725	0.729	0.727
	LGBM	0.721	0.716	0.721	0.718
	XGBoost	0.705	0.706	0.705	0.706
	SVM	0.705	0.707	0.705	0.706
GPT2	XGBoost	0.705	0.706	0.705	0.705
	LGBM	0.702	0.703	0.702	0.702
	RandomForest	0.651	0.649	0.651	0.650
	SVM	0.399	0.740	0.399	0.332
BERT	SVM	0.353	0.385	0.353	0.350
	LGBM	0.345	0.349	0.345	0.346
	XGBoost	0.341	0.343	0.341	0.340
	RandomForest	0.341	0.343	0.341	0.340

Fuente: Elaboración propia.

En la Tabla 11, FAST-TEXT Embedding es más exitoso cuando se aplica a un modelo supervisado basado en Boosting XGBoost. Sin embargo, RandomForest con GLOVE también logra una predicción muy alta. El Embedding de Bert es el que menos Accuracy muestra en su resultado, este podría descartar como modelo de bajo rendimiento porque la predicción es muy baja.

En la Figura 17, La matriz de confusión de los modelos tiende a tener un margen de error en las predicciones.

Figura 17: Matriz de confusión - Peruvian Food Reviews.



Fuente: Elaboración propia.

Tabla 12: Resultados de los modelos no supervisados - Peruvian Food Reviews.

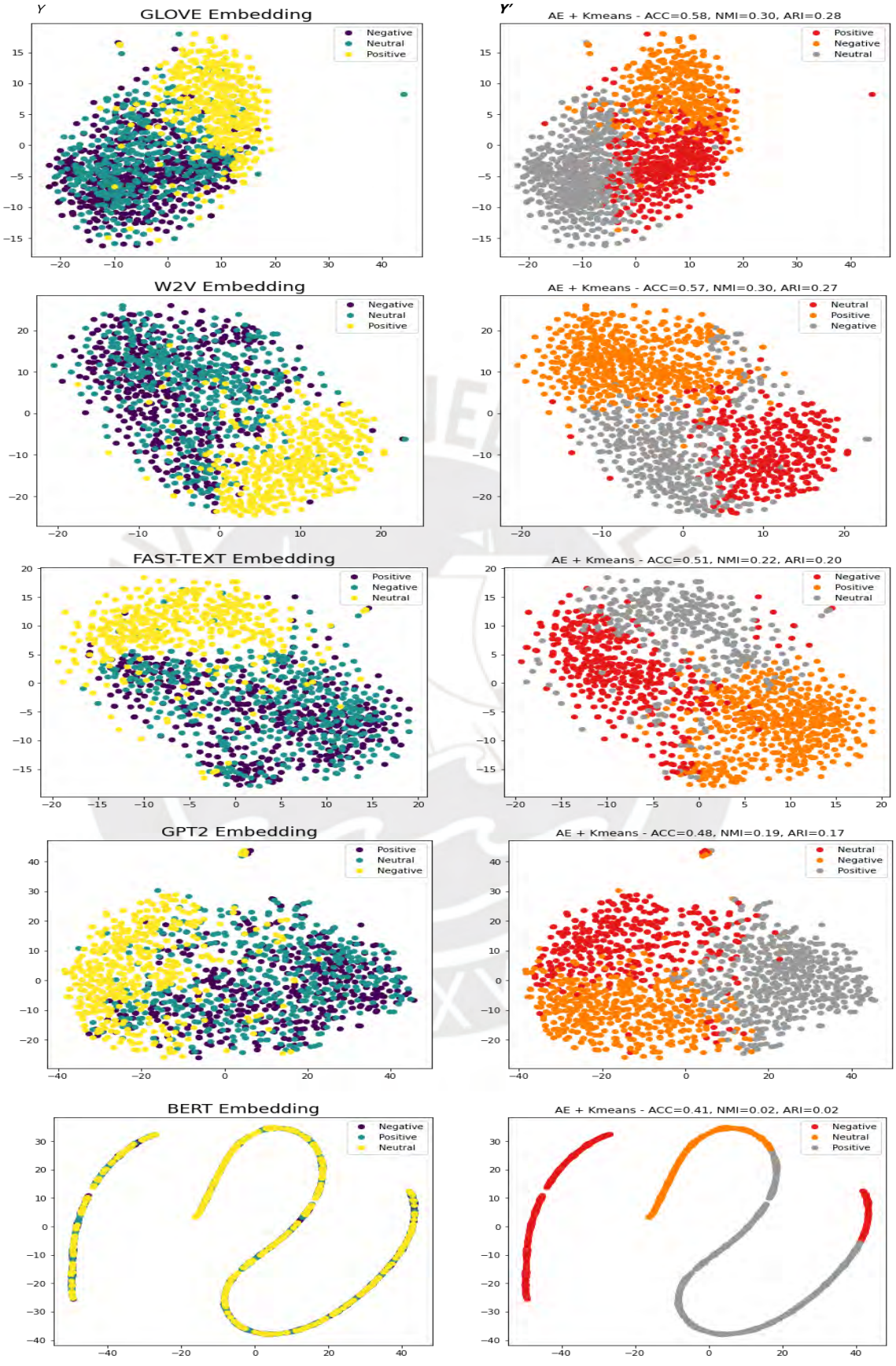
MODEL	EMBEDDING	ACC	NMI	ARI
AE K-MEANS	Glove	0.58	0.31	0.28
	W2v	0.57	0.30	0.27
	Fast-Text	0.51	0.23	0.21
	Gpt2	0.48	0.19	0.17
	Bert	0.41	0.02	0.02
K-MEANS	Glove	0.57	0.27	0.28
	Fast-Text	0.54	0.21	0.23
	W2v	0.53	0.20	0.21
	Gpt2	0.47	0.20	0.18
	Bert	0.41	0.02	0.02
HDBSCAN	Gpt2	0.35	0.02	0.00
	Fast-Text	0.34	0.01	0.00
	Glove	0.34	0.01	0.00
	W2v	0.34	0.02	0.00
	Bert	0.00	0.00	0.00

Fuente: Elaboración propia.

En la Tabla 12, Glove Embedding tiene un poco más de éxito cuando se aplica al modelo AutoEncoder. Sin embargo, las puntuaciones NMI y ARI son bastante bajas. El conjunto de datos tiene tres categorías, en la cual, si se observa la distribución en Y de la Figura 18, la categoría negativa no está bien distribuida.

