ISSN 1870-4069

Towards Semantic Data Extraction based on Long-Texts Clustering Using Parallel Computing

Diego Martínez-Maqueda[1], Cecilia Reyes-Peña[3], Jesús García-Ramírez[1,2]

[1] Universidad Politécnica Metropolitana de Hidalgo,
Mexico

[2] Unidad Profesional Interdisciplinaria de Ingeniería Campus Tlaxcala,
Mexico

[3] Instituto de Investigaciones en Matemáticas y en Sistemas,
Mexico

231220009@upmh.edu.mx, ceciliareyes@turing.iimas.unam.mx,
jegarciara@ipn.mx

Abstract. The World Wide Web and the development of increasingly complex websites and platforms generate large amounts of written data on a daily basis, which contain potential information that can be used for various purposes. The use of literary texts as a source of information for different Artificial Intelligence (AI) techniques is more limited compared to other types of texts, however, working with them can result in obtaining useful information as well as problems with their processing due to the peculiarities of these texts. This work proposes data pre-processing algorithms applied to short stories and novels written in Spanish for the subsequent extraction of information using pre-trained language models and other Natural Language Processing (NLP) techniques. In addition, the use of parallelization techniques is also included to optimize the execution times of some of the algorithms used.

Keywords: Text clustering, semantic similarity, natural language processing, parallel computing.

1 Introduction

The Big Data era is characterized by the massive access to diverse data types, including text, images, audio, video, and raw data. This abundance of web-derived data necessitates efficient processing techniques, as parallel computing, especially for machine learning applications. Incorporating metadata into the data analysis process can significantly enhance data comprehension, particularly for literary texts. Automatic information extraction techniques offer a promising solution for metadata acquisition.

Specialized literature encompasses a wide range of text sources, including medical notes, news articles, and even web resources like Wikipedia. However, the use of literary or fictional texts for information extraction remains limited.

pp. 17–28; rec. 2024-07-11; acc. 2024-09-03

This is because the context of literary texts can vary significantly, even within the same genre. Labaut and Bost [5] argue that analyzing fictional texts requires special attention, as improper techniques can lead to erroneous results. One approach to mitigate errors is to propose a genre-specific information extraction analysis. However, this necessitates the identification of more representative features, making the process more complex.

This work presents the implementation of a clustering algorithm specifically designed for Spanish literary texts (focusing on short stories and novellas). The algorithm aims to identify the most semantically similar texts based on extracted features. This facilitates the identification of named entities and the extraction of semantic relationships, ultimately leading to a comprehensive character description within each text. Additionally, we prioritize reducing information pre-processing runtime by leveraging parallel processing techniques to maximize computational efficiency.

The proposed clustering method successfully groups texts based on shared characteristics, such as author or country of origin. These well-defined clusters demonstrably enhance the effectiveness of information extraction tasks. Furthermore, the implementation of parallel computing techniques significantly reduces the computational time required for various tasks, including data vectorization and the elbow method.

This paper is structured as follows. Section 2 reviews existing research on named entity extraction and text vectorization. Section 3 details the proposed text clustering methodology, including the pre-processing steps. Section 4 presents the experimental findings. Finally, Section 5 discusses the conclusions and outlines potential future work.

2 Related Work

Named entity extraction is a relevant subtask within Natural Language Processing. NER aims to identify and classify specific words or phrases (entities) within a text according to predefined categories. These categories, often referred to as typological tags, can include people, organizations, locations, dates, monetary values, among others. In the next paragraphs we describe some related work to this task.

Van Dalen-Oskam et al. [3] adapted a system named namespace with conditional random fields (CRFs), support vector machines, and distributional word vectors. Their system achieved acceptable performance in identifying various entities (names, places, and organizations) within modern Dutch literary texts. The F1-measure results were 83.8%, 84.5%, and 89.3% for names, places, and organizations, respectively. Long et al. [7] focused on information extraction from complex materials like Qing and Ming dynasty novels. Their work employed CRFs to address the challenges posed by this genre. They achieved an F1-measure of 80.31%.

Bick [2] leverage a combination of named entity recognition (NER), the PALAVRAS morpho-syntactic and semantic analyzer [1], and an extension of this method to identify various entities in Portuguese and Brazilian literature. Their focus extends beyond named entities, encompassing genres, titles, professions, social statuses, and familial relationships. While achieving good F1-measure results for character identification (63.4%) and genre identification (89.5%). Nonetheless, for the identification of professions (26.6% F1-measure) and familial relationships (15.5% F1-measure) yielded lower performance.

