big data and cognitive computing

MDPI

Article

A Machine Learning-Based Pipeline for the Extraction of Insights from Customer Reviews

Róbert Lakatos [1,2,*], Gergő Bogacsovics [1,2,†], Balázs Harangi [1,†], István Lakatos [1,2,†], Attila Tiba [1,†], János Tóth [1,†], Marianna Szabó [2,3,†] and András Hajdu [1,†]

[1] Department of Data Science and Visualization, Faculty of Informatics, University of Debrecen, H-4032 Debrecen, Hungary
[2] Doctoral School of Informatics, University of Debrecen, H-4032 Debrecen, Hungary
[3] Department of Applied Mathematics and Probability Theory, Faculty of Informatics, University of Debrecen, H-4032 Debrecen, Hungary
* Correspondence: lakatos.robert@inf.unideb.hu
† These authors contributed equally to this work.

Abstract: The efficiency of natural language processing has improved dramatically with the advent of machine learning models, particularly neural network-based solutions. However, some tasks are still challenging, especially when considering specific domains. This paper presents a model that can extract insights from customer reviews using machine learning methods integrated into a pipeline. For topic modeling, our composite model uses transformer-based neural networks designed for natural language processing, vector-embedding-based keyword extraction, and clustering. The elements of our model have been integrated and tailored to better meet the requirements of efficient information extraction and topic modeling of the extracted information for opinion mining. Our approach was validated and compared with other state-of-the-art methods using publicly available benchmark datasets. The results show that our system performs better than existing topic modeling and keyword extraction methods in this task.

Keywords: machine and deep learning; topic modeling; keyphrase extracting; natural language processing

Citation: Lakatos, R.; Bogacsovics, G.; Harangi, B.; Lakatos, I.; Tiba, A.; Tóth, J.; Szabó, M.; Hajdu, A. A Machine Learning-Based Pipeline for the Extraction of Insights from Customer Reviews. *Big Data Cogn. Comput.* **2024**, *8*, 20. https://doi.org/10.3390/bdcc8030020

Academic Editors: Tim Schlippe and Matthias Wölfel

Received: 22 January 2024
Revised: 15 February 2024
Accepted: 19 February 2024
Published: 22 February 2024

1. Introduction

Users of social platforms, forums, and online stores generate a significant amount of textual data. One of the most valuable applications of machine learning-based text processing is to extract words and phrases that describe the content of these texts. In e-commerce, the knowledge contained in data such as customer reviews can be of great value and can provide a tangible and measurable financial return.

The difficulty in solving this problem effectively with automated methods is that human-generated texts often contain much noise in addition to substantive details. Filtering the relevant information is further complicated by the fact that different texts can have different characteristics. For example, the document to be analyzed may contain frequently occurring words that can be considered noise or irrelevant information for the analysis. What is considered noise may differ depending on how the data are viewed and what is considered relevant. This makes it difficult to solve this task: it is not enough to find some specific information in texts; we also have to decide what information we need.

Our goal was to extract information from textual data in the field of e-commerce. Our application is an end-to-end system that uses machine learning tools developed for natural language processing and can identify those sets of words and phrases in customer reviews that characterize their opinions.

To build an application that can be used in an e-commerce environment, we needed a model to identify topics in texts and to provide a way to determine which topics are relevant

to our analysis goals. Therefore, we investigated the N-gram model [1], dependency-parsing [2] and embedded-vector-space-based keyword extraction solutions, and various distance- or density-based and hierarchical clustering [3,4] techniques. In addition, we tested the LDA [5,6], Top2Vec [7], and BERTopic [8] methods for modeling complex topics. It was also important to us that these methods have a stable implementation and be verifiably executable.

Extracting information from customer feedback and reviews will achieve the desired result if we can identify the words and phrases that describe the customers' opinions and thus help further improve a product or service from both a sales and a technical point of view. In other words, it can help increase revenue or avoid potential loss. However, it is not enough to know the frequency of certain words and phrases; it is also necessary to provide the possibility of grouping the phrases according to different criteria. For example, different sentiment features can affect the information value of a frequent phrase. Furthermore, highly negative or positive reviews should be excluded from the analysis due to their bias.

To produce good quality results, it is important to identify the text parts that we consider noise. These include stop words [9], special characters, and punctuation. Removing these words is often one of the first steps in text pre-processing. However, noise is often more difficult to identify. In the case of stop words, for example, a complicated problem is the removal of elements that express negation. While this may seem trivial, it may result in a loss of information, depending on the model's behavior. In addition, removing certain punctuation marks can distort the semantics and lead to information loss. Therefore, it is not possible to address these high-level issues with general word-level methods.

We can encounter further difficulties when we analyze these issues at the sentence level. Namely, customer reviews can consist of several separate parts describing different problems. These topics are easier to capture in more complex situations at the sentence or sub-sentence level. Therefore, solving the problems above requires a specialized approach.

Suppose we identify separate text parts, phrases, sentences, or sub-sentences with different meanings. In this case, they can be grouped into useful and useless texts according to their meaning. This is another level of abstraction with its own difficulties in information extraction. One of the critical problems with organizing text into topics is that the number of topics is unknown in advance, and this number changes as the amount of text increases. Finding these semantically distinct sets of text belongs to the topic modeling and text clustering subfields of natural language processing. This is an unsupervised machine learning problem, which is particularly difficult because it requires finding the optimal clusters. Optimal results are obtained when we can extract sufficiently dissimilar clusters in useful customer reviews about a given product. The corpus should be clustered according to the product's substantive information, not word frequency or other properties common to textual data.

In our examination, among the topic modelers, distance- or density-based and hierarchical clustering methods, and keyword extraction solutions we have investigated, LDA, Top2Vec, and BERTopic could meet our requirements. However, none of them offered a comprehensive solution to all problems. The clustering and keyword extraction solutions cannot be considered complex enough. The topic modeling tools did not provide us with adequate results without requiring significant modifications in their implementation to adjust their functionality. Our solution draws on the experience gained with the models and tools listed above. We have taken the building blocks of these approaches and reworked them so we can better address the problems described. The main contributions of this work can be summarized as follows:

- We designed an easy-to-extend pipeline for extracting keywords/keyphrases from text that relies on a semantic embedding approach.
- The pipeline applies recursive hierarchical clustering to find relevant topics. This enables our system to better adapt to the content and structure of the text.
- The pipeline can adjust the density of the resulting sets and remove outliers through iterative hierarchical clustering.

- The performance of our model was compared with that of the topic modeling methods LDA, Top2Vec, and BERTopic.
- We found that using the keywords and keyphrases extracted using our method led to better classification results than those extracted using the other methods. Thus, the extracted keywords effectively preserve the core information content of the original text.

In summary, we were able to build a solution that was better suited to our specific problem than the available solutions, and according to our experiments, it resulted in more sophisticated and usable keyword/keyphrase groups.

The rest of this paper is organized as follows. In Section 2, we overview the recent related work. We present our dataset in Section 3. In Section 4, we present how we prepared the data used for the measurements and what kind of development environment we worked in. Our methodology, implemented keyphrase extraction techniques, and pipeline details, are presented in Section 5. Then, in Section 6, we provide the performance of our pipeline-based model compared to state-of-the-art solutions. Finally, some conclusions are drawn in Section 7.

2. Related Work

The best techniques for extracting relevant information are based on extracting keywords and grouping related information. The models of these solutions are keyword extraction or topic modeling solutions based on word frequency measurement, dependency analysis, text embedding, or clustering algorithms. The most popular of these are the following.

2.1. Keyphrase Extraction

Approaches based on word frequency [10] can be an effective solution for comprehensively analyzing large texts. However, this approach is less efficient for shorter texts, unless we have some general information about the frequency of words in that language. Furthermore, such approaches can be sensitive to the quality of the text cleaning, depending on the nature of the text.

Dependency parsing [2] is another approach that can be used to extract keyphrases from text. This technique attempts to identify specific elements of a text based on the grammatical rules of the language. It can be particularly useful if we have prior knowledge about the type of words that carry the information we are looking for. Dependency-parsing-based solutions tend to work better for smaller texts and sentence fragments compared to frequency-based approaches. When dealing with large amounts of text, it is often helpful to break it down into smaller parts, e.g., sentences. This approach can improve the accuracy of information extraction. One potential drawback of using dependency parsing is that it can be sensitive to the pre-processing of the text.

Semantic approaches based on text embedding [11–13] can also be used to identify keywords. Such an approach involves identifying the relationship between words and parts of the text. This can be achieved by vectorizing the elements of the text and their larger units, such as sentences, and measuring the similarity between them using some metric. The advantage of methods based on this approach is that they are less sensitive to the lack of text cleaning. Their disadvantage is that the quality of the vector space required for similarity measurement largely determines the model's functionality. Furthermore, unlike the previous two approaches, it currently imposes a higher computational burden and works better on smaller texts than on larger texts. However, due to the semantic approach, if the choice of text-splitting rules, similarity metrics, and vector space are chosen well, better results can be obtained than with approaches based on frequency or dependency parsing.