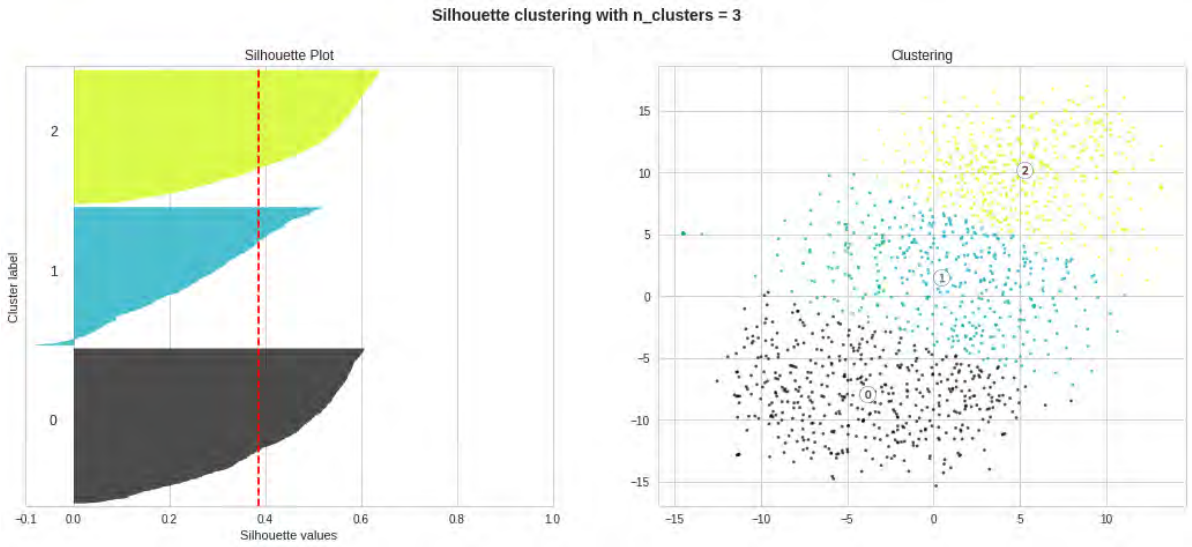
Así mismo podemos contrastar estos resultados en la matriz de confusión de la Figura 17.

Figura 18: Clustering - Peruvian Food Reviews.



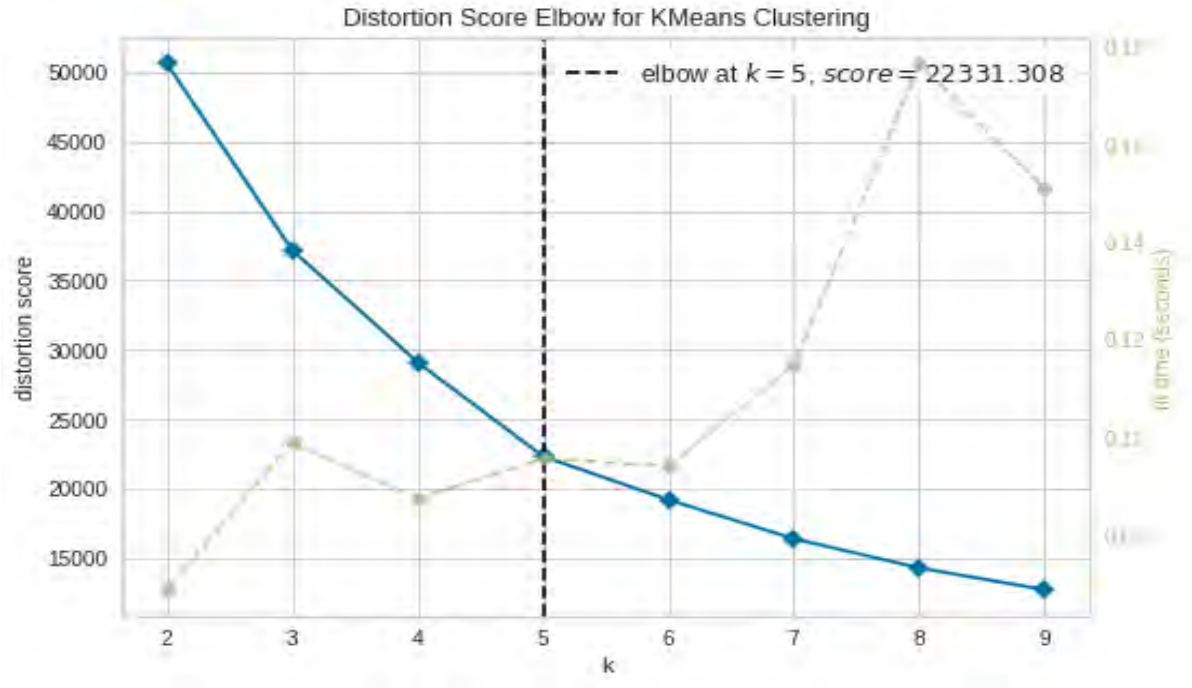
Fuente: Elaboración propia.

Figura 19: Silhouette Clustering - Peruvian Food Reviews.



Fuente: Elaboración propia.

Figura 20: Elbow - Peruvian Food Reviews.



Fuente: Elaboración propia.

En la Figura 19, se visualiza la distribución del método Silhouette con tres clústeres en la cual se puede observar que una de las variables se aleja un poco del resto, comparando con la Figura 20 al aplicar el método Elbow, se puede observar que el número de clústeres óptimo es 5, esto podría ser debido a que la data etiquetada en una de las etiquetas podría ser confundida con otra etiqueta.

DATASET 4: Consumer Complaint.

Tabla 13: Resultados de los modelos supervisados - Consumer Complaint.