The aforementioned works in this section focus on named entity extraction but do not utilize parallel computing to expedite processing times. However, research by Fu et al. [4] demonstrates the potential of parallel computing in this domain. Their work employs tree-structured conditional random fields (CRFs) for named entity identification. The inherent parallelism of this approach, facilitated by the tree structure implementation, allows for significant reductions in processing times, as demonstrated by the authors.

If we talk about text clustering there are some researchers that work with literary texts. Omar [10] worked with a corpus of 74 novels applying a hybrid method to select the appropriate number of features to obtained good defined clusters. Using bag-of-words, he converted the texts into numeric representations obtaining a 74 x 37534 matrix. Then, with a variance analysis using ANOVA, a term frequency-inverse document Frequency (TF-IDF) analysis and a Principal Components Analysis, he reduced the number of features to 50. After that and with the new matrix, he used K-means method and obtained three good defined clusters.

Another research is by Sobchuk and Šeļa [11], they made combinations of level of thematic foregrounding, features analysis and measure of distance in literary texts clustering. The objective was to determine the best combination for the clusters. The results said that using a strong thematic foregrounding, which include lemmatization words, remove 100 most frequent words, remove entities, nouns, verbs, adjectives and adverbs and simplify the vocabulary replacing the less frequent words with their more frequent synonyms, the doc2vec algorithm to feature analysis and the cosine distance were the best combination.

On the other hand, Wang et al. [12] applied parallelization to the K-means algorithm to clustering texts. They divided all their process in three steps: first, they obtain a numerical representation of the words using the Wrod2Vec algorithm; second, they use the Canopy algorithm to calculate the initial centers and clusters; finally, they use de K-means algorithm to update the centers and obtain the final clusters using the Euclidian distance. In all this process, they applied parallel computing in the second and third step. They observed than increasing the number of processors, the acceleration ratio increased and the expansibility decreased.

3 Text Clustering through Parallel Computing

Extracting named entities effectively necessitates the development of methods that can capture descriptive features from the text and group them based on similarities. This task can be particularly challenging due to the inherent complexities of literary text analysis. The diverse writing styles employed in literary works can further complicate the feature extraction and clustering processes.

To achieve this objective, we establish the following selection criteria for the text corpus used in the clustering process. We will select a dataset of short stories written in Spanish. Each story will be limited to a maximum of 15,000 words and must be categorized as either a short story or a novella. This focus on short narrative forms allows for a more controlled analysis while still enabling the exploration of potential feature variations between these categories. By employing these classifications (short story and novella), we aim to investigate the presence of distinct descriptive features within each class. We hypothesize that novellas, with their inherent narrative complexity compared to short stories, may exhibit a richer set of descriptive features. The documents were obtained from many different websites which offer a big variety of literary works. These texts were obtained manually and they were chosen based on their popularity. In general, the average length for all the texts is 17.84 words, but if we talk about novels the average length is 16.28 words and for the short stories the average length is 20.08 words.

To facilitate analysis and interpretation by a computational model, we propose the use of a numerical text representation that incorporates semantic information. We selected the Doc2Vec representation [6] for this purpose. Doc2Vec is a neural network-based approach that excels at generating numerical vector representations of text documents. This technique captures not only the surface meaning of the words themselves but also the broader context in which they appear within the document. This capability is particularly valuable for our task, as it allows us to account for the nuances of language use frequently encountered in literary texts. The decision of use doc2vec instead another option like sentence2vec is based on work with a only vector representation for each document that contains the particular context of them. In this point, we discarded the use of transformers because when working with long documents, the transformer could lose context information due to the concentration of attention on non-priority elements of the context.

After obtaining the numerical representation of the text data, we employ a dimensionality reduction technique called t-SNE [8] to visualize the high-dimensional vectors in a two-dimensional space. This visualization serves two purposes: 1) to assess the overall distribution of the data and 2) to guide the selection of an appropriate clustering method. The t-SNE shows a inverse symmetric data dispersion with a central axis parallel to dimension 2. Consequently, we propose using a distance-based clustering algorithm to identify the closest documents within each cluster based on their numerical vector representations.

