# Evaluation on embeddings application for spanish automatic text clustering

*Evaluación de la aplicación de embeddings para el agrupamiento automático de textos en español*

Anthony Wainer Cachay-Guivin[1]* https://orcid.org/0000-0002-0096-0920

Recibido 10 de octubre de 2023, aceptado 29 de agosto de 2024
*Received: October 10, 2023* *Accepted: August 29, 2024*

**ABSTRACT**

The vast amount of information on the Internet, primarily composed of texts, makes clustering reliable information a complicated task. This research aims to improve the automatic clustering of Spanish texts by applying embeddings and unsupervised learning algorithms. Five datasets were used, and embedding generation techniques such as Word2Vec, FastText, Glove, BERT, and GPT-2 were applied. Models like K-means, HDBSCAN, and AutoEncoder combined with K-means were employed for clustering. The results showed that the AutoEncoder model combined with the K-means using Glove embeddings achieved superior performance with an accuracy of 0.92, NMI of 0.79, and ARI of 0.81 on the BBC News dataset. In other datasets, the results varied, but the AutoEncoder with the K-means model consistently outperformed other methods. We conclude that neural network models with AutoEncoder and K-means layer are highly effective for automatically clustering Spanish texts, especially when using high-quality embeddings like Glove.

Keywords: Natural processing language, model analysis, artificial intelligence, datasets.

***RESUMEN***

*La gran cantidad de información en Internet, compuesta principalmente por textos, hace que agrupar información confiable sea una tarea complicada. Esta investigación tiene como objetivo mejorar el agrupamiento automático de textos en español mediante la aplicación de embeddings y algoritmos de aprendizaje no supervisado. Se utilizaron cinco conjuntos de datos y se aplicaron técnicas de generación de embeddings como Word2Vec, FastText, Glove, BERT y GPT-2. Para el agrupamiento, se emplearon modelos de K-means, HDBSCAN y AutoEncoder combinado con K-means. Los resultados mostraron que el modelo de AutoEncoder combinado con K-means utilizando embeddings de Glove obtuvo un rendimiento superior con una precisión de 0,92, NMI de 0,79 y ARI de 0,81 en el conjunto de datos de BBC News. En otros conjuntos de datos, los resultados variaron, pero el modelo de AutoEncoder con K-means consistentemente superó a los otros métodos. Concluimos que los modelos de redes neuronales con AutoEncoder y capa de K-means son altamente efectivos para el agrupamiento automático de textos en español, especialmente cuando se utilizan embeddings de alta calidad como Glove.*

*Palabras clave: Lenguaje de procesamiento natural, análisis de modelos, inteligencia artificial, datasets.*

[1] Pontificia Universidad Católica del Perú, Escuela de Postgrado, Lima, Perú.
* Autor de correspondencia: anthony.cachay@pucp.edu.pe

## INTRODUCTION

Nowadays, a significant advancement of various techniques for text clustering [1], including their validation and application, makes our approaches increasingly more diverse. Having a model that only solves a specific task in a particular text language can become a challenge to process. The most recent and successful techniques use transfer learning through pre-trained embeddings, while traditional methods rely on vector representations [2].

One of the most advanced contemporary models is word2vec [3]. Algorithms based on this technique are constantly evolving, such as FastText [4] or BERT [5]. The most common text clustering models are K-means [6], Agglomerative Hierarchical Clustering [7], and DBScan [8]. Performance outcomes depend on the distances chosen and their normalisation method for the input data.

Text clustering is becoming a complicated task due to the lack of knowledge of our information, particularly when applied to the Spanish language. The focus of this research is to study some algorithms and labelled datasets to validate and obtain results.

The experiments carried out in this research analyze and assess embeddings extracted from Spanish datasets to compare which gives the best results when applied to text clustering models. Five datasets were used; three were downloaded from a public web, and the others were created using web scraping techniques.

The Clustering results were validated using the Clustering Accuracy [9], Normalized Mutual Information - NMI [10] and the Adjusted Rand Index - ARI [11].

The article starts by describing the embedding techniques and the datasets used. The evaluation metrics, the models applied, and the results are as follows.

### Related work

Text clustering has been widely studied in the field of natural language processing, for example, clustering methods [12], [13], [14], feature distance [15], [16], and feature selection [17], [18], [19]. Cluster validation [20] allows evaluations to be carried out and applied to specific tasks.

Embedding extraction has been used in natural language processing and text-mining applications [21].

Recent research has evaluated the use of models for extracting embeddings such as Word2Vec, FastText, Glove, Bert, and Elmo; when applied to texts such as the Turkish language, the outcome has not been satisfactory [22]. However, applied to the Polish language a favourable result was achieved in the FastText model [23]. Another study also showed embedding extraction when training models using FastText and Word2Vec [24].