In the case of frequency-based techniques, dependency parsing, or semantic embedding, it can generally be said that, although they offer the possibility of finding the essential elements of a text, none of them provides a clear answer to the question of the relationship between the words found and the topics of the text. If we need to find the main terms of

the text but also group them according to their content to answer higher-level questions, we need to use clustering or topic modeling.

2.2. Clustering

The effectiveness of text clustering is determined by ways to transform the text into an embedded vector space that best represents the documents regarding the target task.

There are several ways to vectorize a text. There are frequency-based techniques, such as one-hot encoding or count vectorization [14]. However, there are also solutions using more complex statistical methods, such as TF-IDF techniques or the LDA [15]. In addition, we can use semantic embedding like GloVe [11] as a first pioneer, which generates vectors for each word on a statistical basis. However, statistical models were soon replaced by neural networks-based solutions with the rise of word2vec [12] and fastText [13]. With the advent of transformers [16], neural networks with special architectures have emerged that can create semantically well-measured embedded vector spaces.

Several clustering techniques are available to group entities in the embedded vector space, for instance, k-means [17], agglomerative [4], DBSCAN [18], and HDBSCAN [19] clustering.

The source of the problem is that we find that embedded textual entities do not usually form enough dense regions in the vector space, even for terms with the same meaning. For this reason, centroid, hierarchical, or density-based methods also have difficulties in handling vector spaces created by text and word embedding.

Another problem of working with text-based embedded vector spaces is that, to achieve good semantic quality, textual data are currently converted into high-dimensional vectors, which are resource-intensive to develop. Although the resource demand can be reduced using various dimension reduction techniques such as PCA [20–22], UMAP [23,24], or T-SNE [25], it results in a loss of information, which in turn can lead to distortions due to the particular structure of the embedded vector space generated from the text.

2.3. LDA

Latent Dirichlet Allocation (LDA) is a popular model for extracting text topics. It has the advantage of having efficient and reliable implementations, making it easy to integrate into machine learning systems. To adapt it to a specific context, Batch [26] or Online Variational Bayes [27] methods have become popular. To measure the quality of the topics generated by LDA and thus find the right number of topics, the perplexity [15] values of each topic are computed. For dictionary construction, the bag-of-words [28] technique is a common choice, with lemmatization and stop-word removal.

For our system, a drawback of this model is that it is a probabilistic statistical method that ignores word order and semantics and is less sensitive for finding smaller topics. This means that it finds fewer topics and cannot isolate finer details. In practice, when using LDA, we tend to focus on, e.g., the top n words, which narrows down the list of key terms, especially when there are few topics. Although these properties of the model are parameterizable, the more words we select from a given topic, the worse the quality of the words or phrases associated with that topic. This leads to greater overlap between the words of the topics, which requires further corrections.

2.4. Top2Vec

Unlike LDA, Top2Vec uses semantic embedding and does not require stop-word removal or lemmatization. Such pre-processing steps can even be detrimental to the model's performance, as they can distort the semantic meaning of the text in the documents.

The quality of semantic embedding-based methods is determined by the embedded vector space they use. This makes Top2Vec sensitive to the vector space of the model it uses on the target corpus. It has the advantage that the model offers a compact topic–number–determination method and can therefore automatically search for topics that it considers to be related. Like LDA, Top2Vec is less suitable for finding smaller topics.

2.5. BERTopic

BERTopic is another topic modeling technique that uses semantic embedding. It forms dense clusters using c-TF-IDF [8] for easier clustering. The model is based on the BERT transformer-based neural network [29] and considers the internal representation of BERT to generate the embedded vector space. Like Top2Vec, it does not require stop-word removal or lemmatization. As with techniques based on semantic embedding, the model's effectiveness is highly dependent on the quality of the embedded vector space it uses.

3. Dataset

We used data from the Amazon Reviews (2018) [30] and the 20 Newsgroups [31] dataset to evaluate our system.

3.1. Amazon Reviews (2018) Dataset

The Amazon Reviews (2018) dataset contains 233.1 million Amazon product reviews (review text, ratings, and auxiliary votes), along with corresponding product metadata and links. For our purposes, we selected the *electronic* subset of this dataset, which contains 7,824,482 reviews, and created a dataset (hereafter Electronics_50K) as follows:

1. We tokenized the review text using the vocabulary and tokenization method of the BERT neural network architecture. The vocabulary used for BERT tokenization is built using the WordPiece [32] subword tokenization algorithm.
2. We kept only those reviews with lengths between 16 and 128 tokens. We fixed the input size to 128 for efficiency reasons.
3. We performed a uniform sampling to obtain 10,000 reviews for each rating (1 to 5, without fractions). This resulted in a dataset of 50,000 reviews.

For a quick impression of what the dataset looks like, see Figures 1 and 2 for its 2D visualizations obtained by PCA and TSNE, respectively. In the visualization, the relative location of the data points representing the individual text highlights the difficulty of clustering the vectors produced from the texts.

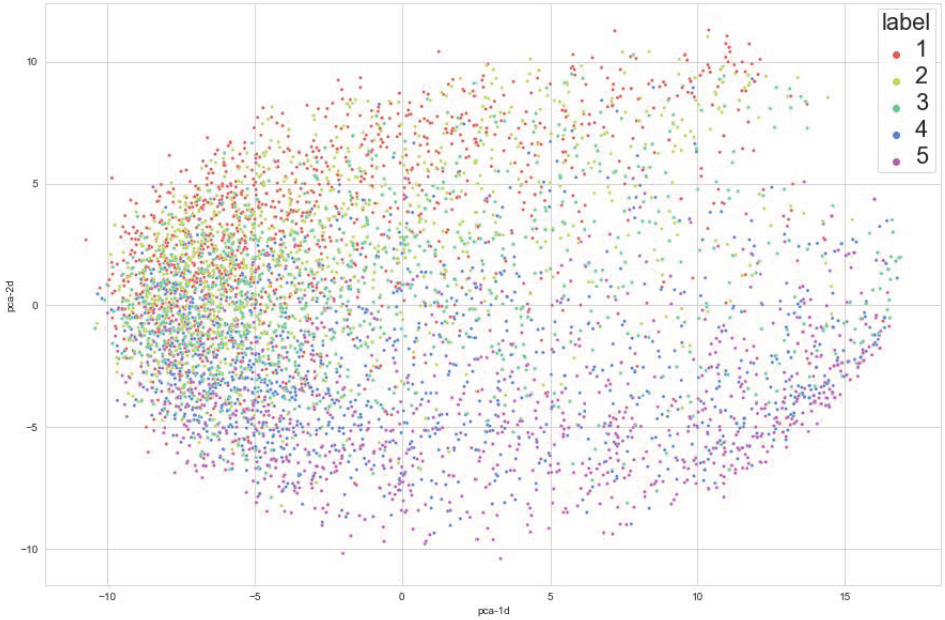

Figure 1. Embedded vector space from the Amazon Reviews dataset visualized using PCA.

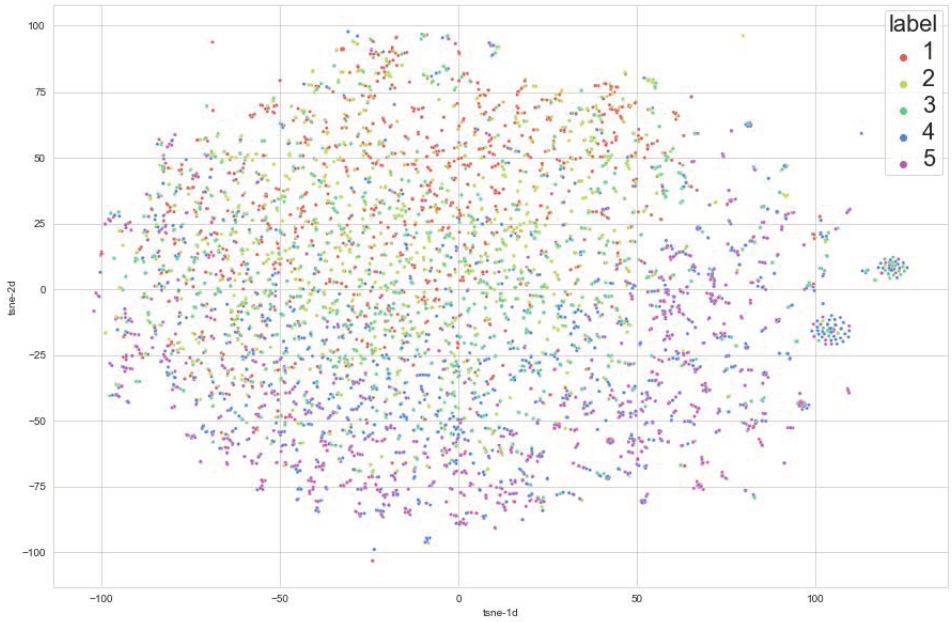

Figure 2. Embedded vector space of the Amazon Reviews dataset visualized using TSNE.