Certain aspects of the proposed method are computationally expensive. To address this challenge, we leverage parallel processing techniques to accelerate the following tasks: text embedding transformation, data clustering, and hyperparameter optimization for the clustering algorithm. By employing parallel computing, we anticipate a significant reduction in overall computation time compared to a sequential implementation. To achieve that we propose the use of programming language Python with the library multiprocessing which take advantages of devices with multiple processors. There are another languages that can work with parallel computing, for example C or C++ but, many projects related with machine learning and data science use Python as programming language. Moreno Arboleda et al. [9] made a comparative between Python and C++ in parallel algorithms and they demonstrated that for parallel process C++ has a better performance. For a small dataset, we expect that difference is no big.

Finally, we will evaluate the quality of the resulting clusters. For this purpose, we will employ various evaluation metrics, including the Silhouette coefficient. The Silhouette coefficient is a valuable metric that assesses the separation between clusters. It considers both the average distance between points within a cluster and the average distance to points in the closest neighboring cluster. A high Silhouette coefficient indicates well-separated clusters where points are closely grouped within their assigned cluster and far from points in other clusters.

4 Experimental Results

This section presents the experimental findings obtained from the proposed clustering method for short stories and novellas. The experiments address two primary objectives: (i) we aim to identify a clustering method that demonstrates effectiveness within this specific domain of literary text; and (ii) we evaluate the performance of our parallel implementation by testing it with datasets of varying sizes and utilizing different computational resources.

The experiments were conducted on a machine equipped with an Intel Core i5-9400F processor (6 cores) and 32 GB of memory. The operating system is Ubuntu 22.04. This configuration provided sufficient computational resources to effectively leverage parallel processing techniques.

Our experiments leverage a collection of 52 Spanish-language documents, comprised of 32 short stories and 20 short novellas. Associated metadata for each document includes title, year, author, and country of origin. Each document was converted into a numerical vector representation. Figure 1 depicts a t-SNE projection of this data in a two-dimensional space, which serves to visualize the distribution of the document embeddings.

Following the t-SNE dimensionality reduction, well-defined clusters may not be readily apparent in the data visualization (Figure 2). To address this challenge and determine the optimal number of clusters for the k-means algorithm, we employ the elbow method. This method involves running k-means with varying numbers of clusters and analyzing the resulting sum of squared errors (SSE) for each k value. The elbow method seeks the value of k where the SSE starts to plateau or decrease less significantly, indicating the point at which adding additional clusters yields diminishing returns in terms of intra-cluster variance reduction. The goal is to find a balance between minimizing intra-cluster variance

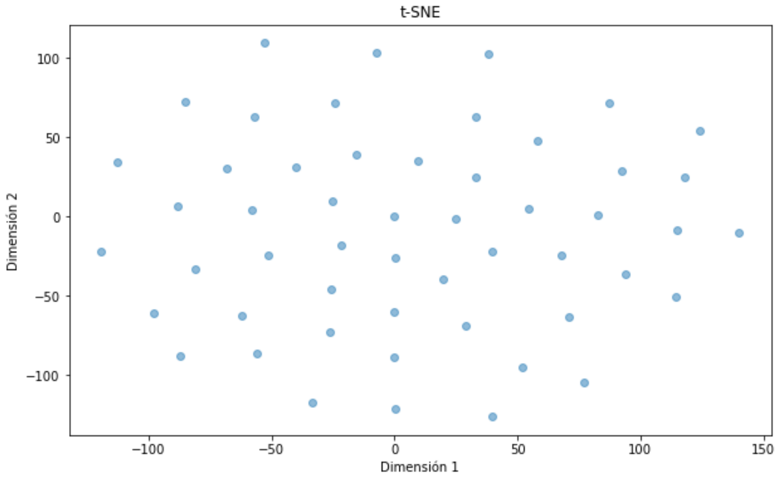


Fig. 1. Projection of the numerical data into a two-dimensional space using t-SNE algorithm.

(where points within a cluster are similar) and maximizing inter-cluster variance (where points between clusters are dissimilar). According to the results we consider that 3 is the best number of clusters.

The text transformation hyperparameters are described next:

- Vector Size: The dimensionality of the word vectors was set to 10, this value obtain good performance in our experiments.
- Minimum Count: This hyperparameter in Doc2Vec determines the minimum frequency a word needs to appear in the corpus to be included in the vocabulary. We set this value to 2.
- Epochs: The number of training epochs for Doc2Vec was set to 40.