### Word Embeddings

Embeddings are low-level vector representations that are based on semantic similarity and relatedness. The models used for embedding extraction are Bert, Glove, FastText, Word2Vec and GPT2.

### Bert

The BERT model architecture is an unsupervised bidirectional transformer encoder [25] published in the tensor library [5]. As the Spanish version of Bert, Beto was born and trained in the whole-word masking technique. Therefore, Figure 1 shows a

Source: Devlin *et al.* [5]

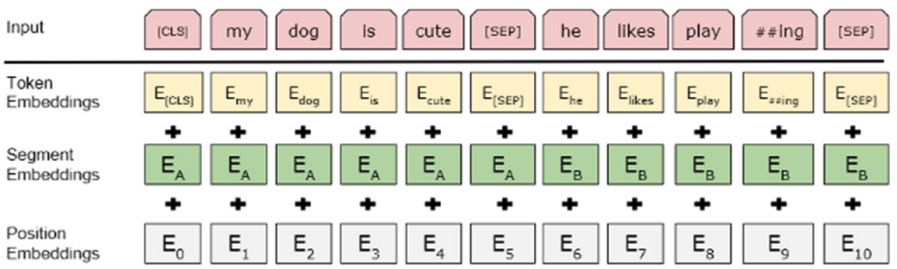


Figure 1. Representation of BERT input.

representation of the BERT model, where both token embeddings, segmentation and position are summed.

Input embeddings are the sum of token embeddings, segmentation embeddings and position embeddings. [5].

**Glove**

This algorithm is based on the occurrence matrices, representing the occurrence frequency of the words with each other. Glove follows equation (1) to get Word embeds:

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

Where element value $x_{lj}$ in the co-occurrence matrix $X$ corresponds to the number of times the word $l$ appears next to the word $j$ [23], [26].

The Glove model has 300 dimensions and 855 380 vectors applied to a Spanish corpus - Corpora [27].

**FastText**

FastText uses n-gram characters to represent words as the smallest element. Each word is divided into n-gram characters where: $3 \leq n \leq 6$. FastText learns from embeddings and can be used for text classification [23].

The FastText model has 300 dimensions and 1 313 423 vectors applied to a Spanish corpus. In addition, it was developed using the Corpora corpus, which has a weight of 3 billion words [27].

**Word2Vec**

Word2vec is probably the most integrated major model that started a new trend in distributed semantics. The Word2vec algorithm is based on applying a neural network model where it learns word associations with information from text corpora [23].

The Word2Vec model has 300 dimensions and 1 000 653 vectors applied to a Spanish corpus. In addition, it was developed using the Corpora corpus, which has a weight of 1.4 billion words [27].

**GPT-2**

GPT-2 was trained on Spanish Wikipedia using Transfer Learning and Fine-tuning techniques. The model was fitted from the pre-trained English model using the Hugging Face libraries (Transformers and Tokenizers) wrapped in the fastai v2 deep learning framework [28]. The data used for this model comes from Wikipedia in Spanish, which contains several unfiltered content [3].

## EXPERIMENTS

**Datasets**

Five datasets were used for the following research, which are presented below:

**IMDB**

This dataset contains the sentiment analysis of film reviews with 50 000 labelled reviews of high polarity. Each of the labelled opinions has a binary sentiment label, i.e., "pos" for a positive review or "neg" for a negative review [29]. Table 1 shows an example:

**BBC News**

Table 2 shows the dataset, which contains BBC News, and is classified into five categories (business, entertainment, politics, sports, and technology) and has 2225 data points [30].

**Peruvian Food Reviews**

This Spanish-language dataset contains food reviews of 8791 Peruvian restaurants, 20 categories and 1 160 666 reviews data [31]. In this example from Table 3, a polarity was applied using TextBlob:

**Consumer Complaint**

Table 4 shows the data set containing consumer complaints about financial products and services

Table 1. Sample extracted from IMDB.

| Sentiment | Quantity |
|---|---|
| Negative | 1500 |
| Positive | 1500 |

Table 2. Sample extracted from BBC News.

| Category | Quantity |
|---|---|
| Sport | 491 |
| Entertainment | 366 |
| Business | 500 |
| Policy | 394 |
| Technician | 334 |

Table 3. Sample extracted from Peruvian Food Reviews.

| Sentiment | Quantity |
|---|---|
| Negative | 430 |
| Neutral | 430 |
| Positive | 430 |

sent to companies for response, with 19 categories and 125 010 data points [31].

Table 4. Sample extracted from Consumer Complaint.

**Abstracts Scopus 2021**

Table 5 shows a self-made dataset extracted from the Scopus database containing summaries of scientific articles labelled in 6 categories (agriculture, culture, sports, economics, health, and technology) and 2 754 data points.