3.2. 20 Newsgroups Dataset

The 20 Newsgroups dataset is a widely used benchmarking dataset that provides training and testing subsets. The dataset contains twenty classes in six categories. Furthermore, different software packages contain this dataset, allowing quick and easy use without pre-processing steps. For an impression of this dataset, see Figures 3 and 4 for its 2D visualizations obtained using PCA and TSNE, respectively.

As with the Amazon Reviews dataset, the locations of the vectors produced from 20newsgroup also highlight the difficulty of clustering.

Figure 3. Embedded vector space of the 20 Newsgroups dataset visualized using PCA.

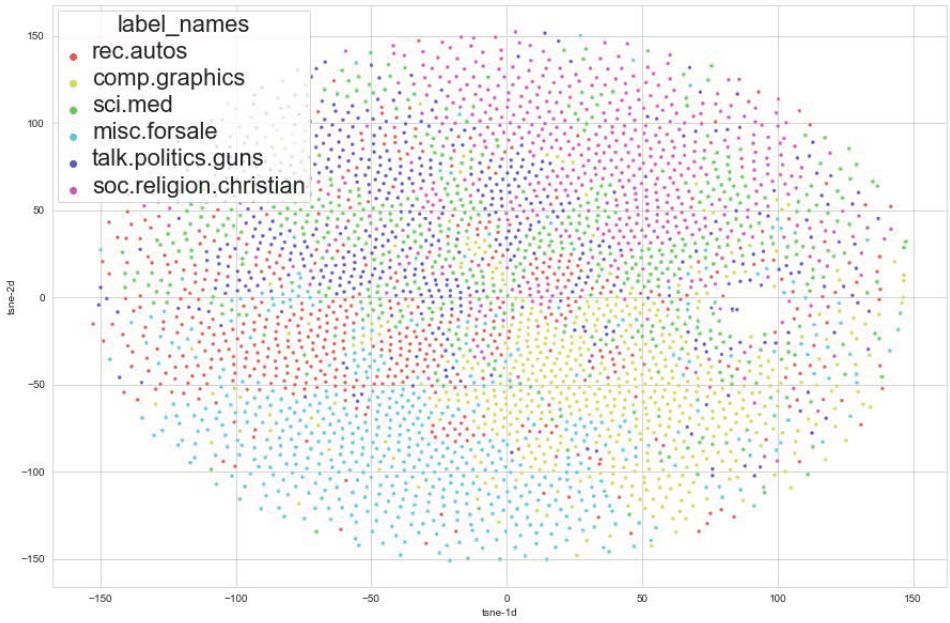

Figure 4. Embedded vector space of the 20 Newsgroups dataset visualized using TSNE.

4. Experimental Setup

We have created multiple different classification tasks from the Amazon Reviews dataset. Due to its relatively large size compared to the available resources, the Cell Phones and Accessories (CPA) subset was used, similar to the evaluation of the keyword extraction models. From the dataset, we selected products with a uniform set of 100 reviews for each class (ratings from 1 to 5). Thus, we created a classification task with 6000 training and 1500 testing records, which we used for a five-label multi-classification. In addition, from the CPA dataset, we took a random sample of 100 sentences, the details of which are described in Section 5.1.4.

Furthermore, using the 20 Newsgroups dataset, we defined binary classification tasks as follows:

1. We chose a single class from each category to obtain sufficiently different elements for binary classification.
2. From the six classes selected, we created all possible binary combinations as classification tasks, i.e., differentiating between each pair of classes.

During the development of the pipeline, we worked in a cloud environment. For data storage, text data were stored in JSON and CSV files in a Hadoop-based storage system [33]. We also used two different clusters for computations for cost efficiency. We worked with a CPU-optimized virtual environment for operations requiring more CPU computations. This environment was configured with a 16-core AMD EPYC 7452 processor (Advanced Micro Devices, Inc., Santa Clara, CA, USA), 128 GB RAM, and 400 GB of storage. For operations with transformer-based neural networks, we used a GPU-optimized virtual environment that was configured with a 12-core Intel Xeon E5-2690 processor (Intel Corporation, Santa Clara, CA, USA), 224 GB RAM, 1474 GB (SSD) cache, and two NVIDIA Tesla P100 32 GB GPUs (NVIDIA Corporation, Santa Clara, CA, USA).

5. Methodology

In this section, we present our evaluation strategy and describe how we implemented the most popular keyword extraction techniques and measured their performance to choose the method we incorporated into our system. Finally, we present our solution, which we included in a unified pipeline.

5.1. Examination of Keyphrase Extraction Techniques

We implemented solutions based on N-gram, dependency parsing, and embedded vectors to determine which keyword extraction technique to implement in our system. We then assessed the capabilities of each technique. In Sections 5.1.1–5.1.3, we describe the details of each implementation. In Section 5.1.4, we describe the method and results of the evaluation of each technique, which determined the type of solution we chose.

5.1.1. N-Gram-Based Approach

N-grams refer to sequences of N-adjacent words within a given text or speech, where N is the number of words in the sequence (e.g., 2-grams or bigrams, 3-grams or trigrams). N-grams are widely used in NLP tasks because they help capture local context and relationships between words. N-gram-based approaches aim to extract keywords and keyphrases by assessing the significance of N-grams that occur in a given text.

To extract keyphrases from a text using N-grams and traditional language-processing tools, we can follow the steps below:

1. Pre-process and clean the text by removing stop words using an appropriate dictionary.
2. Extract nouns and their frequency from the text to create a dictionary of key terms, as they often encapsulate the main concepts.
3. Extract N-grams around the identified key terms. This contextual information can provide a more nuanced understanding of the role of the noun in the text. See also Figure 5.
4. Select the top N-grams based on their term frequency [34] and inverse document frequency-based [35] (TF-IDF) using a threshold for the TF-IDF score.

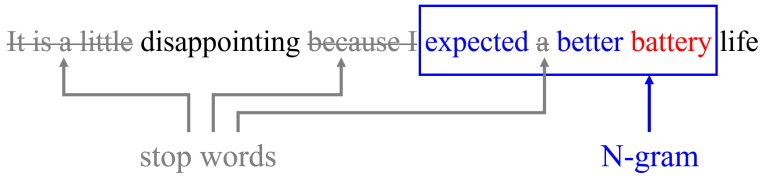

Figure 5. Extracting a keyphrase candidate from a simple review using an N-gram-based approach ($N = 3$).

Although it is a straightforward method, it has limitations. For instance, it cannot eliminate synonyms and variations. Furthermore, the co-occurrence of phrases should also be considered, for example, by using techniques like collocation extraction to find proper keyphrases for a given text.

5.1.2. Dependency Parsing

In the case of this method, we break down the reviews into sentences and identify their respective keywords. Then, we look for the context of the keywords, which we achieve by applying dependency analysis. An example of dependency analysis is shown in Figure 6. During this process, we parse the whole sentence to obtain its grammatical structure and identify "head" words and words that are grammatically connected to them. After this, we identify the keyword *kw* and its "head" word *h*; then, we look for words that were connected to *h* while keeping the original order of the words in the sentence. We look for words connected to *h* instead of *kw* because *kw* was usually an adjective, adverb, or some other word that expresses emotions or qualities and has no particular meaning by itself. Therefore, instead of looking for words connected to this term, we focused on the word that is grammatically connected to *kw* (e.g., a noun). During the search procedure, we exclude common words, such as prepositional modifiers and words representing coordinating conjunctions. This way, the resulting phrase contained the most important parts (nouns, adjectives, etc.).

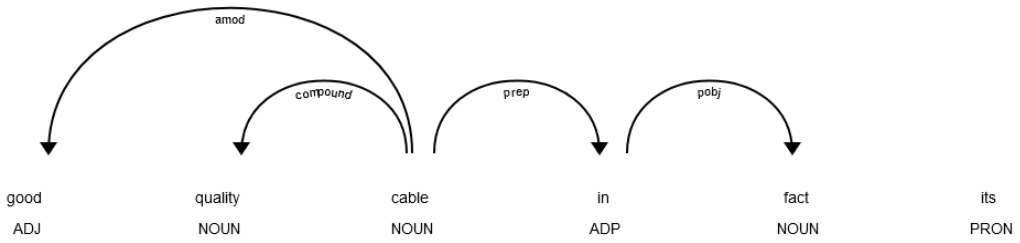

Figure 6. An example of dependency parsing.

This procedure results in very long phrases that are sometimes not much shorter than the original sentence. To decrease their lengths, we integrated a thresholding mechanism into the search procedure to decide which keywords to keep and which to discard. This thresholding was based solely on the sentiment of the keyword: positive keywords were kept if their sentiment score [36] was above 0.89, while negative and neutral ones were kept if their score was above 0.79. The exact thresholding levels were calculated based on our training set and were optimized based on the average length of the resulting contexts and their interpretability. Optimization was a step-by-step method with expert reinforcement. During this, starting from a still meaningful lower value of 0.49 with a step size of 0.1, we produced all possible outputs of a 100-item randomly sampled validation set, based on which the optimal parameters were defined, relying on evaluation by human experts. We performed this step-by-step optimization for both positive and negative keywords.