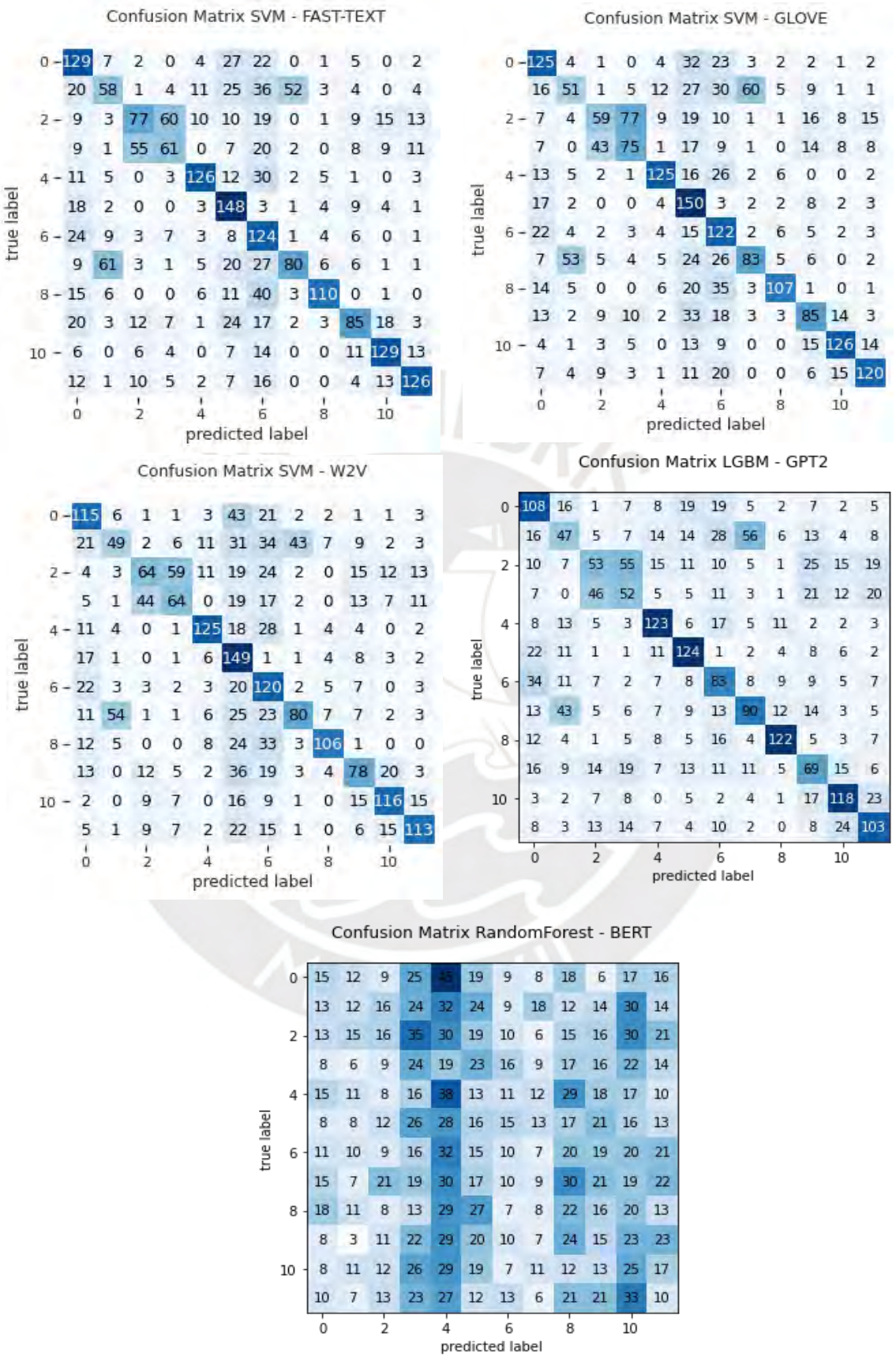
Embedding	Model	Accuracy	Precision	Recall	F1 Score
FAST-TEXT	SVM	0.522	0.545	0.522	0.519
	LGBM	0.517	0.512	0.517	0.512
	XGBoost	0.500	0.486	0.500	0.489
	RandomForest	0.435	0.424	0.435	0.424
GLOVE	SVM	0.512	0.533	0.512	0.507
	LGBM	0.496	0.487	0.496	0.489
	XGBoost	0.490	0.479	0.490	0.480
	RandomForest	0.430	0.419	0.430	0.420
W2V	SVM	0.491	0.520	0.491	0.488
	LGBM	0.478	0.468	0.478	0.470
	XGBoost	0.462	0.449	0.462	0.452
	RandomForest	0.410	0.399	0.410	0.400
GPT2	LGBM	0.455	0.448	0.455	0.449
	XGBoost	0.431	0.421	0.431	0.421
	RandomForest	0.360	0.348	0.360	0.336
	SVM	0.273	0.385	0.273	0.258
BERT	RandomForest	0.088	0.089	0.088	0.085
	SVM	0.085	0.076	0.085	0.062
	LGBM	0.082	0.082	0.082	0.079
	XGBoost	0.082	0.081	0.082	0.079

Fuente: Elaboración propia.

En la Tabla 13, FAST-TEXT Embedding es más exitoso cuando se aplica a un modelo supervisado basado en vectores SVM. Sin embargo, LGBM también logra una predicción adecuada. El Embedding de Bert en este Dataset es que da peor Accuracy en su resultado, se podría descartar como modelo de bajo rendimiento debido a que la predicción es muy baja.

En la Figura 21, La matriz de confusión de los modelos en algunas variables tiende a tener un margen de error en las predicciones.

Figura 21: Matriz de confusión - Consumer Complaint.



Fuente: Elaboración propia.

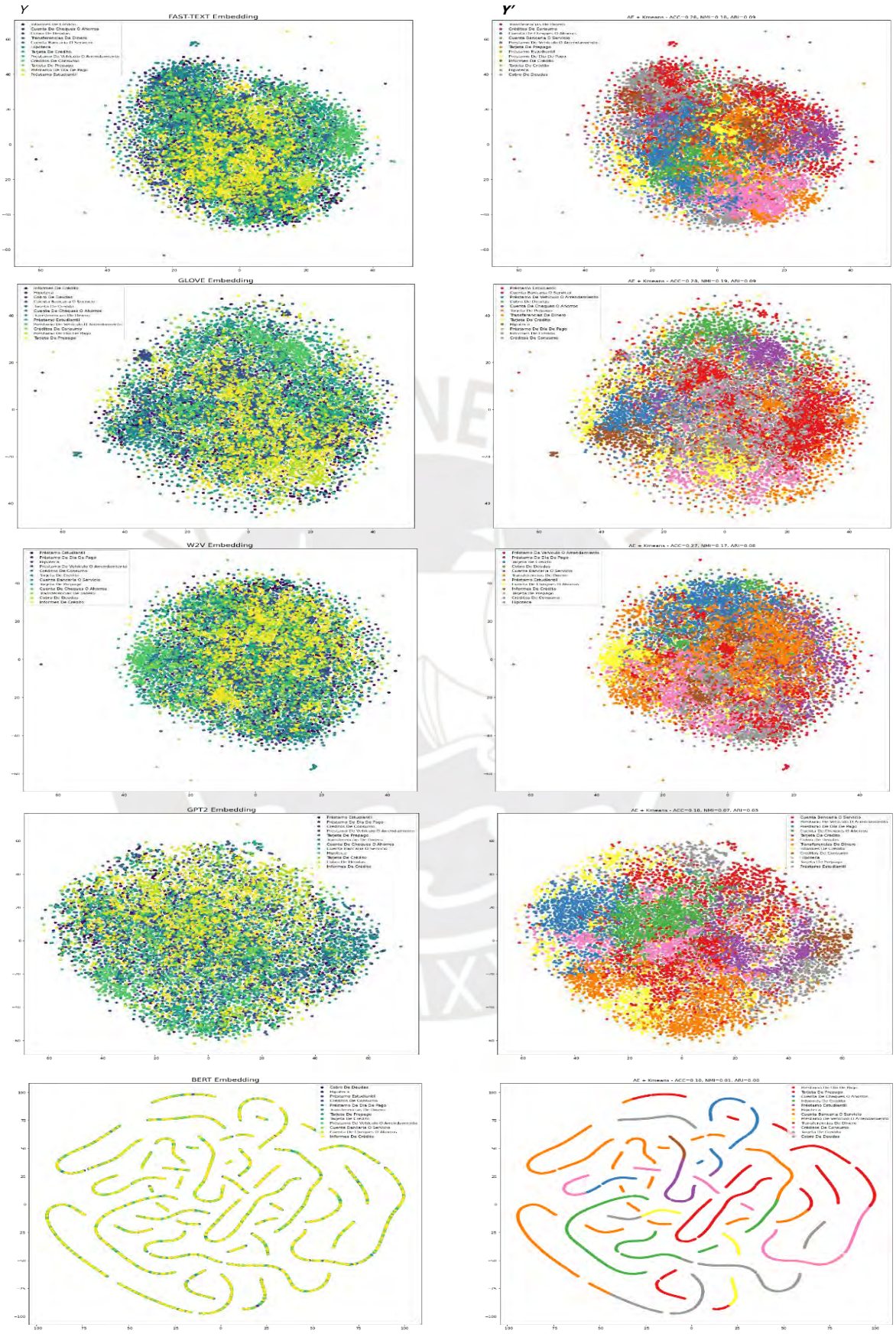
Tabla 14: Resultados de los modelos no supervisados - Consumer Complaint.