The clustering results are visualized in the scatter plot of Figure 3, which utilizes t-SNE dimensionality reduction similar to Figure 1. The resume of meta data documents for each clusters is shown in Table 1, where there are the values of data and their frequencies within the clusters. The Silhouette coefficient score for this clustering is 0.16676, indicating a relatively low degree of separation between the clusters. This may be partially attributed to the presence of sparse documents within the dataset, which can introduce noise and hinder the clustering process.

In Table 1, the clusters obtained by the proposed method are shown. We can see that the embeddings capture information about the authors who wrote the novels or short stories. Similarly, the embeddings capture information about the years in which the texts were written. However, the embeddings do not perform well for the country or type of text (story or novella).

For the parallel implementation, we leverage the multi-process library in Python to facilitate multi-threading. This approach enables us to distribute tasks

Table 1. Summary of the obtained from the clustering pre-process, the numbers between the parentheses represent the frequency of the data obtained in each cluster.

	Cluster 0	**Cluster 1**	**Cluster 2**
Author	Herman Melville (1) Julio Cortazar (1) Gabriel García Marquez (2) Juan Rulfo (5) Amado Nervo (2) Albert Camus (1) Pedro Antonio de Alarcón (4) Jorge Luis Borges (3) Julia de Asensi (3) Frank Kafka (2) José Emilio Pacheco (1)	Horacio Quiroga (2) Edgar Allan Poe (2) Julio Cortazar (1) Juan Rulfo (1) Pedro Antonio de Alarcón (2) Julia de Asensi (1)	Ambrose Bierce (1) Juan Rulfo (4) Amado Nervo(2) Frank Kafka (1) Guy de Maupassant (1) Edgar Allan Poe (2) Gabriel García Márquez (1) Alberto Moravia (1) Fernando Díaz-Plaja (1) Pedro Antonio de Alarcón (1) José Emilio Pacheco (3)
Country	US (1) AR (4) CO (2) MX (8) FR (1) ES (7) GE (2)	UR (2) US (2) AR (1) MX (1) ES (3)	US (3) MX (9) GE (1) FR (1) CO (1) IT (1) ES (2)
Year	1853 (1) 1946 (1) 1961 (1) 1953 (5) 1968 (1) 1916 (1) 1917 (1) 1942 (2) 1877 (1) 1970 (1) 1941 (1) 1882 (1) 1972 (1) No date (7)	1917 (2) 1835 (1) 1964 (1) 1953 (1) 1843 (1) 1854 (1) No date (2)	2010 (1) 1953 (4) 1918 (1) 1915 (1) 1870 (1) 1832 (1) 1981 (1) 1957 (1)
Type	Story (14) Novella (11)	Story (6) Novella (3)	Story (13) Novella (5)

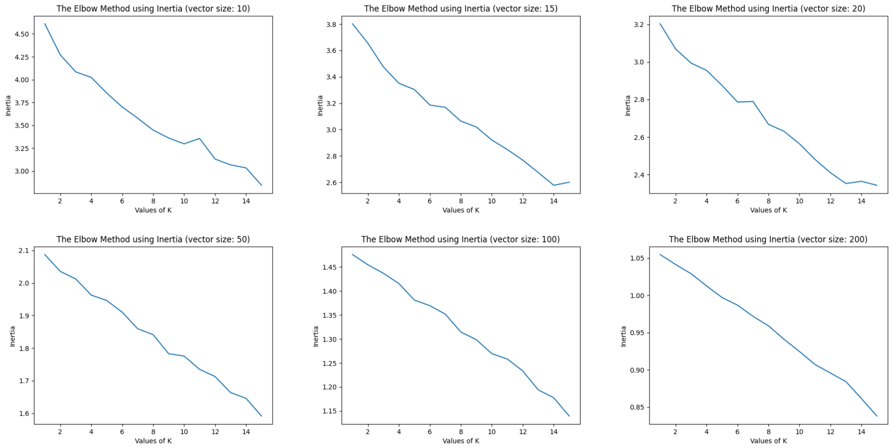


Fig. 2. Elbow method results using different vector size. According to the experimental results, the best value for the number of cluster is three.

across multiple cores, accelerating the computational processes. Additionally, we utilize NumPy for vectorization, which significantly optimizes operations involving numerical vectors.