5.1.3. Embedding

One commonly used technique to measure the similarity between two words, phrases, or even sentences is to transform them into embedded vectors. The advantage of converting into a vector is that the semantic relationship between individual vectors can be measured using cosine similarity. By measuring the similarity, we can find the expressions in a given sentence whose meaning is closest to the meaning of the entire sentence.

The cosine similarity, mathematically, measures the cosine of the angle between two vectors $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in R^d$ projected in a vector space: the smaller the angle, the higher the cosine similarity.

$$\cos(xy) = \frac{xy}{\|x\|\|y\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} (x_i)^2}\sqrt{\sum_{i=1}^{n} (y_i)^2}} \quad (1)$$

The similarity value is between -1 and 1; full similarity (identity) is described as 1, full dissimilarity is described as -1, and neutral behavior (orthogonality) is described as 0.

Its main advantage is that it is less sensitive to noise thanks to the embedding. The disadvantage is that the performance depends to a large extent on the method of valorization. Therefore, choosing the wrong embedding method can lead to bad results. One implementation of embedded-based keyword extraction is KeyBERT [37]. An example of a KeyBERT output is shown in Figure 7.

```
recently purchased this mobile case and must say it been great addition to my phone accessories The sleek design
not only enhances the look of my device but also provides excellent protection against everyday wear and tear The
material feels durable yet lightweight which is perfect for my on the go lifestyle particularly appreciate the
precise cutouts that allow easy access to all ports and buttons without compromising on protection Plus the added
grip feature on the sides gives me extra confidence when handling my phone While it not the most extravagant case
on the market it certainly gets the job done without breaking the bank Overall quite satisfied with this purchase
and would recommend it to anyone looking for reliable and affordable mobile case
```

Figure 7. An example for keywords extracted by KeyBERT.

5.1.4. Experiments

We compared the embedding, N-gram, and dependency parsing solutions to find the best keyword extraction method. For independent experts to be able to compare and evaluate each model, we produced a test dataset. For our experiments, from the Cell

Phones and Accessories (CPA) dataset, we took a random sample of 100 sentences (see Appendix A). Seven independent human experts evaluated their results.

We randomly selected 100,000 products from this subset and chose the four products with the most reviews. We introduced this step to ensure that there would be a sufficient number of reviews for the same product, including a sufficient number of explicitly positive and negative samples. The four products with the most reviews already provided a sufficient sample for further narrowing. In the next step, we removed the neutral reviews with a rating of 3. This step was introduced because our experience had shown that extracting information from extreme emotional expressions is more difficult. Words with a strong emotional charge and negative sentences can obscure the information, making it difficult to extract.

The remaining reviews were split into sentences and sub-sentences, and we corrected the grammatical abbreviations. We then removed sentences containing fewer than 6 or more than 14 words. In our experience, sentences shorter than 6 words generally do not contain meaningful information, and the 14-word upper limit (by using the splitting into sub-sentences) fits well with the length of an average English sentence. From the remaining sentences, we randomly selected 100.

Finally, N-gram, dependency parsing, and semantic embedding-based keyword extraction methods were applied to each of the 100 sentences. The keywords generated using the tested models were evaluated by independent experts, who voted for each sentence which method best extracted the meaning of that sentence. The experts could vote for multiple models or choose the "none of them" category.

The evaluation's result (see Table 1) showed that the embedding-based technique dominated in 61% of the cases. Therefore, we implemented this method into our pipeline.

Table 1. Evaluation of keyword extraction models.

Models	Votes
None of them	15
Embedded	61
N-gram	15
Dependency parsing	9

5.2. Pipeline

We implemented the information extraction process as a single pipeline. We designed the pipeline to ensure that the model applied can evolve with the growth and changes in the dataset. In addition, it needs to be stable and easy to maintain so that it can be made available as a service to other applications.

5.2.1. Text Cleaning

The purpose of text cleaning is to remove characters and text elements from the raw data that do not contain relevant information and may distort the results obtained by the model. This is the first level of our pipeline, where we performed these text transformations to ensure that the text meets the tokenization requirements of the model.

As described in more detail in Section 3, our requirement of tokenization is to work properly with the specialized BERT+R neural network. Text cleaning removes punctuation and unnecessary spaces that we think may cause noise. In addition, the entire text was lower-cased, and language-specific abbreviations were removed to meet the tokenization requirements of the BERT+R model used.

5.2.2. Splitting the Text into Sentences

A single review may contain multiple relevant topics that express the opinion of the customer. In addition, one sentence or sub-sentence usually contains one or more

statements about the same topic, and users make several statements within a sentence without using punctuation.

Therefore, in our implementation, we divided the text by sentence- and sub-sentence-ending punctuation used in English. Thus, the information carried by each statement is extracted sentence by sentence.

5.2.3. Generating Regression-Based Sentiment Values

Dataset labels are formed by user ratings. This means that users rated their elements on a scale from 1 to 5. This means a five-element classification. However, our goal was to create a more sophisticated solution. We have therefore created a model that can predict the sentiment of a sentence containing a statement on a scale from 1 to 5 with high accuracy. For this purpose, we used a pre-trained BERT network and modified its architecture by adding a regression layer to the output instead of a classification layer. We then fine-tuned the specific BERT+R model on our dataset. The resulting model, depicted in Figure 8, allows us to predict sentiment values with finer granularity. Furthermore, our system has become more sensitive to the differences between negation and assertion sentences, and this function of our system allows us to filter the texts based on sentiment.

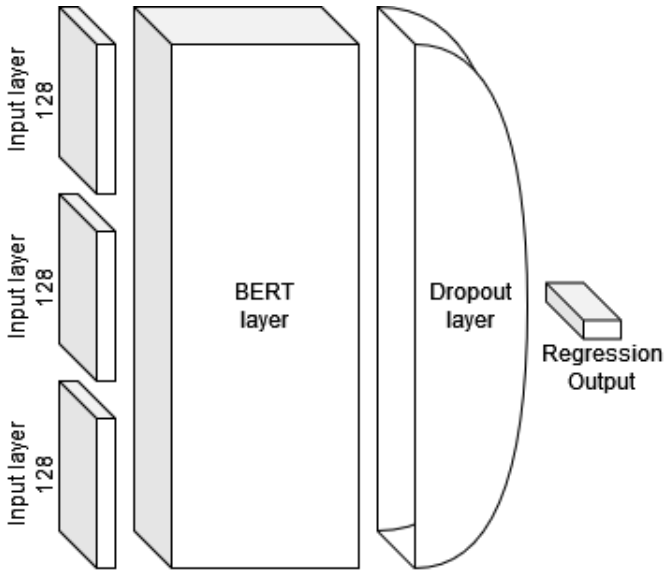

Figure 8. The BERT+R architecture modified for our system.

After decomposing user reviews into sentences, we predicted sentiment scores with our BERT+R model for each sentence.

5.2.4. Keyphrase Extraction

The Sentence-BERT [38] is a special transformer-based architecture designed for sentence embedding. The network is designed to assign vectors to each sentence such that their similarity is measured by cosine similarity. Sentence-BERT (SBERT) is a modification of the pre-trained BERT network that uses Siamese and triplet network structures to derive semantically meaningful sentence embeddings.

We applied semantic embedding to extract the keywords. For this purpose, the text was split into sentences, and the words belonging to each sentence and their bigram and trigram combinations were embedded separately in a 768-dimensional vector space using SBERT. Then, we measured the cosine distance of the given words and their bigrams and trigrams from the sentence that contained them. In the next step, we selected from each sentence the top three expressions whose semantic meaning was closest to the sentence containing it.

This gave us an item triplet for each sentence, which could contain words and phrases or their negation auxiliaries. In addition, we used combinations of bigrams and tri-

grams with elementary words since these are the N-gram combinations that still make sense in natural languages. In fact, by maximizing the value of N, we can eliminate redundant combinations.

5.2.5. Hierarchical and Density-Based Recursive Clustering

After cleaning, classifying, and extracting the key terms from the text, we then highlighted key terms and their embedded vectors. Finally, we applied our clustering approach to topic search in the nested vector space created from key terms.

A special property of embedded vector spaces generated from text data is that they are difficult to cluster, as the distances between adjacent vectors are usually similar. However, slightly dense clusters appear in the vector space for similar contents. To extract these slightly dense clusters, we need to remove outliers. To achieve this, we used our special hierarchical solution. This consists of using hierarchical clustering with recursive execution, as shown in Figure 9.