MODEL	EMBEDDING	ACC	NMI	ARI
AE K-MEANS	Fast-Text	0.28	0.18	0.09
	Glove	0.28	0.19	0.09
	W2v	0.27	0.17	0.08
	Gpt2	0.19	0.07	0.03
	Bert	0.10	0.01	0.00
K-MEANS	Glove	0.21	0.10	0.05
	Fast-Text	0.20	0.09	0.04
	W2v	0.20	0.08	0.04
	Gpt2	0.18	0.06	0.03
	Bert	0.10	0.01	0.00
HDBSCAN	Gpt2	0.10	0.02	0.00
	Glove	0.09	0.02	0.00
	W2v	0.09	0.01	0.00
	Fast-Text	0.09	0.02	0.00
	Bert	0.09	0.00	0.00

Fuente: Elaboración propia.

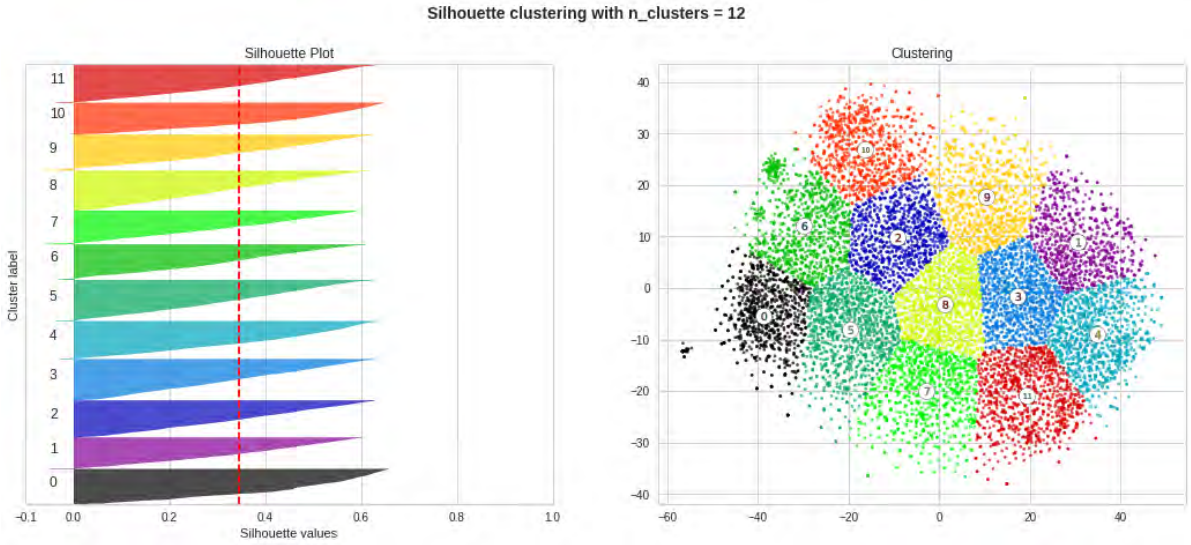
En la Tabla 14 los resultados son realmente bajos, sólo Fast-Text fue capaz de conseguir un rendimiento un poco aceptable. La causa de este mal resultado podría ser que la distribución Y de las categorías en la Figura 22 están demasiado acopladas, mostrando mucho ruido. Esto también se puede contrastar en la matriz de confusión de los modelos supervisado en la Figura 21.

Figura 22: Clustering - Consumer Complaint.



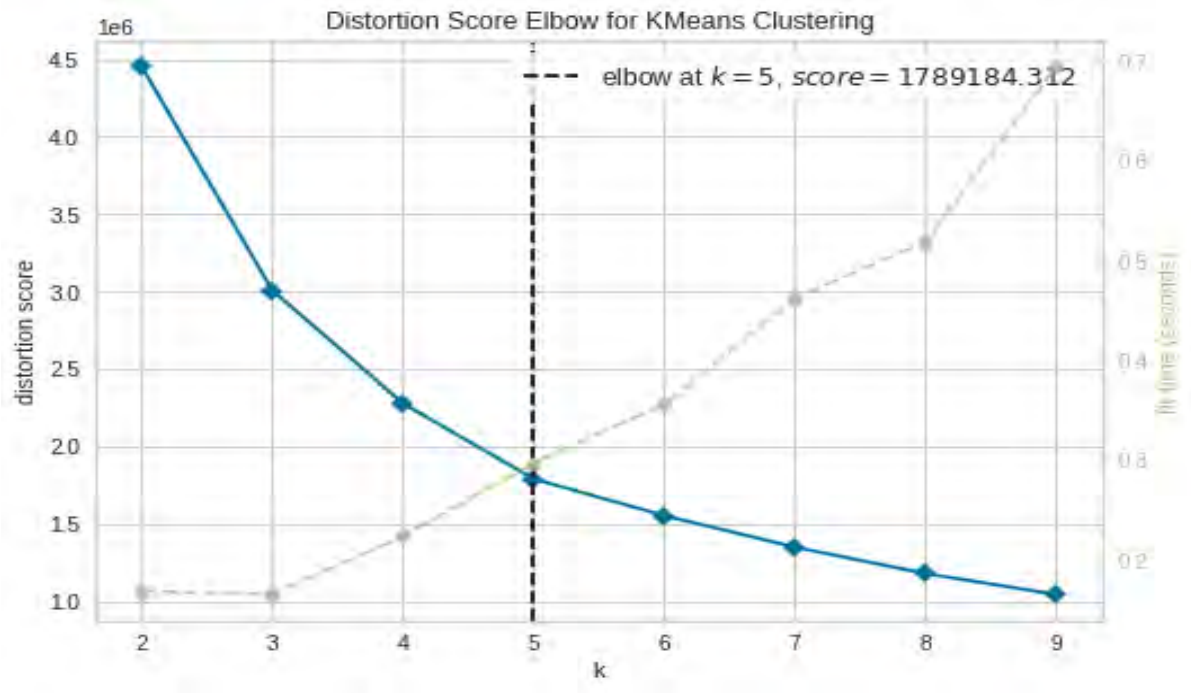
Fuente: Elaboración propia.

Figura 23: Silhouette Clustering - Consumer Complaint.



