applied sciences

MDPI

Article

Re-Engineered Word Embeddings for Improved Document-Level Sentiment Analysis

Su Yang [1,*] and Farzin Deravi [2]

[1] Department of Computer Science, Faculty of Science and Engineering, Swansea University, Swansea SA1 8EN, UK
[2] School of Engineering, University of Kent, Canterbury CT2 7NZ, UK
* Correspondence: su.yang@swansea.ac.uk

Abstract: In this paper, a novel re-engineering mechanism for the generation of word embeddings is proposed for document-level sentiment analysis. Current approaches to sentiment analysis often integrate feature engineering with classification, without optimizing the feature vectors explicitly. Engineering feature vectors to match the data between the training set and query sample as proposed in this paper could be a promising way for boosting the classification performance in machine learning applications. The proposed mechanism is designed to re-engineer the feature components from a set of embedding vectors for greatly increased between-class separation, hence better leveraging the informative content of the documents. The proposed mechanism was evaluated using four public benchmarking datasets for both two-way and five-way semantic classifications. The resulting embeddings have demonstrated substantially improved performance for a range of sentiment analysis tasks. Tests using all the four datasets achieved by far the best classification results compared with the state-of-the-art.

Keywords: sentiment analysis; semantic classification; feature re-engineering; NLP

Citation: Yang, S.; Deravi, F. Re-Engineered Word Embeddings for Improved Document-Level Sentiment Analysis. *Appl. Sci.* **2022**, *12*, 9287. https://doi.org/10.3390/app12189287

Academic Editors: Jae-Hoon Kim and Kichun Lee

Received: 9 August 2022
Accepted: 15 September 2022
Published: 16 September 2022

1. Introduction

Automated sentiment analysis has been receiving much attention in recent years with its importance continuously increasing due to the surge in the availability and use of large textual databases. Sentiment analysis (or opinion mining) could be defined as the task of interpreting the opinions of authors about specific entities [1]. One of the classical scenarios is to find the opinion expressed by a piece of text, i.e., whether it is semantically positive or negative. Nowadays, there is a great necessity in the industry for automatically detecting such opinions across large datasets, for example, the analysis of Amazon product reviews and Twitter reviews. Such existing needs, as well as the expensive and arguably subjective opinion mining by human alternatives, have made the decision making based on machine learning the de facto choice for large-scale sentiment analysis tasks.

Indeed, there have been more than 7000 articles published to address various aspects of sentiment analysis [1], and more related research is conducted each year [2,3]. These analyses could be generally categorized into document-level, sentence-level, aspect-based, and comparative sentiment analysis [1]. Depending on the nature of the available data and the application scenario, different levels of textual analysis may also be combined to enhance the performance. The most common, but still yet to be fully resolved, problem is the document-level sentiment analysis.

A number of recent studies have been focusing on opinion mining using different customer review databases, thanks to their relatively polarized opinions in principle. For example, Vashishtha and Susan proposed a method using sentiment scoring and fuzzy entropy for unsupervised binary (positive and negative) semantic classification [4]. Two movie review databases were used to evaluate their method. For the Pang–Lee dataset [5], an accuracy of 70% and F_1-score of 0.701 were reported, which was an improvement of

about 5% compared to other reported works used for comparison in their study. For the IMDB Review dataset [6] they employed, the accuracy and F_1-score reduced slightly to 69.3% and 0.691, respectively. These results seem to suggest that the state-of-the-art results for document-level binary sentiment classification still have much room for improvement.

In addition to identifying the sentimental polarities of sentences, efforts have also been made to explicitly extract the sentiment-specific word embeddings (SSWEs) [7] to further highlight the semantic meanings of the given texts. In [7], a supervised learning framework was proposed to learn the continuous word representations as features for Twitter sentiment classification. The SSWE with sentiment information was integrated into three neural networks. Their proposed model, combining syntactic context of words and sentiment information of sentences, had the best performance in their experiments.

To leverage additional information for accurate sentiment detection, emotion inclination of the sentences has also been considered. In a pilot study, a text-based emotion classifier, combining long short-term memory (LSTM) and convolutional neural network (CNN) architectures, was developed, able to distinguish seven basic human emotions in tweets [8]. The extracted word embeddings proposed in that paper were based on the concepts of users' emotion vectors and movies' emotion vectors.