**Evaluation Metric**

Unsupervised evaluation metrics were used to validate the results. Clustering Accuracy, NMI and ARI.

**Clustering accuracy**

Equation (2) is used to obtain the clustering accuracy.

$$ACC = \frac{\sum_{i=1}^{n} \delta\left(l_i = m\left(c_i\right)\right)}{n} \tag{2}$$

Where $l_i$ is the true label, $c_i$ is the cluster mapping produced by the algorithm, and $m$ encompass all

Table 4. Sample extracted from Consumer Complaint.

| Product | Quantity |
|---|---|
| Debt collection | 1000 |
| Consumer credit | 1000 |
| Bank account or service | 1000 |
| Checking or savings account | 1000 |
| Mortgage | 1000 |
| Credit reports | 1000 |
| Payday loan | 1000 |
| Vehicle loan or lease | 1000 |
| Student loans | 1000 |
| Credit card | 1000 |
| Prepaid card | 1000 |
| Money transfers | 1000 |

Table 5. Scopus 2021 abstracts categories.

| Product | Quantity |
|---|---|
| Agriculture | 369 |
| Culture | 500 |
| Sport | 500 |
| Economy | 500 |
| Health | 408 |
| Technology | 477 |

possible one-to-one assignments between clusters and labels. The Hungarian algorithm [32] should be used to improve efficiency.

**Normalized Mutual Information**

Normalized mutual information (NMI) is a normalization of the mutual information score, in which the results are scaled between 0 (no mutual information) and 1 (perfect correlation).

$$NMI\left(\alpha,\beta\right) = \frac{I\left(\alpha,\beta\right)}{\sqrt{H\left(\alpha\right)H\left(\beta\right)}} \tag{3}$$

In equation (3), $I$ is the mutual information, and $H$ is the entropy. When the NMI value is higher, the closer the two divisions are. However, once the NMI reaches the maximum value, the cluster number is the optimal number for the model [33].

**Adjusted Rand Index**

The Rand index (equation (4)) calculates a similarity measure between two clusters by considering all pairs of samples and counting pairs where they are mapped into the same or different groupings in the predicted and actual clusters.

$$ARI = \frac{\left(RI - Expected\_RI\right)}{\max\left(RI\right) - Expected\_RI} \tag{4}$$

The ARI score is guaranteed to have a value close to 0 for random labelling, regardless of cluster numbers and samples, and precisely 1 when clusters are identical up to a permutation [11].

**Models**

The research used TensorFlow was used as the library, and the algorithm employed the AutoEncoder architecture over K-means. As seen in the figure, the hyperparameters were configured with six layers for the encoder and five for the decoder. The first layer

representing the dimension of the embeddings, had a weight of 300, while the other layers increased with weights of 500, 2000, 5000, and 10000. The final layer of the encoder was extracted into 100 features. The weights of the decoder layers were the inverse of those of the encoder.

In this case, it was considered to work with two models and a third model based on and re-adjusted using the K-means model.

**K-means**

James Macqueen introduced the k-means algorithm around 1965 to minimize the objective function. It starts from a set of "n" observations in "z" groups, in which each observation belongs to the group whose mean value is closest [12].

**HDBSCAN**

Also known as hierarchical spatial clustering based on the density of noisy applications, it is based on the DBSCAN algorithm [8]. It generates a complete clustering hierarchy from which a simplified hierarchy composed of only the most significant clusters can be easily extracted [34].

**AutorEncoder + K-means**

This algorithm is an application of the autoencoder neural network model used in unsupervised learning. It first encodes the data into a smaller dimension and then decodes it back into the original data.

The input data for the autoencoder are the embeddings; these data are passed to the encoder and processed by the activation function. The result of the encoding is fed into the decoder, and finally, the resulting vector is obtained after reconstruction. For the creation of the custom layer in TensorFlow, equation (5) and equation (7) are added:

- Calculation of the probability assigned to each central point of the cluster:

$$q_{ij} = \frac{\left(1+\left\|z_i - u_j\right\|\right)^{-1}}{\sum_{j'}\left(1+\left\|z_i - u_{j'}\right\|^2\right)^{-1}} \tag{5}$$

Based on the t-SNE algorithm [35], the similarity between the embedded points $z_i$ and the centroids $\mu_j$ are measured. The result $q_{ij}$ is the assignment of embeddings to centroids [36].

- Calculation of probabilities in the distribution (equation (6)).

$$p_{ij} = \frac{q_{ij}^2 / \sum_{i'} q_{i'j}}{\sum_{j'}\left(q_{ij}^2 / \sum_{i'} q_{i'j}\right)} \tag{6}$$

This calculation prevents large clusters from skewing the hidden feature space [37].