Fuente: Elaboración propia.

Figura 24: Elbow - Consumer Complaint.



Fuente: Elaboración propia.

En la Figura 23, se visualiza la distribución del método Silhouette con doce clústeres, comparando con la Figura 24 al aplicar el método Elbow, se puede observar que el número de clústeres óptimo es cinco. Sin embargo, esto se encuentra muy alejado, representando que las variables en la data etiquetada tengan mucho ruido.

DATASET 5: Abstracts Scopus 2021.

Tabla 15: Resultados de los modelos supervisados - Abstracts Scopus 2021.

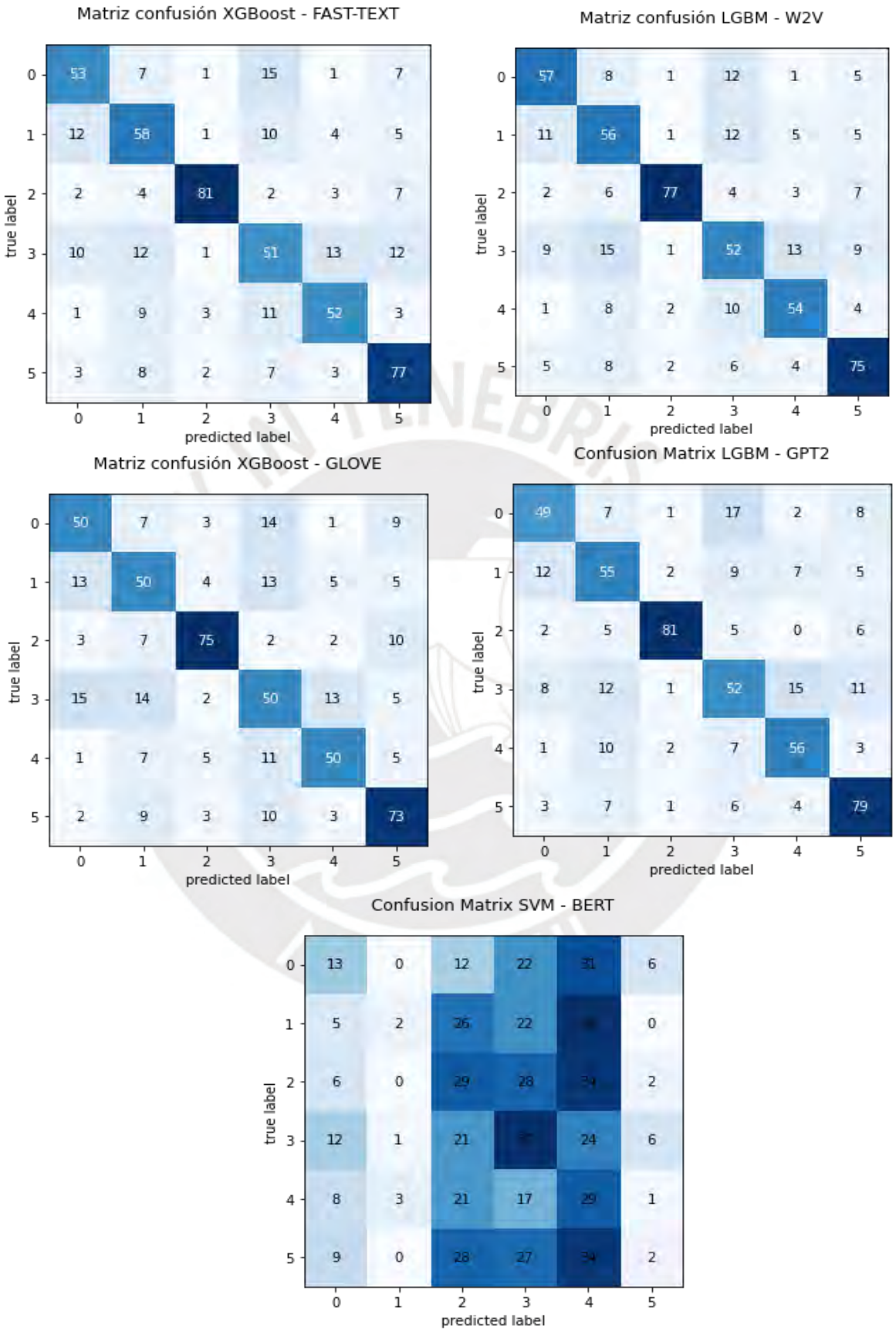
Embedding	Model	Accuracy	Precision	Recall	F1 Score
FAST-TEXT	XGBoost	0.697	0.701	0.697	0.698
	LGBM	0.697	0.702	0.697	0.699
	RandomForest	0.661	0.668	0.661	0.663
	SVM	0.652	0.685	0.652	0.660
W2V	LGBM	0.688	0.690	0.688	0.687
	XGBoost	0.675	0.677	0.675	0.674
	SVM	0.652	0.681	0.652	0.658
	RandomForest	0.633	0.639	0.633	0.634
GLOVE	LGBM	0.679	0.680	0.679	0.678
	XGBoost	0.646	0.649	0.646	0.647
	SVM	0.641	0.670	0.641	0.647
	RandomForest	0.635	0.638	0.635	0.635
GPT2	LGBM	0.675	0.679	0.675	0.676
	XGBoost	0.673	0.678	0.673	0.675
	RandomForest	0.637	0.645	0.637	0.638
	SVM	0.427	0.681	0.427	0.412
BERT	SVM	0.200	0.215	0.200	0.168
	XGBoost	0.192	0.188	0.192	0.190
	RandomForest	0.192	0.188	0.192	0.190
	LGBM	0.171	0.176	0.171	0.172

Fuente: Elaboración propia.

En la Tabla 15, FAST-TEXT Embedding es más exitoso cuando se aplica a un modelo supervisado basado en boosting XGBoost. Sin embargo, LGBM también logra una predicción adecuada. El Embedding de Bert en este Dataset es que da peor Accuracy en su resultado, se podría descartar como modelo de bajo rendimiento debido a que la predicción es muy baja.

En la Figura 25, La matriz de confusión de los modelos es un poco aceptable, tiene un margen de error no tan alto.

Figura 25: Matriz de confusión - Abstracts Scopus 2021.



Fuente: Elaboración propia.