In the first configuration, the text-to-numerical vector conversion and distance calculation tasks were parallelized, while the elbow method was executed sequentially. A reduction in execution time was observed primarily for the text vectorization step. This is likely because vectorization using libraries like NumPy is well-suited for parallelization. However, the overall speedup was limited because the elbow method remained sequential and potentially dominated the total execution time. In the Figure 4a depicts the execution time variations for the last configuration using different numbers of threads.

Building upon the observation that vectorization aided distance calculations in the first configuration, we designed a second configuration to explore potential speedups within the elbow method itself. This configuration leverages vectorization throughout the elbow method, aiming to reduce its computational cost. Our experiments demonstrated that this configuration achieved better overall performance compared to the first, as illustrated in Figure 4b.

Considering the execution time variations depicted in Figures 4, the second configuration demonstrably achieved superior performance compared to the first one. This highlights the potential benefits of incorporating vectorization within the elbow method itself. If the proposed method is to be implemented for larger datasets or more computationally intensive tasks, leveraging parallel processing techniques like those employed in the second configuration would be particularly advantageous to ensure efficient execution by maximizing the utilization of available computational resources.

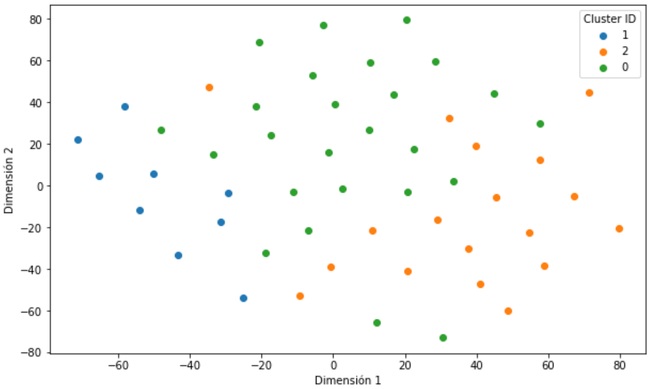
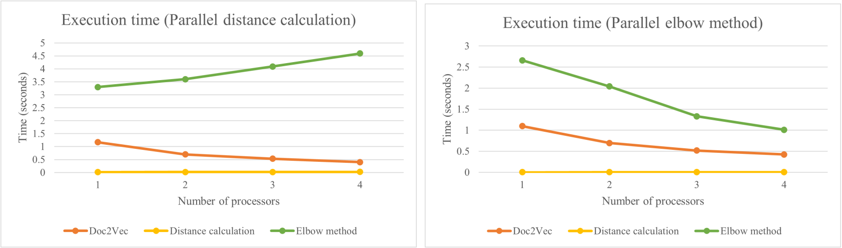


Fig. 3. Projection of the clusters obtained by the k-means algorithm.



(a) First Configuration

(b) Second Configuration

Fig. 4. Execution time for the first (a) and second (b) configuration.

In addition to the execution time analysis, we can evaluate the effectiveness of parallel computing using two key metrics: speedup and efficiency. These metrics are calculated based on the following equations (Equations 1 and 2, respectively):

$$S_p = \frac{sequential\ time}{parallel\ time}, \tag{1}$$

$$Efficiency = \frac{S_p}{Number\ of\ cores}. \tag{2}$$

Tables 2 and 3 present the calculated speedup and efficiency values for the text vectorization and elbow method, respectively. The results indicate that for text transformation, the highest efficiency is achieved with two cores. In contrast, the elbow method demonstrates the best efficiency when using three cores.

These findings suggest that the optimal number of threads for parallel execution can vary depending on the specific task characteristics. The text vectorization process appears to benefit from a lower number of threads. Conversely, the elbow method might exhibit improved scalability with a slightly

Table 2. Speed up and efficiency for the experimental results for the numerical vector transformation.

Number of cores	Speedup	Efficiency
2	1.5797	0.7898
3	2.1161	0.7053
4	2.5917	0.6479

Table 3. Speed up and efficiency for the experimental results for the elbow method.

Number of cores	Speedup	Efficiency
2	1.3035	0.6517
3	1.9980	0.6660
4	2.6298	0.6574

higher number of threads, possibly because it involves more independent computations suitable for parallelization.