- Calculation of loss function – KL Divergence

$$L = KL\left(P\middle\|Q\right) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{7}$$

In the autoencoder architecture (Figure 2), six layers were configured for the encoder and five for the decoder. The first layer has a weight of 300, representing the embedding dimension, and the other layers keep increasing from 500, to 2000, 5000, and 10000. Then, the final layer of the encoder is extracted from 100 features. The weights of the decoder layers are the inverse of the encoder.

The algorithm below was used to cluster the texts. The algorithm's input is the Embedded Texts with their respective dimensions. In the first and second steps, the autoencoder model is trained with Embeddings (Figure 1). In the third step, the characteristic vectors of the encoder layer are extracted. In the fourth and fifth steps, the K-means model is trained with the original embeddings, and then the weights assigned to each cluster centroid are extracted. In the sixth step, the customised layer is configured on TensorFlow, in which the vectors extracted from the Encoder are passed as input. In step seven, the weights extracted from the K-means training result are assigned. In step eight, the equations are assigned equation (5) and equation (7), in steps nine to twenty-three, are trained and predicted in batches, set up 100 iterations, a batch size of 512, an update interval of 140 and a tolerance threshold of 1e-3. Once the batch training is done, the results are validated using the performance metrics shown in Figure 3.

## RESULTS

Clustering performance was evaluated based on the correspondence between clusters and partitions according to the class labels assigned to each text.

Source: Self-made

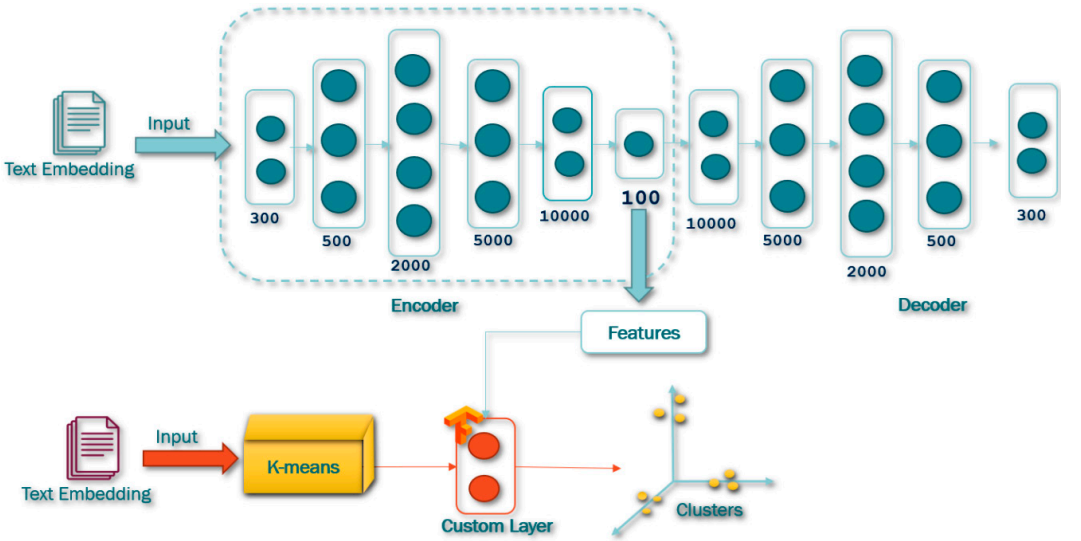


Figure 2. Autoencoder + K-means clustering.

| **Algorithm 1: Autoencoder + K-means Clustering** |
|---|
| **Result**: Enhancing clustering |
| **Input**: Embeddings with their vector dimensions |
| 01 **Pass** the embeddings ***E*** to the autoencoder; |
| 02 **Train** autoencoder model; |
| 03 **Extract** result vectors from the encoder layer ***RE***; |
| 04 **Pass** ***E*** to K-means model; |
| 05 **Extract** the weights *W* from the K-means result ***RK***; |
| 06 **Configure** ***RE*** as output in the custom layer; |
| 07 **Configure** ***W*** in the custom layer; |
| 08 **Configure** Eq. 05 and Eq. 07 in the custom layer; |
| 09 **Set** the iterations number ***I*** = *100I = 100*; |
| 10 **Set** batch size ***BS*** = *512BS = 512;* |
| 11 **Set** update interval ***UI*** = *140 UI = 140*; |
| 12 **Set** the tolerance threshold to stop training ***T*** = *1e – 3T = 1e – 3*; |
| 13 **For** ***iter*** in range from elements of ***E*** **do**; |
| 14 \| **If** the rest between ***iter*** and ***UI*** is 0 **then;** |
| 15 \| \| **Define** *q* with result on predict model with ***E***; |
| 16 \| \| **Update** auxiliary target distribution with ***q*** and **Eq. 06**; |
| 17 \| \| **Calculate** delta label from ***sum (q! = RK)/q***; |
| 18 \| \| **If** ***iter*** > 0 and delta label < ***T*** **then;** |
| 19 \| \| **I** ***break;*** |
| 20 \| \| **end*;*** |
| 21 \| **end*;*** |
| 22 \| **Train** on batch assign ***BS;*** |
| 23 **end*;*** |
| 24 **Validate** results with metrics; |

Figure 3. Hybrid clustering process.