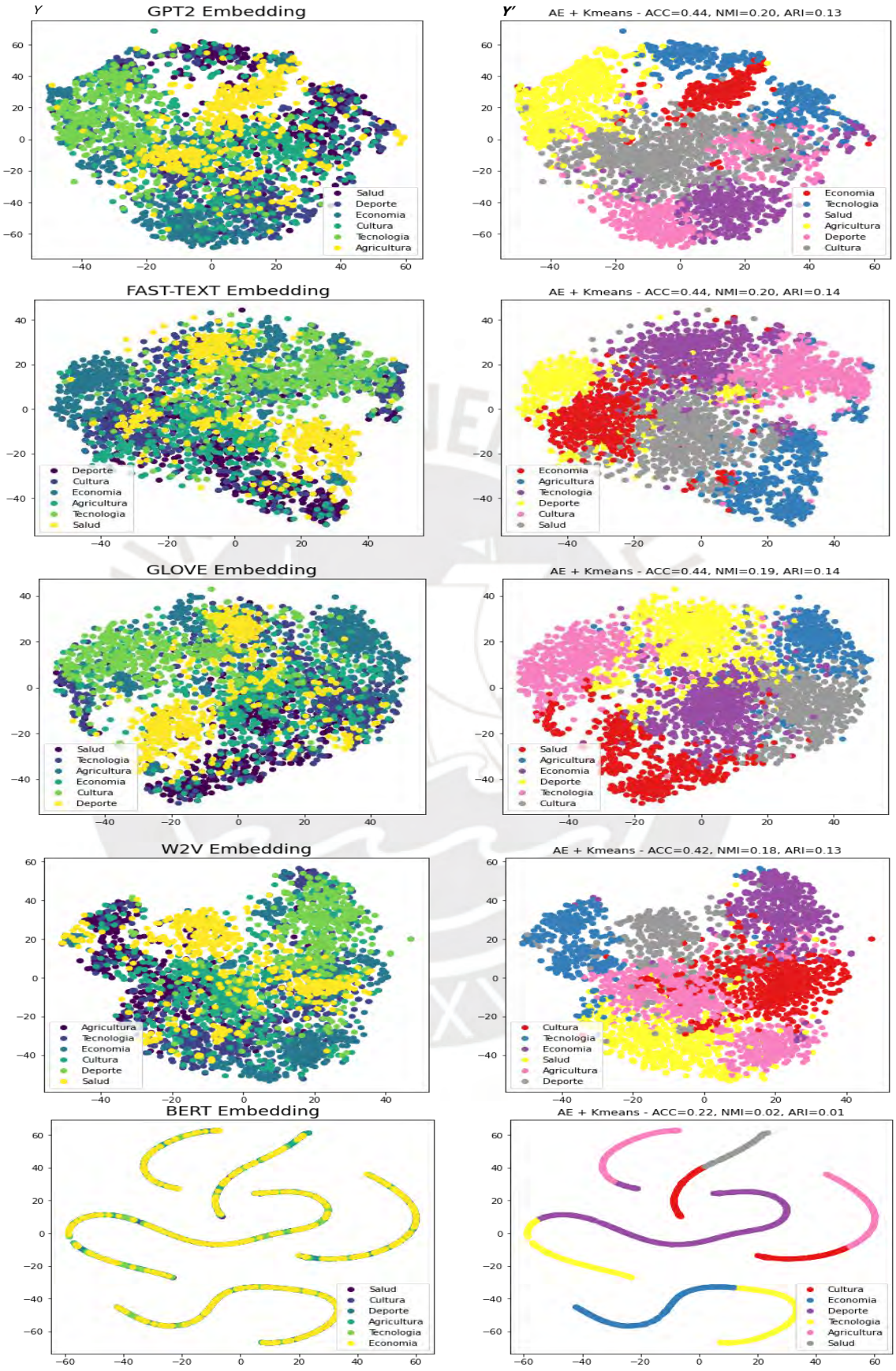
Tabla 16: Resultados de los modelos no supervisados - Abstracts Scopus 2021.

MODEL	EMBEDDING	ACC	NMI	ARI
AE K-MEANS	Gpt2	0.44	0.20	0.13
	Fast-Text	0.44	0.20	0.14
	Glove	0.44	0.20	0.14
	W2v	0.42	0.18	0.13
	Bert	0.22	0.02	0.01
K-MEANS	Glove	0.42	0.19	0.13
	Gpt2	0.41	0.19	0.12
	W2v	0.41	0.19	0.13
	Fast-Text	0.39	0.18	0.12
	Bert	0.21	0.02	0.01
HDBSCAN	Gpt2	0.20	0.04	0.00
	Glove	0.20	0.03	0.00
	Fast-Text	0.20	0.03	0.00
	W2v	0.20	0.03	0.00
	Bert	0.19	0.01	0.00

Fuente: Elaboración propia.

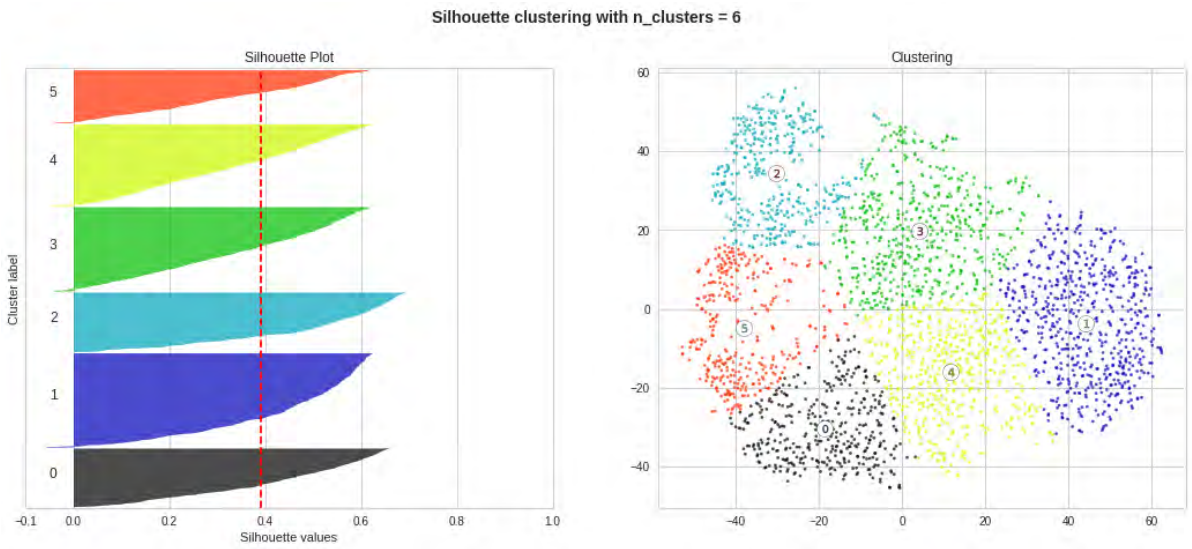
En la Tabla 16, GPT2 Embedding es ligeramente más exitoso cuando se aplica al modelo de AutoEncoder. Sin embargo, las puntuaciones NMI y ARI son bastante bajas en lo que respecta al Accuracy. El conjunto de datos se etiquetó en seis categorías, en las que la distribución Y de la Figura 26 muestra que algunas categorías no están bien agrupadas. Este es otro claro ejemplo de cómo se comporta el modelo de AutoEncoder en datos con mucho ruido

Figura 26: Clustering - Abstracts Scopus 2021.