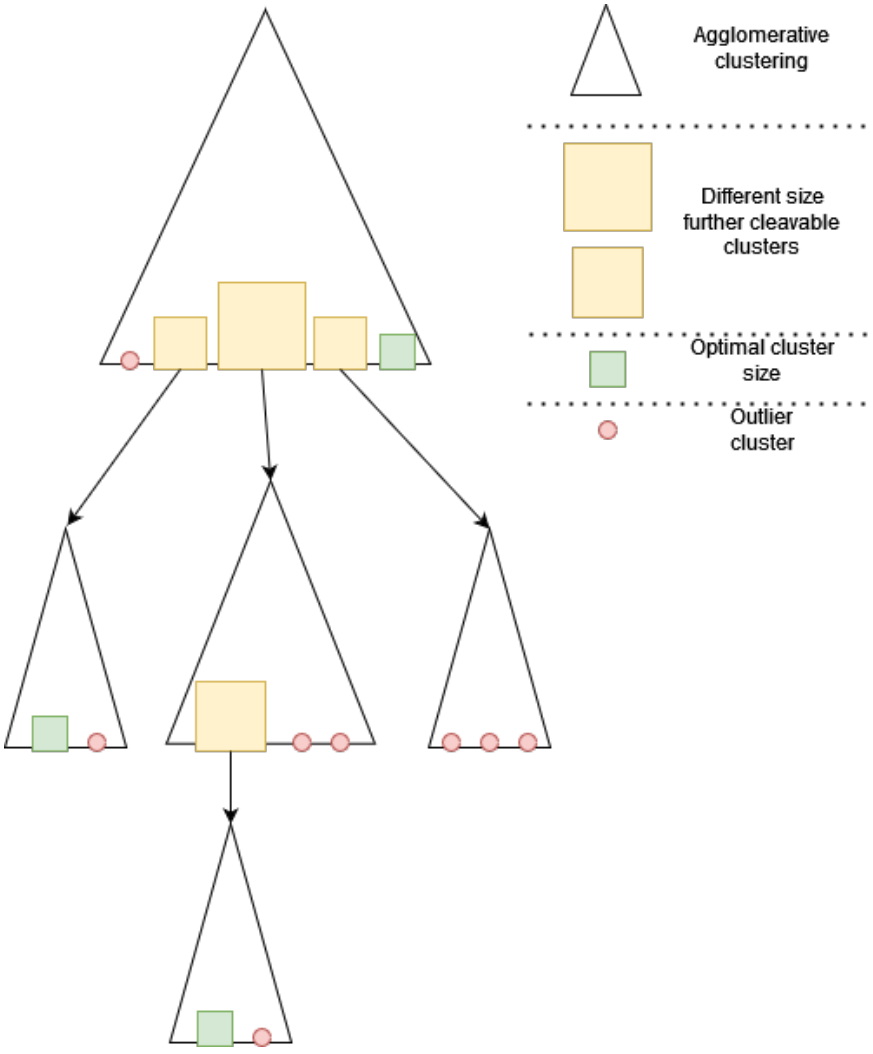

Figure 9. Recursive clustering considered in our system.

Based on our investigations, the elements of slightly dense clusters describe the same topics. Our experience shows that the cosine similarity between these clusters is above 0.7. These slightly denser clusters could be extracted by recursively applying hierarchical clustering. This method re-clusters the clusters with densities below 0.7 in the next clustering cycle. This is repeated until the minimum number of elements (in our case, 5) or a density value of 0.7 is reached. Of course, these hyperparameters of the pipeline

(see Figure 10) can be freely adjusted to improve the generalization ability of our system depending on the nature of the text.

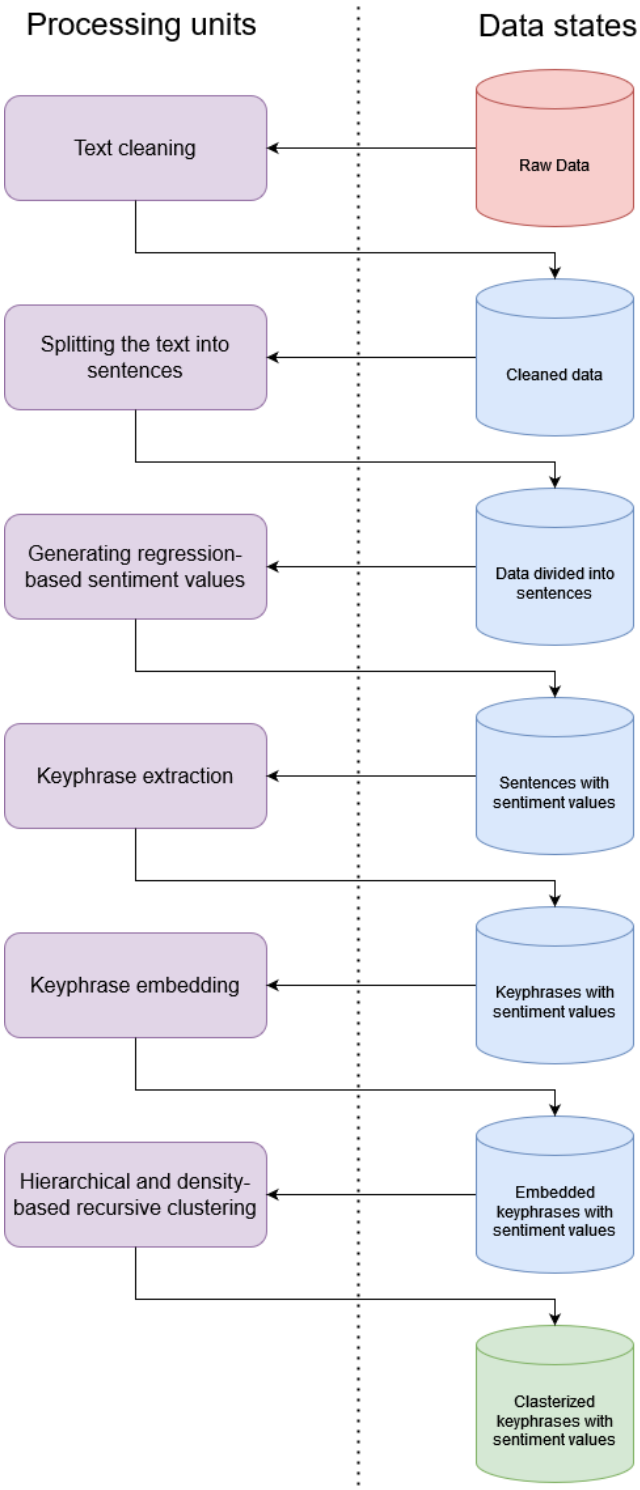

Figure 10. Flow diagram of the pipeline.

6. Results

To assess our own and the referenced models' information-retention capabilities, we brought unsupervised keyword extraction back to a supervised text classification task. Evaluating unsupervised learning models often involves relating them to a supervised

classification task, e.g., as seen in the case of LDA, where authors use binary classification to measure its effectiveness.

If we hypothesize that the models can extract meaningful keywords, the accuracy of an independent classification model using only those extracted keywords should be correlated with the information value of the words. This assumption is supported by the observation that text cleaning, which removes noise, improves classification accuracy. However, this improvement is only seen when elements considered as noise are removed. Consequently, if words with information value for classification are eliminated, the accuracy of the classification models will decrease.

It is plausible that the quality of the extracted words can impact classification outcomes. If the classification accuracy significantly drops compared to when all text elements are used, this suggests that the topics and their associated words may not have been correctly extracted. Conversely, when relevant words are extracted, classification accuracy should ideally increase. Nevertheless, this evaluation method allows for comparing the relevance of words extracted by different models for information retention.

Due to the different approaches of the different models, each model has been set up to maximize their performance as follows:

- BERTopic has limited configuration options, so we optimized the keyword extraction of the model. The BERTopic version we used allowed us to define a maximum of 30 words for a topic.
- For Top2Vec, the *mincount* parameter of the model is responsible for the number of words to extract. Therefore, we systematically tested values from 10 to 600 and found that 500 was the maximum value where the model remained stable.
- The automatic determination of the number of topics is not built into LDA by default, so a more complex optimization was performed for this model. In LDA, a separate parameter can be used to set the number of topics searched for. This parameter was set to be between 2 and 50. Within this range, we tried both online and batch optimization techniques to find the best number of topics. The optimal topic number for LDA has the best average perplexity among the selected topics.
- In the case of our model, we left the parameters at the values we had found during the development. This means that each of the topics (sets) that our model selected had more than five elements, and the cosine similarity value between the elements of each topic was greater than 0.7. Sets outside this range were treated as outliers, i.e., irrelevant.

The topic words extracted from the models were used as a dictionary, and only these words were retained from the datasets for each evaluation of the model for the classification tasks. The texts filtered by the model dictionaries were vectorized using one-hot, count vectorization, and TF-IDF techniques. We then used a regression model for each model evaluation. The advantage of the regression model is that it is simple, and the operation of the model can be clearly explained using weights. In each case, we trained the classification models on a training dataset filtered by the dictionaries generated by the keyword extractor models, and their accuracy was measured on the test set.