In Table 6, Glove Embedding is slightly more successful when applied to the Autoencoder model. However, with a baseline, K-means GPT2 also achieves an appropriate prediction, slightly better than Glove.

From the distribution in Figure 4, the variable ***Y*** is highly correlated to the variable ***Y'.***

In Table 7, W2V Embedding has a better Accuracy, although the NMI and ARI scores were very low. The autoencoder model could not find any improvement over the base K-means. The noise-based HDBSCAN model also performed poorly.

Table 6. BBC News dataset results.

| 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 |

Table 7. IBDB dataset results.

| 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 |

The ***Y*** distribution in Figure 5 shows that the information is not clearly clustered; this could also be an indicator to check how the autoencoder algorithm behaves in ***Y'***.

In Table 8, Glove Embedding is slightly more successful when applied to the Autoencoder model. However, the NMI and ARI scores are quite low. The dataset was labelled in three categories, whereas the Y-distribution in Figure 6 shows that the negative category is not well distributed.

The results in Table 9 could have been better; only Fast-Text achieved a quick performance. This bad result could be caused by the categories' too-strongly

Source: self-made

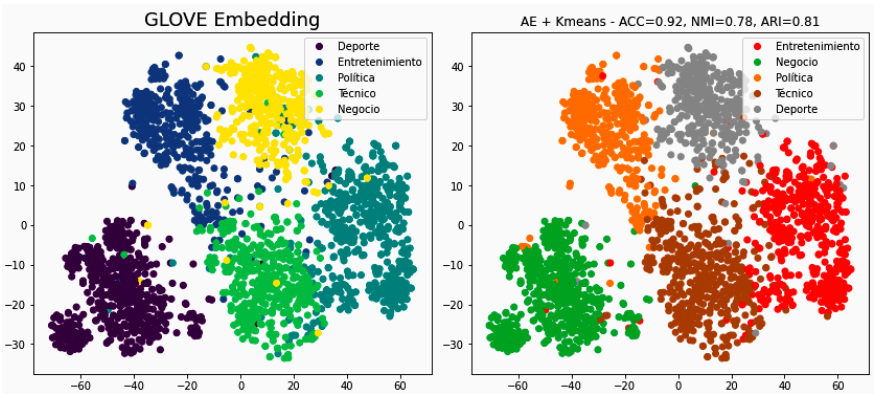


Figure 4. Clustering BBC news.

Source: self-made

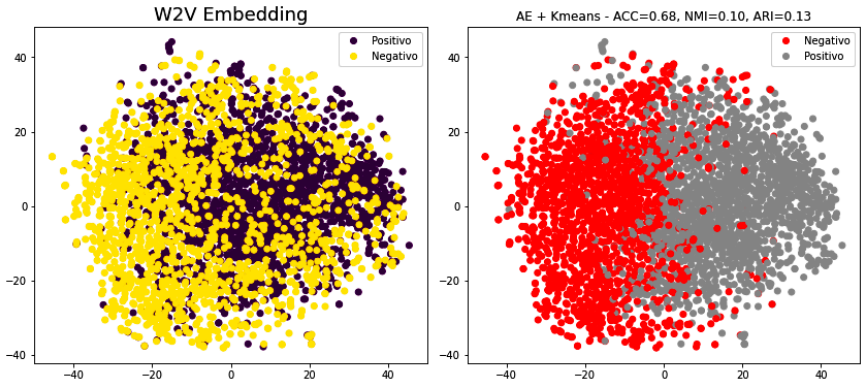


Figure 5. Clustering IBDB.

Table 8. Peruvian Food dataset results.

| 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 |

Source: self-made

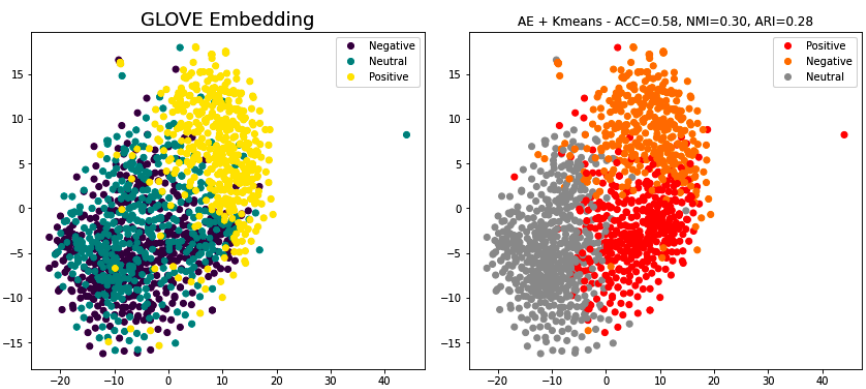


Figure 6. Clustering Peruvian food.

coupled Y-distribution in Figure 7, which shows a lot of noise.