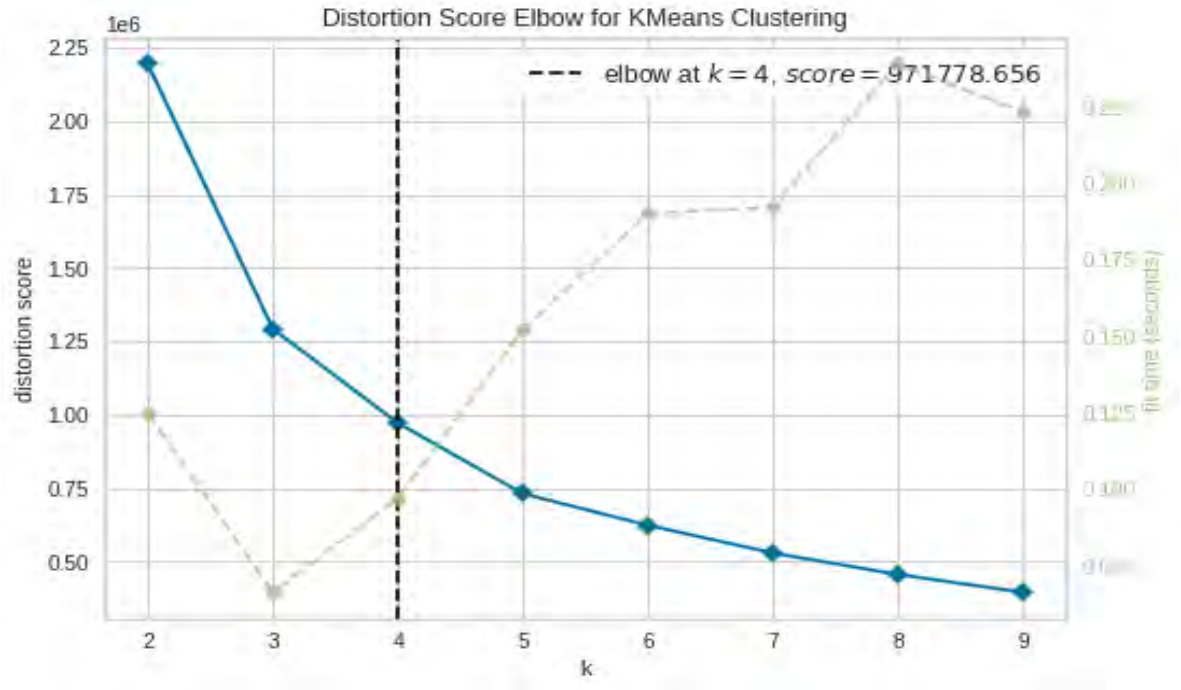
Fuente: Elaboración propia.

Figura 27: Silhouette Clustering - Abstracts Scopus 2021.



Fuente: Elaboración propia.

Figura 28: Elbow - Abstracts Scopus 2021.



Fuente: Elaboración propia.

En la Figura 27, se visualiza la distribución del método Silhouette con seis clústeres, en la cual se puede observar que algunas variables se alejan un poco del resto, comparando con la Figura 28 al aplicar el método Elbow, se puede observar que el número de clústeres óptimo es cuatro. Sin embargo, esto no se encuentra tan alejado del número de categorías.

Tabla 17: Resumen de resultados de los modelos supervisados.

Dataset	Model	Embedding	Accuracy	Precision	Recall	F1 Score
BBC News	**LGBM**	**GLOVE**	0.97	0.97	0.97	0.97
IBDB Reviews	**LGBM**	**FAST-TEXT**	0.85	0.85	0.85	0.85
Peruvian Food Reviews	**XGBoost**	**FAST-TEXT**	0.76	0.76	0.76	0.76
Abstracts Scopus 2021	**XGBoost**	**FAST-TEXT**	0.70	0.70	0.70	0.70
Consumer Complaint	**SVM**	**FAST-TEXT**	0.52	0.55	0.52	0.52

Fuente: Elaboración propia.

Tabla 18: Resumen de los resultados de los modelos no supervisados.

Dataset	Model	Embedding	Accuracy	NMI	ARI
BBC News	AE + K-MEANS	GLOVE	0.92	0.79	0.81
IBDB Reviews	AE + K-MEANS	W2V	0.68	0.10	0.13
Peruvian Food Reviews	AE + K-MEANS	GLOVE	0.58	0.31	0.28
Abstracts Scopus 2021	AE + K-MEANS	GPT2	0.44	0.20	0.13
Consumer Complaint	AE + K-MEANS	FAST-TEXT	0.28	0.18	0.09

Fuente: Elaboración propia.

Tabla 19: Resumen de los puntajes de Clustering.

Dataset + Embedding	Labels	Silhouette		Calinski Harabasz		Elbow	
		Clusters	Score	Clusters	Score	Clusters	Inertia
BBC News + Glove	5	5	0.54	5	3998.94	5	368121.1
Imdb Reviews + W2v	2	3	0.37	7	2607.36	4	566207.9
Food Reviews + Glove	3	2	0.51	2	2023.09	5	22332.4
Paper Scopus + Gpt2	6	3	0.43	11	3242.86	4	607575.7
Complaints + Fast text	12	2	0.43	2	12044.01	5	1789223

Fuente: Elaboración propia.

Capítulo 5.

Conclusiones y Trabajos Futuros.

5.1. Conclusiones.

La recopilación de cinco conjuntos de datos de diferentes fuentes ha permitido analizar las técnicas de PLN para la limpieza y normalización de los datos. La librería utilizada para procesar el texto es NLTK en español. Se eliminaron los duplicados, los nulos, los StopWords y los caracteres especiales. Para normalizar los datos se utilizaron LabelEncoder y StandardScaler.

La generación de Embeddings en los Datasets permitió la aplicación de cinco Embeddings: W2V, Glove, FastText, Bert y GPT2, que dio como resultado y mejor desempeño en el Dataset de noticias "BBC News". Siendo Glove el Embedding de mayor calidad aplicado al modelo supervisado LGBM con un Accuracy de 0.97. y aplicado en un modelo no supervisado AutoEncoder + K-means con una precisión de 0.92 NMI 0.79 y ARI 0.81.

En consecuencia, para todos los conjuntos de datos es complicado saber qué Embedding se comporta mejor porque, como se pudo observar en los resultados, la agrupación correcta depende del tipo de procesamiento de la información, debido a que los textos pueden tener semánticas en común. El Embedding que obtuvo mayor resultado fue FastText implementado con Gensim y aplicado en modelos supervisados basados en boosting. Por otro lado, se puede determinar que la extracción de Embedding basados en Gensim y aplicados a modelos no supervisados de redes neuronales con AutoEncoder y capa K-means dan mejores resultados que los basados en Transformers como Bert y GPT2. El número de categorías afectó al resultado de los modelos, debido a que algunas categorías tenían una semántica en común con otras. Esto se pudo determinar con los métodos de Elbow, Silhouette y Calinski Harabasz.

5.2. Trabajos futuros y recomendaciones.

El avance en el estudio del procesamiento del lenguaje natural ha propiciado la aparición de nuevas técnicas, sin embargo, el idioma español no está tan estandarizado

como el inglés, que ha avanzado mucho en esta área.