5 Conclusions

This work presented a pre-processing method for Spanish literary texts. The proposed method utilizes Doc2Vec to generate numerical representations of the texts, facilitating the identification of potential groupings through clustering techniques. We explored the effectiveness of parallel computing to improve the efficiency of key steps within the pre-processing pipeline, including text vectorization and distance calculations for clustering. The results demonstrate that parallelization can significantly reduce execution time, particularly for computationally intensive tasks like text vectorization.

The experimental results reveal an interesting interplay between vectorization and parallelization. For text vectorization, as observed in the figures, increasing the number of cores used generally leads to a significant improvement in execution time. This aligns with the strengths of vectorization libraries like NumPy, which are well-optimized for parallel processing.

In contrast, parallelization for distance calculation between points did not yield a substantial performance boost compared to vectorization alone. This suggests that for the current dataset size, the computational efficiency gained through vectorization the benefits of additional parallelization using multiple process. However, it is important to note that this behavior might change with larger corpora. As the number of texts in the corpus increases, the potential benefits of parallelization for distance calculations are likely to become more pronounced.

Future work will focus on extracting information from the pre-processed texts, such as character identification and feature extraction. Additionally, we plan to expand our experiments by incorporating a larger corpus of literary texts. This will allow us to gather more comprehensive data and validate the effectiveness of the proposed parallel processing configurations for a broader domain with increased data volume. By analyzing larger datasets, we can gain a more robust understanding of how the method scales and identify potential optimizations for even more efficient text processing.

Acknowledgments. The authors thankfully acknowledge the computer resources, technical expertise and support provided by the Laboratorio Nacional de Supercómputo del Sureste de México, CONAHCYT member of the network of national laboratories, with the project 202301003N.

References

1. Bick, E.: PALAVRAS - A Constraint Grammar-Based Parsing System for Portuguese, pp. 279–302. Bloomsbury Academic (2014)
2. Bick, E.: Extraction of Literary Character Information in Portuguese: Extração de Informação sobre Personagens Literários em Português. Linguamática 15, 31–40 (2023)
3. van Dalen-Oskam, K., de Does, J., Marx, M., Sijaranamual, I., Depuydt, K., Verheij, B., Geirnaert, V.: Named Entity Recognition and Resolution for Literary Studies. Computational Linguistics in the Netherlands Journal 4, 121–136 (Dec 2014)
4. Fu, Y., Tan, C., Chen, M., Huang, S., Huang, F.: Nested Named Entity Recognition with Partially-Observed TreeCRFs. Proceedings of the AAAI Conference on Artificial Intelligence 35(14), 12839–12847 (May 2021), https://ojs.aaai.org/index.php/AAAI/article/view/17519
5. Labatut, V., Bost, X.: Extraction and Analysis of Fictional Character Networks: A Survey. ACM Computing Surveys 52(5), 1–40 (Sep 2020), https://dl.acm.org/doi/10.1145/3344548
6. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 1188–1196. PMLR, Bejing, China (22–24 Jun 2014), https://proceedings.mlr.press/v32/le14.html
7. Long, Y., Xiong, D., Lu, Q., Li, M., Huang, C.R.: Named Entity Recognition for Chinese Novels in the Ming-Qing Dynasties. In: Dong, M., Lin, J., Tang, X. (eds.) Chinese Lexical Semantics, vol. 10085, pp. 362–375. Springer International Publishing, Cham (2016), series Title: Lecture Notes in Computer Science
8. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research 9(11) (2008)
9. Moreno Arboleda, F.J., Rincón Arias, M., Hernández Riveros, J.A.: Performance of Parallelism in Python and C++. IAENG International Journal of Computer Science, 50 (2023)
10. Omar, A.: Feature Selection in Text Clustering Applications of Literary Texts: A Hybrid of Term Weighting Methods. International Journal of Advanced Computer Science and Applications 11(2) (2020)
11. Sobchuk, O., Šeļa, A.: Computational thematics: comparing algorithms for clustering the genres of literary fiction. Humanities and Social Sciences Communications 11(1), 438 (Mar 2024), https://www.nature.com/articles/s41599-024-02933-6

12. Wang, H., Zhou, C., Li, L.: Design and Application of a Text Clustering Algorithm Based on Parallelized K-Means Clustering. Revue d'Intelligence Artificielle 33(6), 453–460 (Dec 2019), http://www.iieta.org/journals/ria/paper/10.18280/ria.330608