In Table 10, GPT2 Embedding is slightly more successful when applied to the Autoencoder model. However, the NMI and ARI scores are quite low in Accuracy. The dataset was labelled into six categories, and the Y distribution in Figure 8 illustrates that some categories are not well clustered; this is another clear example of how the autoencoder model behaves in noisy data.

The results of this research highlight the effectiveness of using embeddings and unsupervised learning models for clustering Spanish texts. In particular, the AutoEncoder model combined with K-means demonstrated superior performance compared to other methods. This superior performance is consistent

Table 9. Consumer Complaint dataset results.

| 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 |

Table 10. Peruvian Food dataset results.

| 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 |

with previous studies that have noted the capability of AutoEncoders to reduce dimensionality and capture essential features of textual data, facilitating the clustering process [37]. However, our study goes further by utilizing a variety of embeddings, showing that high-quality embeddings like Glove can significantly improve clustering results; this is consistent with the findings of Giordano *et al.* [21], who found that the quality of embeddings is crucial for the success of natural language processing tasks.

Furthermore, comparing our findings with similar research in natural language processing of Spanish texts, a consistent trend towards using transformer-based models such as BERT and GPT-2 is observed. Although these models have shown promising results in other NLP

Source: self-made

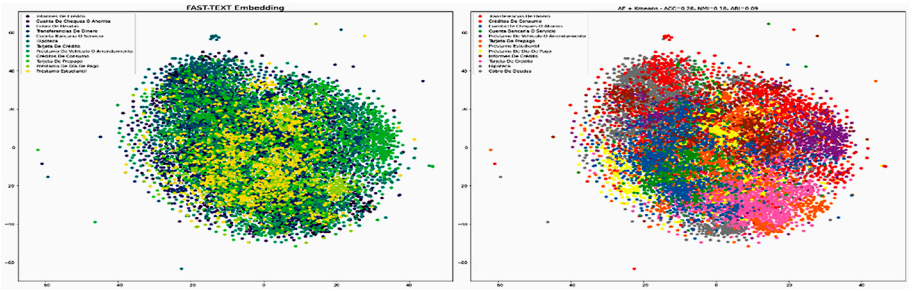


Figure 7. Clustering abstracts consumer complaint.

Source: self-made

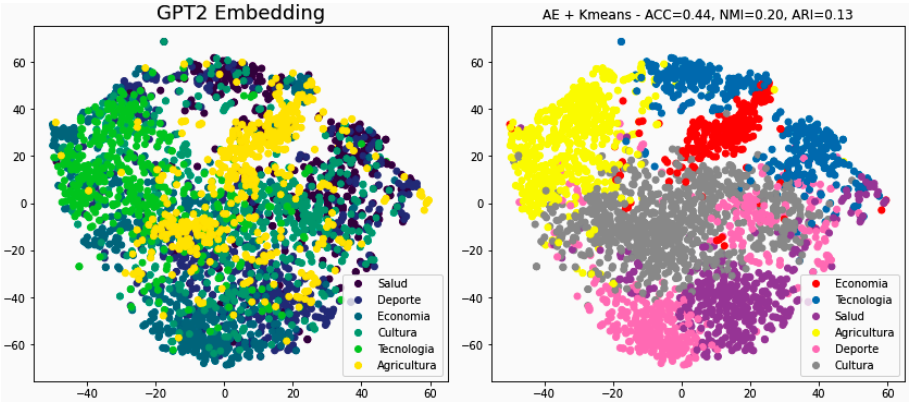


Figure 8. Clustering abstracts scopus 2021.

applications, our study suggests that for specific text clustering tasks, traditional methods like K-means, when combined with dimensionality reduction techniques such as AutoEncoders, can offer significant advantages. For example, Carrera and Obregon [28] found that while GPT-2 is effective for text generation, its performance in clustering tasks does not surpass more traditional methods when specific embeddings are applied. This performance highlights the need to consider the task's nature and the data's quality when choosing the most suitable NLP approach [38].

## CONCLUSIONS

The collection of five Datasets from different sources allowed the analysis of NLP techniques for data cleaning and normalization. The library used to process the text is NLTK in Spanish. Duplicates, Nulls, StopWords and special characters were eliminated. LabelEncoder and StandardScaler were used to normalize the data.

The generation of Embeddings in the Datasets allowed the comparison of five Embeddings: W2V, Glove, FastText, Bert and GPT2, which resulted in the best performance in the news Dataset "BBC News". The glove was the highest-quality embedding applied to the unsupervised AutoEncoder + K-means model with ACC 0.92, NMI 0.79, and ARI 0.81.