Para seguir mejorando el modelo del AutoEncoder, se recomienda trabajar con conjuntos de datos bien etiquetados, como BBC News, que no generen pérdidas en el momento de la validación. Por otro lado, para mejorar los demás resultados en el resto de conjuntos de datos presentados en esta investigación, se podrían aplicar diferentes pesos y hacer un fine-tuning añadiendo algunas capas más a los modelos. Es importante analizar los puntajes de los métodos de Elbow, Silhouette y Calinski Harabasz para validar si los clústeres elegidos son correctos y así descartar en un Dataset etiquetado que categorías permiten mejorar el modelo.

En contraste con otras técnicas de Clustering, podría ser beneficioso utilizar técnicas de Clustering basadas en la semántica y aplicar otra técnica de Embedding.

Bibliografía.

Peruvian Food Reviews (August 2021) – *Extraído de Kaggle del usuario Lázaro. https://www.kaggle.com/lazaro97/peruvian-food-reviews*

He, B., Ahamad, M., & Kumar, S. (2021, August). PETGEN: Personalized Text Generation Attack on Deep Sequence Embedding-based Classification Models. *In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (pp. 575-584).*

Wang, J., Ma, Z., Nie, F., & Li, X. (2021, June, p1.). Progressive Self-Supervised Clustering with Novel Category Discovery. *IEEE Transactions on Cybernetics.*

Liu, N., Wang, Q., & Ren, J. (2021). Label-Embedding Bi-directional Attentive Model for Multi-label Text Classification. *Neural Processing Letters, 53(1), 375-389.*

Ontañón, S., Ainslie, J., Cvicek, V., & Fisher, Z. (2021). Making Transformers Solve Compositional Tasks. *arXiv preprint arXiv:2108.04378.*

El-Alami, F. Z., El Alaoui, S. O., & Nahnahi, N. E. (2021). Contextual semantic embeddings based on fine-tuned AraBERT model for Arabic text multi-class categorization. *Journal of King Saud University-Computer and Information Sciences.*

Berny Josúe & Obregon Carrer (2021). Datificate gpt2 small Spanish model. *Retrieved 5 March 2022, from Hugging face website: https://metatext.io/models/datificate-gpt2-small-spanish*

Bureau, C. F. P. (2020, November). Consumer Financial Protection Bureau.

Dai, Z., Li, K., Li, H., & Li, X. (2020, October). An Unsupervised Learning Short Text Clustering Method. *In Journal of Physics: Conference Series (Vol. 1650, No. 3, p. 032090). IOP Publishing.*

Utia Deza, J. V. (2020, August, p.10) Representación vectorial de relación de hiponimia e hiperonimia en español.

Li, Y., & Ye, M. (2020, April). A Text Classification Model Base on Region Embedding AND LSTM. *In Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence (pp. 152-157).*

Bounabi, M., El Moutaouakil, K., & Satori, K. (2020, April). Neural Embedding & Hybrid ML Models for Text Classification*. In 2020 1st International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET) (pp. 1-6). IEEE.*

Shin, H. S., Kwon, H. Y., & Ryu, S. J. (2020). A new text classification model based on contrastive word embedding for detecting cybersecurity intelligence in twitter. *Electronics, 9(9), 1527.*

Zhou, M., Liu, D., Zheng, Y., Zhu, Q., & Guo, P. (2020*). A text sentiment classification model using double word embedding methods. Multimedia Tools and Applications, 1-20.*

Swysen Cachaña, T. B. (2020*).* Validation of vector representations of words.

Lee, J. S., & Hsiang, J. (2020). Patent claim generation by fine-tuning OpenAI GPT-2. *World Patent Information, 62, 101983.*

Kilimci, Z. H., & Akyokuş, S. (2019, September). The evaluation of word embedding models and deep learning algorithms for Turkish text classification. *In 2019 4th International Conference on Computer Science and Engineering (UBMK) (pp. 548-553). IEEE.*

Hadifar, A., Sterckx, L., Demeester, T., & Develder, C. (2019, August). A self-training approach for short text clustering. *In Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019) (pp. 194-199).*

Dong, Y., Liu, P., Zhu, Z., Wang, Q., & Zhang, Q. (2019). A fusion model-based label embedding and self-interaction attention for text classification. *IEEE Access, 8, 30548-30559.*

Liu, W., Liu, P., Yang, Y., Yi, J., & Zhu, Z. (2019). A< word, part of speech> embedding model for text classification. *Expert Systems, 36(6), e12460.*

Jinarat, S., Manaskasemsak, B., & Rungsawang, A. (2018, December). Short text clustering based on word semantic graph with word embedding model. *In 2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS) (pp. 1427-1432). IEEE.*

Perez Rojas, J. (2018). Spanish Word Embeddings. *Retrieved 4 March 2022, from GitHub website: https://github.com/dccuchile/spanish-word-embeddings*

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805.*

Cha, M., Gwon, Y., & Kung, H. T. (2017, November). Language modeling by clustering with word embeddings for text readability assessment. *In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 2003-2006).*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems, 30.*

Wei, T., Lu, Y., Chang, H., Zhou, Q., & Bao, X. (2015). A semantic approach for text clustering using WordNet and lexical chains. *Expert Systems with Applications, 42(4), 2264-2275.*

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. *In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).*

Le, Q., & Mikolov, T. (2014, June). Distributed representations of sentences and documents. *In International conference on machine learning (pp. 1188-1196). PMLR.*

Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. *EMNLP, 2014.*

Campello, R. J., Moulavi, D., & Sander, J. (2013, Abril). Density-based clustering is based on hierarchical density estimates. *In Pacific-Asia conference on knowledge discovery*

and data mining (pp. 160-172). Springer, Berlin, Heidelberg.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. *In HLT-NAACL, 2013.*

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011, June). Learning word vectors for sentiment analysis. *In Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies (pp. 142-150).*

Joyce, J. M. (2011). Kullback-leibler divergence. *In International encyclopedia of statistical science (pp. 720-722). Springer, Berlin, Heidelberg*

Awasthi, P., & Zadeh, R. B. (2010, December). Supervised Clustering. *In NIPS (Vol. 4, pp. 2-2).*

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research, 9(11).*

Greene, D., & Cunningham, P. (2006, June). Practical solutions to the problem of diagonal dominance in kernel document clustering. *In Proceedings of the 23rd international conference on Machine learning (pp. 377-384).*

Hotho, A., Staab, S., & Stumme, G. (2003, November). Ontologies improve text document clustering. *In Third IEEE international conference on data mining (pp. 541-544). IEEE.*

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. *In KDD (Vol. 96, No. 34, pp. 226-231).*

Milligan, G. W., & Cooper, M. C. (1986). A study of the comparability of external criteria for hierarchical cluster analysis. Multivariate behavioural research, 21(4), 441-458.

MacQueen, J. B. (1965). *On the Asymptotic Behaviour of k-means. California university Los Angeles western management science institute.*