In addition, since we tested several parameters for the LDA and Top2Vec models and tried to reach the maximum performance of every model, we considered their average performances. On the binary classification tasks, we measured an average accuracy of 89.82% for our model, 82.75% for LDA with batch optimization, 82.38% for LDA with online optimization, 76.21% for Top2Vec, and 79.46% for BERTopic (see Table 2). In terms of the number of topics found, our model found 58 topics on average, LDA with batch optimization found seven topics, LDA with online optimization found seven topics, To2Vec found three topics, and BERTopic found seven topics, as also shown in Table 3. The detailed results for each binary classification task and topic modeling approach are included in Appendix B.

Table 2. Average accuracy for binary classification.

	Models	TF-IDF	Count Vect.	One-Hot
Baselines	LDA (batch)	83.99%	82.09%	82.25%
	LDA (online)	83.58%	81.69%	81.86%
	Top2Vec	77.11%	76.04%	75.48%
	BERTopic	80.21%	79.57%	78.61%
	Our pipeline	**90.92%**	**89.18%**	**89.35%**

Table 3. Number of topics and vocabulary sizes for binary classification.

	Models	Topic Number	Vocabulary Size
Baselines	BERTopic	7	127
	LDA (batch)	7	5859
	LDA (online)	7	6061
	Top2Vec	3	100
	Our pipeline	**58**	**3867**

For the multi-classification tasks, our model provided an average accuracy of 45.24%, and we obtained 35.73% for LDA with batch optimization, 35.48% for LDA with online optimization, 39.41% for Top2Vec, and 42.8% for BERTopic; see Table 4. In terms of the number of topics found, our model found 64 topics on average, LDA with batch optimization found two topics, LDA with online optimization found two topics, Top2Vec found 17 topics, and BERTopic found 94 topics; see Table 5.

Table 4. Average accuracy for multiple classification.

	Model	TF-IDF	Count Vect.	One-Hot
Baselines	BERTopic	46.13%	40.27%	42.00%
	LDA (batch)	37.29%	34.87%	35.03%
	LDA (online)	36.98%	34.41%	35.03%
	Top2Vec	41.88%	37.89%	38.47%
	Our pipeline	**48.47%**	**43.33%**	**43.93%**

Table 5. Average number of topics and vocabulary sizes for multiple classifications.

	Models	Topic Number	Vocabulary Size
Baselines	BERTopic	92	1357
	LDA (batch)	2	2242
	LDA (online)	2	2275
	Top2Vec	17	371
	Our pipeline	**64**	**2860**

7. Conclusions

This study investigated the inherent challenges of extracting relevant information from unstructured free texts. We demonstrated that the widely used methods are not suitable to comprehensively address more complex problems. Our experiments confirmed the current lack of a single, universally applicable solution for keyword/keyphrase extraction and topic modeling.

In response, we developed a novel pipeline by iteratively reviewing, combining, and refining various potential solutions. This consolidated pipeline streamlines the information extraction process through efficient data cleaning, keyphrase identification, and topic organization. Notably, this framework demonstrates increased efficiency compared to the existing literature models.

Furthermore, our pipeline exhibits robust information-retention capabilities. By leveraging semantic embedding, the system excels at identifying key phrases and their associated

topics, delivering higher quality results than alternative solutions within the given text processing domain. Additionally, the modular architecture facilitates the seamless integration of future components, adapting the pipeline to address diverse tasks.

Author Contributions: Conceptualization, methodology, writing, and visualization, R.L., G.B., B.H., I.L., A.T., J.T. and M.S.; validation, J.T. and B.H.; review and editing, J.T. and A.H.; supervision, A.H. and B.H.; project administration, B.H.; funding acquisition, A.H. and B.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partly funded by the projects GINOP-2.3.2-15-2016-00005, supported by the European Union and co-financed by the European Social Fund, and by the projects TKP2021-NKTA-34 and 2020-1.1.2-PIACI-KFI-2021-00223, implemented with the support provided by the National Research, Development, and Innovation Fund of Hungary under the TKP2021-NKTA and 2020-1.1.2-PIACI KFI funding schemes respectively. Further funding was provided by the Project no. KDP-2021 that has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the C1774095 funding scheme.

Data Availability Statement: The authors declare that the publicly available Amazon Reviews (2018) and 20 Newsgroups datasets were used to conduct the experiments on which this article is based. Apart from the selection of subsets from the datasets, no other transformation was performed.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A

Here, we summarize all the human expert evaluations of the performance of the different keyword extraction approaches we studied. The experts could also vote for more than one keyword extraction solution in the case of every text. The Amazon Reviews (2018) dataset was used for this evaluation. Next, we present the extracted keywords and the corresponding votes for 100 reviews from this dataset.

Table A1. Expert evaluation of keyword extraction models for reviews 1 to 23.

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
	if anyone has had a similar failure post a comment					
anyone similar failure	failure post	failure post	0	6	1	2
	well this past weekend i went on a kayak camping trip					
ak camping trip	weekend camping trip	weekend camping trip	0	1	7	2
	it will not charge your phone while it is in a case					
not charge phone	charge phone case	charge phone case	0	7	1	0
	this is the best quality powerbank i have owned					
best quality power	best quality	best quality	0	1	2	7
	better than plugging in you phone					
better plug ging	better plugging phone	better plugging phone	3	0	3	0
	definitely recommended to anyone looking for a good, quick car charger					
good quick car	car charger	car charger	0	0	2	7
	for the money, i wish anker would be more reliable					
money wish ker	wish anker	wish anker	7	0	0	0
	heats up your phone a lot and does not charge up past 60%					
heats phone lot	heats phone lot	heats phone lot	0	7	7	2

Table A1. *Cont.*

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
	love that it is not very heavy and has a super sleek look					
sleek look	love sleek look	love sleek look	0	7	4	4
	didt work as the first one					
first one	work didt	work didt	0	3	2	5
	no cables, nothing to plug in					
no cables nothing	cables nothing	cables nothing	0	7	0	3
	i thought this was a fast charging, but i have realized that is not					
charging realized not	fast charging	fast charging	2	5	0	0
	i have to admit i am an anker addict					
ker addict	anker addict	anker addict	0	0	7	1
	charges her battery pack, iphone, ipad, and lipstick battery all at the same time					
pack iphone ipad	pack iphone battery	pack iphone battery	7	0	0	0
	it charges 5 things at once					
charges 5 things	charges things	charges things	0	7	1	1
	anker apologized for my inconvenience and sent me a replacement within 2 days					
within 2 days	anker apologized inconvenience	anker apologized inconvenience	0	0	7	3
	i bought this not to long ago and now it is already not working					
	bought not long ago	bought not long ago				
ago already not	already not working	already not working	0	7	2	0
	working great using the basic wall unit from apple					
wall unit apple	unit apple	unit apple	5	2	0	0
	this charger does not do quick charge					
not do quick	charger charge	charger charge	1	6	0	3
	no big deal, right, still have two three port chargers					
three port chargers	port chargers	port chargers	0	6	1	1
	loved it for 3 days and now it starts to charge and stops immediately					
loved 3 days	stops immediately	stops immediately	0	1	7	7
	i am so sick of poor quality imports					
sick poor quality	sick quality	sick quality	0	7	0	0

Table A2. Expert evaluation of keyword extraction models for reviews 24 to 48.

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
	i did not return it in time, so i get to eat the cost					
get eat cost	get eat cost	get eat cost	2	5	5	0
	the problem is, this new one, only charges it normally					
only charges normally	charges normally	charges normally	0	7	3	0
	i used both, but more so the smaller model					
used smaller model	smaller model	smaller model	1	6	1	1
	super glad i got one of these					
super glad got	super glad got	super glad got	0	7	7	7
	then within seconds it turns off					
seconds turns off	within seconds turns	within seconds turns	1	5	2	0
	it is a little disappointing because i expected more from anker					
disappointing expected more	disappointing anker	disappointing anker	0	7	0	0

Table A2. *Cont.*

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
not an issue for me because i am at work						
not issue work	not issue work	not issue work	5	2	2	0
anker is very reliable though so i would stick with this brand						
very reliable though	anker reliable though	anker reliable though	0	6	5	3
one remaining works 75% of the time						
works 75 time	time works	time works	1	5	0	0
it can easily take 8-12 hours because it used micro usb to charge						
micro usb charge	micro usb charge	micro usb charge	1	6	6	0
this was not any different and lives up to the brands reputation						
lives up brands	lives brands reputation	lives brands reputation	0	7	6	0
so when anker let me give you my opinion after testing it out						
give opinion testing	opinion testing	opinion testing	0	6	1	0
just got it so will add on if something significant comes up						
add something significant	something significant	something significant	2	2	0	3
still a good h charger for the price						
good h charge	charger price	charger price	2	4	1	1
hard to get phone in the right spot						
hard to get	hard get	hard get	2	5	1	1
takes forever to charge the actual product when need charging charge over night						
charging charge night	forever charge	forever charge	0	7	1	7
works better than samsung pod which does not work at all						
works better samsung	works better pod	works better pod	0	6	0	1
i love the first one so much i am giving this as a gift						
love first much	love gift	love gift	1	6	0	1
as such, it is not a replacement for the original charger						
not replacement original	replacement original charger	replacement original charger	0	7	0	0
but mine come with one usb port not working on day one						
usb port not	mine usb port	mine usb port	4	3	0	2
works well in my subaru outback does not pop out from regular road vibrations						
regular road vibrations	works subaru outback	works subaru outback	1	1	6	0
its subtle enough to be classy and informative						
subtle enough class	classy informative	classy informative	0	1	6	6
i use it to power my hotspot device on the go						
use power hot	power hotspot device	power hotspot device	1	0	6	1
good but it is too long in size						
long size	long size	long size	0	7	7	5
i need something dependable for charging						
need something depend	need something	need something	3	3	1	1
always the best for the price point						
always best price	best price	best price	0	5	5	1
now i have to shop again and it will not be for this						
shop again not	shop not	shop not	1	5	1	0
i am constantly checking for new releases						
constantly checking new	constantly releases	constantly releases	2	4	1	0

Table A3. Expert evaluation of keyword extraction models for reviews 49 to 74.