As a result, it is not easy to know which embedding behaves better for all the datasets because, as the results show, the correct clustering depends on the type of information processing because the texts may have semantics in common. It can be determined that the models implemented with Genism and applied to neural network models with Autoencoder and K-means layer give better results than Bert and GPT2 based on transformers. The categories' numbers affected the models' results because some categories had semantics in common with others; this could be determined with the Elbow, Silhouette, and Calinski Harabasz methods scores.

## RECOMMENDATIONS

The progress in the study of natural language processing has led to the emergence of new techniques; however, the Spanish language is not as standardized as the English language, which has made significant progress in this area.

On the one hand, to improve the autoencoder model further, it is recommended that well-labelled datasets, such as BBC News, be worked with, which will not generate losses at validation time. On the other hand, to improve the other results on the rest of the datasets presented in this research, it could be possible to apply different weights and fine-tunby adding some more layers.

In contrast with other clustering techniques, semantic-based clustering and implementing a different embedding technique could be useful. It is important to analyze the Elbow, Silhouette and Calinski Harabasz method scores to validate whether

the chosen clusters are correct and thus discard in a labelled Dataset, which categories allow for improvement of the model.

## REFERENCES


[1] M. Danesh and H. Shirgahi, "Text document clustering using semantic neighbors," *Journal of Software Engineering*, vol. 5, no. 4, pp. 136-144, Sep. 2011, doi: 10.3923/jse.2011.136.144.

[2] S. Bozinovski, "Reminder of the first paper on transfer learning in neural networks, 1976," *Informatica*, vol. 44, no. 3, pp. 291-302, Sep. 2020, doi: 10.31449/inf.v44i3.2828.

[3] J.S. Lee and J. Hsiang, "Patent claim generation by fine-tuning OpenAI GPT-2," *World Patent Information*, vol. 62, Art. no. 101983, Sep. 2020, doi: 10.1016/j.wpi.2020.101983.

[4] M. Zhou, D. Liu, Y. Zheng, Q. Zhu, and P. Guo, "A text sentiment classification model using double word embedding methods," *Multimedia Tools and Applications*, vol. 81, no. 14, pp. 18993-19012, Jun. 2022, doi: 10.1007/s11042-020-09846-x.

[5] J. Devlin, M.W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1 (Long and Short Papers)", 2019, pp. 4171-4186, doi: 10.18653/v1/N19-1423.

[6] A.K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651-666, Jun. 2010, doi: 10.1016/j.patrec.2009.09.011.

[7] W.H.E. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *Journal of Classification*, vol. 1, pp. 7-24, Dec. 1984, doi: 10.1007/BF01890115.

[8] M. Ester, H.P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996.

[9] S. Dudoit and J. Fridlyand, "Bagging to improve the accuracy of a clustering procedure," *Bioinformatics,* vol. 19, no. 9, pp. 1090-1099, 2003, doi: 10.1093/bioinformatics/btg038.

[10] A.F. McDaid, D. Greene, and N. Hurley, "Normalized Mutual Information to evaluate overlapping community finding algorithms," 2011, arXiv:1110.2515.

[11] G.W. Milligan and M.C. Cooper, "A study of the comparability of external criteria for hierarchical cluster analysis," *Multivariate Behavioral Research*, vol. 21, no. 4, pp. 441-458, Oct. 1986, doi: 10.1207/s15327906mbr2104_5.

[12] J.B. MacQueen, "On the asymptotic behavior of k-means," *University of California, Western Management Science Institute*, no. 89, pp. 1-5, 1965.

[13] A. Subakti, H. Murfi, and N. Hariadi, "The performance of BERT as data representation of text clustering," *Journal of Big Data*, vol. 9, no. 15, Dec. 2022, doi: 10.1186/s40537-022-00564-9.

[14] X. Li, Z. Chen, L.K.M. Poon, and N.L. Zhang, "Learning latent superstructures in variational autoencoders for deep multidimensional clustering," 2018, arXiv: 1803.05206.

[15] E. Xing, A. Ng, M. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," *Advances in Neural Information Processing Systems*, vol. 15, 2003.

[16] S. Xiang, F. Nie, and C. Zhang, "Learning a Mahalanobis distance metric for data clustering and classification," *Pattern Recognition*, vol. 41, no. 12, pp. 3600-3612, Dec. 2008, doi: 10.1016/j.patcog.2008.05.018.

[17] C. Boutsidis, M.W. Mahoney, and P. Drineas, "An improved approximation algorithm for the column subset selection problem," 2008, arXiv:0812.4293.

[18] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491-502, April 2005, doi: 10.1109/TKDE.2005.66.

[19] S. Alelyani, "On feature selection stability: a data perspective," Ph.D. dissertation, Arizona State University, Arizona, USA, 2013.[Online]. Available: https://keep.lib.asu.edu/items/151587/view

[20] W.N.I. Al-Obaydy, H.A. Hashim, Y.A. Najm, and A.A. Jalal, "Document classification using

term frequency-inverse document frequency and K-means clustering," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 27, no. 3, p. 1517, Sep. 2022, doi: 10.11591/ijeecs.v27.i3.pp. 1517-1524.

[21] V. Giordano, E. Coli, and A. Martini, "An open data repository for engineering design: using text mining with open government data," *Computers in Industry*, vol. 142, Art. no. 103738, Nov. 2022, doi: 10.1016/j.compind.2022.103738.

[22] T. Walkowiak and M. Gniewkowski, "Evaluation of vector embedding models in clustering of text documents," in *Proceedings of Recent Advances in Natural Language Processing*, Varna, Bulgaria, Sep 2-4, 2019, pp. 1304-1311, doi: 10.26615/978-954-452-056-4_149.

[23] Z.H. Kilimci and S. Akyokuş, "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)*, Samsun, Turkey, 2019, pp. 548-553, doi: 10.1109/UBMK.2019.8907027.

[24] F. Role, S. Morbieu, and M. Nadif, "Unsupervised evaluation of text co-clustering algorithms using neural word embeddings," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, New York, USA Oct. 2018, pp. 1827-1830, doi: 10.1145/3269206.3269282.

[25] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017,* I. Guyon *et al.* Eds. Long Beach, CA, USA December 4-9*,* 2017, pp. 5998-6008.

[26] T.B. Swysen Cachaña, "Validación de representaciones vectoriales de palabras", Tesis de Grado, Facultad de Ciencias Físicas y Matemáticas,Departamento de Ciencias de la Computación, Universidad de Chile, Santiago,Chile, 2020. [En línea]. Disponible: https://repositorio.uchile.cl/handle/2250/179647

[27] M. Abdalla, M. Abdalla, G. Hirst, and F. Rudzicz, "Exploring the privacy-preserving properties of word embeddings: algorithmic validation study," *Journal of Medical Internet Research* , vol. 22, no. 7, Jul. 2020, doi: 10.2196/18055, Art. no. p. e18055.

[28] B. Carrera and J. Obregon, "Datificate gpt2 small spanish model," *Metatex.io*, 2021. [Online]. Available: https://metatext.io/models/datificate-gpt2-small-spanish

[29] A. Maas, R.E. Daly, P.T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 142-150. [Online]. Available. https://aclanthology.org/P11-1015/

[30] D. Greene and P. Cunningham, "Practical solutions to the problem of diagonal dominance in kernel document clustering," in *Proceedings of the 23rd international conference on Machine learning (ICML '06)*, New York, USA, 2006, pp. 377-384,

[31] Lázaro, "Peruvian Food Reviews," *Kaggle.com*, 2021. [Online]. Available: https://www.kaggle.com/datasets/lazaro97/peruvian-food-reviews

[32] T. Wei, Y. Lu, H. Chang, Q. Zhou, and X. Bao, "A semantic approach for text clustering using WordNet and lexical chains," *Expert Systems with Applications*, vol. 42, no. 4, pp. 2264-2275, Mar. 2015, doi: 10.1016/j.eswa.2014.10.023.

[33] A. Hotho, S. Staab, and G. Stumme, "Ontologies improve text document clustering," in *Third IEEE International Conference on Data Mining*, FL, USA, 2003, pp. 541-544, doi: 10.1109/ICDM.2003.1250972.

[34] R.J.G.B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Part II*, J. Pei, V.S.Tseng, L. Cao, H. Motoda, and G. Xu, Eds. vol. 7819, Gold Coast, Australia, April 14-17, 2013, pp. 160-172, doi: 10.1007/978-3-642-37456-2.

[35] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579-2605, 2008. [Online]. Available: https://jmlr.org/papers/v9/vandermaaten08a.html

[36] A. Hadifar, L. Sterckx, T. Demeester, and C. Develder, "A self-training approach for short text clustering," in *Proceedings of the 4th*

*Workshop on Representation Learning for NLP (RepL4NLP-2019)*, Florence, Italy, 2019, pp. 194-199, doi: 10.18653/v1/W19-4322.

[37] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised Deep Embedding for Clustering Analysis," 2016, ArXiv:1511.06335.

[38] E. Barón Ramírez, C.W. García Estrella y S.K. Sánchez Gárate, "La inteligencia de negocios y la analítica de datos en los procesos empresariales", *Revista Científica de Sistemas e Informática*, vol. 1, no. 2, pp. 38-53, 2021, doi: 10.51252/rcsi.v1i2.167.