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
v at the receiving end of a 3 foot cable at 1a load						
3 foot cable	foot cable load	foot cable load	0	5	1	0
i have owned this battery bank for over a year and could not be happier						
battery bank year	bank year	bank year	6	1	0	0
takes forever to charge a phone						
forever charge phone	forever charge	forever charge	0	3	1	6
will not charge my phone for longer than a few seconds						
not charge phone	phone seconds	phone seconds	1	6	0	0
and now after 1month use the charger will not fully charge						
not fully charge	use charger charge	use charger charge	0	7	0	0
after troubleshooting, 2 of the ports must have gone bad						
must gone bad	troubleshooting ports	troubleshooting ports	1	5	1	1
only worked for a couple weeks						
worked couple weeks	only worked couple weeks	only worked couple weeks	0	6	6	0
because it looks nice so i give it a 2 stars						
nice give 2	nice stars	nice stars	0	4	0	4
also when i have any question they are nicely to help						
question nicely help	also question nicely help	also question nicely help	0	7	4	1
i ordered this for myself for christmas						
ordered myself christmas	ordered christmas	ordered christmas	0	6	3	3
anker has recently become my go to brand for charging accessories						
brand charging accessories	anker brand charging	anker brand charging	2	5	1	0
did not work well with my phone switched to samsung and now have no problems						
samsung no problems	phone samsung problems	phone samsung problems	0	7	0	0
i am very happy with it and definitely would recommend to others						
happy definitely would	happy others	happy others	0	2	0	6
the best charger we have ever had and we have used quite a few						
ever used quite	best charger	best charger	0	3	5	5
i can not say enough about this portable charger						
portable charge r	portable charger	portable charger	1	4	4	3
nope, anker did not pay me to say that						
not pay say	nope anker	nope anker	1	5	1	1
i bought this charger to have something that can accommodate all my charging needs						
accommodate charging needs	bought charger	bought charger	0	6	1	0
does a great job staying out of the way						
great job staying	job staying way	job staying way	4	3	1	3
this is a powerful little guy						
powerful little guy	powerful guy	powerful guy	0	7	6	7
this power bank has a huge capacity and worked well, but only for a while						
capacity worked well	bank huge capacity	bank huge capacity	0	4	5	5
does not stay plugged into the accessory outlet						
not stay plug	stay accessory	stay accessory	0	7	0	0
it will also not charge my phone						
not charge phone	charge phone	charge phone	0	7	0	0
it will not supply power to my iphone now						
not supply power	power iphone	power iphone	0	7	0	0

Table A3. *Cont.*

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
must be in the center of the phone as expected, for the charge						
must center phone	phone charge	phone charge	1	6	0	0
that is not the case with this anker charger						
not case an	case anker charger	case anker charger	3	0	2	1

Table A4. Expert evaluation of keyword extraction models for reviews 75 to 100.

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
chargers my s7 edge faster than any other charger that i have or tried						
faster charge r	chargers faster charger	chargers faster charger	0	3	3	3
purchased on 3-sep-2015, worked fine till it died on 3-jun-2016						
2015 worked fine	till till	till till	4	2	0	0
we bought two of these anker 40w/5-port chargers a while back						
two ker 40	anker chargers	anker chargers	2	4	2	0
this is an update to my review of this product earlier						
update review product	update review	update review	0	6	5	2
purchased a powerport 5 in sep 2014						
5 sep 2014	sep 2014	sep 2014	4	2	0	0
overall i am still satisfied with the wireless charger						
still satisfied wireless	satisfied charger	satisfied charger	0	3	4	5
does not give a true charge						
not give true	does not give true charge	does not give true charge	1	1	6	0
wish i could give this product no stars						
product no stars	wish product	wish product	1	6	0	0
they were great while they lasted						
great they lasted	great lasted	great lasted	3	3	0	2
i have tried two different email addresses marketing@anker						
email addresses marketing	email addresses	email addresses	0	4	0	4
it is been dead for past several month, i should probably throw in trash						
dead past several dead past several	dead past several month probably throw trash	dead past several month probably throw trash	0	2	3	3
i really like it is portability						
really like it	really like portability	really like portability	0	1	7	5
totally safe to use and a solidly built device						
totally safe use	safe use	safe use	0	7	4	4
pretty light does not take alot of space						
light does not	light alot	light alot	4	0	1	2
this is a great product and charges my devices many times						
great product charges	great product	great product	0	3	7	7
it charges my phone just from being placed on it						
charges phone placed	charges phone	charges phone	1	4	5	5
i made sure to charge it before, then i packed it in my pack						
charge packed pack	charge pack	charge pack	3	4	1	0
we have a few of these and now we are having problems						
having problems	problems	problems	0	6	4	5

Table A4. *Cont.*

Reviews/Models/Extracted Keywords			Experts' Votes			
Embedded (E)	**N-Gram (N)**	**Dependency Parsing (DP)**	**None**	**E**	**N**	**DP**
this one was no better						
one no better	no better	no better	0	3	5	3
i have gotten better response times from a 3rd party seller with bad ratings						
3rd party seller	party seller ratings	party seller ratings	4	3	0	0
got that all figured out after a bit						
figured out after	figured bit	figured bit	1	6	0	0
it is a little heavy but it is good						
little heavy good	little heavy good	little heavy good	0	3	3	5
then i plugged it into my car, all works fine did not notice anything						
car works fine	car works	car works	3	4	0	0
i enjoyed using this i guess						
enjoyed using guess	enjoyed guess	enjoyed guess	0	1	1	7
got it for my brother as a gift						
my brother gift	brother gift	brother gift	0	6	6	3

Appendix B

Here, we summarize all the models and their outputs in terms of the number of topics and vocabulary sizes they found for the 20 Newsgroups dataset. We also present their results for a classification task using different embedding strategies. The results for each model are presented in separate tables below. The columns Topic number and Vocabulary size contain the averages of the different runs for each model. It can be seen that our approach outperforms the others with respect to the classification problem investigated.

Table A5. Results of the LDA model with batch optimization.

Classes	Topic Number	Vocabulary Size	TF-IDF	Count Vectorization	One-Hot
0 vs. 1	4	4730	84.67%	82.22%	82.42%
0 vs. 2	4	5626	82.99%	78.83%	78.99%
0 vs. 3	10	6130	84.78%	82.46%	82.13%
0 vs. 4	4	6067	84.58%	82.38%	82.25%
0 vs. 5	3	5008	84.88%	83.26%	83.42%
1 vs. 2	5	5568	81.09%	78.55%	78.46%
1 vs. 3	4	4382	83.41%	82.21%	82.3%
1 vs. 4	4	5489	81.71%	80.21%	80.15%
1 vs. 5	3	4526	84.98%	83.7%	83.88%
2 vs. 3	30	8714	86.66%	84.63%	85.19%
2 vs. 4	5	6364	80.66%	77.98%	78.67%
2 vs. 5	10	7186	84.18%	82.26%	83.34%
3 vs. 4	5	6220	86.32%	85.28%	85.27%
3 vs. 5	10	6580	86.41%	85.68%	85.8%
4 vs. 5	3	5307	82.56%	81.7%	81.55%

Table A6. Results of the LDA model with online optimization.

Classes	Topic Number	Vocabulary Size	TF-IDF	Count Vectorization	One-Hot
0 vs. 1	4	4910	84.76%	82.37%	82.19%
0 vs. 2	4	5790	82.39%	78.2%	78.59%
0 vs. 3	10	6247	84.6%	82.44%	82.13%
0 vs. 4	4	6147	83.72%	81.35%	81.47%
0 vs. 5	3	5049	84.98%	83.33%	83.67%
1 vs. 2	5	5775	80.16%	77.41%	77.29%
1 vs. 3	4	4647	83.37%	82.34%	82.45%
1 vs. 4	4	5721	80.89%	79.44%	79.31%
1 vs. 5	3	4645	84.79%	83.63%	83.88%
2 vs. 3	30	9135	86.55%	84.62%	85.21%
2 vs. 4	5	6772	80.02%	77.43%	78.08%
2 vs. 5	10	7307	83.27%	81.44%	82.23%
3 vs. 4	5	6342	86.03%	85.22%	85.26%
3 vs. 5	10	6765	86.23%	85.29%	85.29%
4 vs. 5	3	5671	81.96%	80.88%	80.85%

Table A7. Results of the Top2Vec model.

Classes	Topic Number	Vocabulary Size	TF-IDF	Count Vectorization	One-Hot
0 vs. 1	2	66	73.86%	71.83%	71.89%
0 vs. 2	2	56	62.25%	61.19%	60.72%
0 vs. 3	4	142	81.73%	80.12%	80.17%
0 vs. 4	3	94	76.98%	75.21%	75.23%
0 vs. 5	3	132	84.16%	83.84%	83.02%
1 vs. 2	2	86	67.61%	66.27%	66.44%
1 vs. 3	4	138	84.31%	83.99%	83.04%
1 vs. 4	4	150	77.57%	75.46%	75.64%
1 vs. 5	3	116	82.58%	82.56%	81.19%
2 vs. 3	3	120	81.79%	80.98%	80.34%
2 vs. 4	2	86	68.99%	67.87%	67.24%
2 vs. 5	1	51	67.38%	67.86%	66.18%
3 vs. 4	2	62	84.31%	83.07%	82.52%
3 vs. 5	2	90	88.13%	87.12%	86.8%
4 vs. 5	3	106	75.0%	73.21%	71.81%

Table A8. Results of the BERTopic model.

Classes	Topic Number	Vocabulary Size	TF-IDF	Count Vectorization	One-Hot
0 vs. 1	11	161	83.95%	86.62%	85.35%
0 vs. 2	2	58	63.31%	62.29%	60.76%
0 vs. 3	21	374	90.63%	87.8%	87.29%
0 vs. 4	3	67	81.54%	75.56%	78.88%
0 vs. 5	3	69	83.99%	82.85%	82.59%
1 vs. 2	4	71	75.38%	77.27%	75.51%
1 vs. 3	7	134	86.01%	86.13%	85.24%
1 vs. 4	5	89	77.11%	76.84%	75.53%
1 vs. 5	4	70	83.5%	85.01%	84.76%
2 vs. 3	12	216	90.08%	91.09%	88.8%
2 vs. 4	2	60	67.89%	65.53%	62.5%
2 vs. 5	2	60	67.0%	68.01%	67.38%
3 vs. 4	16	318	92.04%	91.91%	90.58%
3 vs. 5	5	98	91.24%	90.86%	92.01%
4 vs. 5	2	60	69.42%	65.75%	61.94%

Table A9. Results of pipeline-based model.

Classes	Topic Number	Vocabulary Size	TF-IDF	Count Vectorization	One-Hot
0 vs. 1	49	3238	90.96%	89.04%	88.79%
0 vs. 2	73	4404	86.24%	85.35%	86.11%
0 vs. 3	55	3266	93.2%	89.86%	89.73%
0 vs. 4	63	4088	91.24%	88.45%	90.44%
0 vs. 5	79	4608	91.99%	91.87%	92.12%
1 vs. 2	42	3367	87.25%	84.22%	83.71%
1 vs. 3	35	2567	92.37%	89.44%	89.31%
1 vs. 4	45	3311	87.63%	85.13%	86.18%
1 vs. 5	51	3777	91.81%	91.18%	91.81%
2 vs. 3	54	3645	92.62%	91.09%	90.46%
2 vs. 4	62	4381	88.42%	85.92%	86.58%
2 vs. 5	76	5010	90.43%	88.79%	88.04%
3 vs. 4	50	3698	94.43%	94.69%	93.5%
3 vs. 5	64	4059	94.29%	93.91%	93.91%
4 vs. 5	70	4630	90.94%	88.71%	89.5%

References

1. Zhai, C. Statistical language models for information retrieval. *Synth. Lect. Hum. Lang. Technol.* **2008**, *1*, 1–141.
2. Kübler, S.; McDonald, R.; Nivre, J. Dependency parsing. *Synth. Lect. Hum. Lang. Technol.* **2009**, *1*, 1–127.
3. Jain, A.K.; Murty, M.N.; Flynn, P.J. Data clustering: A review. *ACM Comput. Surv. (CSUR)* **1999**, *31*, 264–323. [CrossRef]
4. Xu, R.; Wunsch, D. *Clustering*; John Wiley & Sons: Hoboken, NJ, USA, 2008; Volume 10.
5. Falush, D.; Stephens, M.; Pritchard, J.K. Inference of population structure using multilocus genotype data: Linked loci and correlated allele frequencies. *Genetics* **2003**, *164*, 1567–1587. [CrossRef] [PubMed]
6. Pritchard, J.K.; Stephens, M.; Donnelly, P. Inference of population structure using multilocus genotype data. *Genetics* **2000**, *155*, 945–959. [CrossRef]
7. Angelov, D. Top2vec: Distributed representations of topics. *arXiv* **2020**, arXiv:2008.09470.
8. Grootendorst, M. BERTopic: Leveraging BERT and c-TF-IDF to create easily interpretable topics. *arXiv* **2022**, arXiv:2203.05794.
9. Rajaraman, A.; Ullman, J.D. *Mining of Massive Datasets*; Cambridge University Press: Cambridge, UK, 2011.
10. Dicle, M.F.; Dicle, B. Content analysis: Frequency distribution of words. *Stata J.* **2018**, *18*, 379–386. [CrossRef]
11. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
12. Goldberg, Y.; Levy, O. word2vec Explained: Deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv* **2014**, arXiv:1402.3722.
13. Joulin, A.; Grave, E.; Bojanowski, P.; Douze, M.; Jégou, H.; Mikolov, T. Fasttext. zip: Compressing text classification models. *arXiv* **2016**, arXiv:1612.03651.
14. Raschka, S. *Python Machine Learning*; Packt Publishing Ltd.: Birmingham, UK, 2015.
15. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent dirichlet allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
16. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
17. Arthur, D.; Vassilvitskii, S. *k-Means++: The Advantages of Careful Seeding*; Technical Report; Soda: Stanford, CA, USA, 2006.
18. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; Volume 96, pp. 226–231.
19. Campello, R.J.; Moulavi, D.; Zimek, A.; Sander, J. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discov. Data (TKDD)* **2015**, *10*, 1–51. [CrossRef]
20. Pearson, K. LIII. On lines and planes of closest fit to systems of points in space. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **1901**, *2*, 559–572. [CrossRef]
21. Halko, N.; Martinsson, P.G.; Tropp, J.A. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. *arXiv* **2009**, arXiv:0909.4061.
22. Szlam, A.; Kluger, Y.; Tygert, M. An implementation of a randomized algorithm for principal component analysis. *arXiv* **2014**, arXiv:1412.3510.
23. Lawrence, N.D. A unifying probabilistic perspective for spectral dimensionality reduction: Insights and new models. *J. Mach. Learn. Res.* **2012**, *13*, 1609–1638.
24. Lee, J.A.; Verleysen, M. *Nonlinear Dimensionality Reduction*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2007.

25. Van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
26. Hoffman, M.D.; Blei, D.M.; Wang, C.; Paisley, J. Stochastic variational inference. *J. Mach. Learn. Res.* **2013**, *14*, 1303–1347.
27. Hoffman, M.; Bach, F.; Blei, D. Online learning for latent dirichlet allocation. *Adv. Neural Inf. Process. Syst.* **2010**, *23*, 856–864.
28. Harris, Z.S. Distributional structure. *Word* **1954**, *10*, 146–162. [CrossRef]
29. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
30. Ni, J.; Li, J.; McAuley, J. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 188–197. [CrossRef]
31. Rennie, J. 20 Newsgroups Dataset. Available online: http://qwone.com/~jason/20Newsgroups/ (accessed on 22 January 2024)
32. Schuster, M.; Nakajima, K. Japanese and Korean voice search. In Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan 25–30 March 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 5149–5152.
33. Foundation, A.S. Apache Hadoop. Available online: https://hadoop.apache.org/ (accessed on 22 January 2024)
34. Luhn, H.P. A statistical approach to mechanized encoding and searching of literary information. *IBM J. Res. Dev.* **1957**, *1*, 309–317. [CrossRef]
35. Jones, K.S. A statistical interpretation of term specificity and its application in retrieval. *J. Doc.* **1972**, *28*, 11–21. [CrossRef]
36. Mäntylä, M.V.; Graziotin, D.; Kuutila, M. The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. *Comput. Sci. Rev.* **2018**, *27*, 16–32. [CrossRef]
37. Grootendorst, M. KeyBERT. Available online: https://maartengr.github.io/KeyBERT/ (accessed on 22 January 2024)
38. Reimers, N.; Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv* **2019**, arXiv:1908.10084.