With the rise of deep learning (DL) techniques, convolutional neural nets with multiple hidden layers have been popular candidates for solving challenging text mining tasks. However, deep learning models with complex architecture usually involve optimizing quite a large number (millions) of parameters, which makes the fine-tuning of such models not only challenging, but also impacts the robustness of the method across different databases. To tackle this issue, efforts have been made to develop new methods, which can drastically reduce the required parameters to fine-tune the pretrained model. Zhong et al. [9] proposed to only use a small user-specific feature vector to achieve model optimization; their method was evaluated using two datasets of user reviews for sentiment analysis. For those two self-created datasets, classification accuracies of 46.15% (10-way) and 70.22% (5-way) were achieved. It is worth noting, however, rather than pursuing high classification accuracies, their primary focus in the pilot study was seeking to reduce the number of parameters needed for optimization.

To balance the trade-off between classification performance and computational load (time) in solving challenging sentiment analysis problems, a few conventional machine learning (ML) techniques have also been found effective. Many reviewing systems employ the popular five-star ranking scheme to evaluate the customers' opinion in a segmented manner. The Yelp's Academic Review Dataset [10] and Amazon Fine Food Review [11] both use this five-star scheme for opinion ranking.

Xu et al. [12] explored the effectiveness of different ML algorithms for such a five-way classification problem using the Yelp dataset. The performances of naïve Bayes, perceptron, nearest neighbor, and multi-class SVM were tested. Naïve Bayes achieved the best overall performance for all the categories, perceptron had the highest precision and recall for one- and five-star ratings, but the predictions for two, three, and four stars were quite poor.

Their work pointed out one of the major challenges for such sentiment analysis: it is often rather difficult to distinguish the difference amongst the middle ratings (two and four stars in particular), even for humans. On the other hand, such a challenging classification scenario also makes a good test case for evaluating and comparing different ML algorithms. Indeed, while facing this challenge, researchers often tend to merge the middle ratings to form a cohort with reduced categories for better classification performance (88.91% [13] and 95% [14]).

To tackle these problems pointed out in the literature, a novel re-engineering mechanism for the generation of word embeddings is proposed in this study which could also find potential applications in other areas such as biometrics and image processing. The main contributions could be summarized as follows:

(1) A novel mechanism is proposed for re-engineering word embeddings, which boosts the semantic classification performance for sentiment analysis across four public benchmarking datasets.
(2) The proposed mechanism is shown to reduce the system training time drastically (a few seconds) compared to the mainstream DL methods (minutes to hours).
(3) Good performance is achieved using a small amount of data for training, which is particularly advantageous for certain application scenarios where the amount of annotated data is limited.

This work is organized as follows: the basic methods used for data preprocessing and extraction of preliminary embeddings are presented in Section 2. In Section 3, a novel embedding re-engineering mechanism, designed to boost the performance for semantic classifications, is presented in detail. The effectiveness of the proposed mechanism is further evaluated using a range of publicly available datasets in Section 4, including both the binary and five-way classification scenarios. Section 5 provides a comparative analysis with the state-of-the-art reports. A discussion of the proposed algorithm is provided in Section 6, conclusion and some suggestions for future work are included in Section 7.

2. Preliminary Word Embedding Generation

To leverage the full power of machine learning for sentiment analysis, selecting a technique which is able to effectively convert the text to machine-readable embedding feature vectors is of paramount importance. In this work, we choose to implement the classical Word2Vec algorithm to generate the preliminary word embeddings. Word2Vec uses a two-layer neural network to learn the word associations and then convert the tokens into numerical vectors. Each vector is designed to reflect the level of semantic similarity between the words, and such similarity could be measured directly using the cosine distance metric. Each word corresponds to a single vector, and the elements of the vector are generated so that the words with similar semantic meaning tend to be clustered together in the feature space.

The main challenge is to generate such feature vectors (word embeddings) which can effectively reflect the real semantic relationships. The basic architecture of the Word2Vec is similar to a conventional autoencoder, as it takes a large number of vectors as input which could be one-hot vectors ($1 \times N$ matrix (vector) used to distinguish each word in a vocabulary from every other word in the vocabulary), and each corresponds to a unique word. It then compresses them down to dense (fully connected) vectors with relatively small size in the hidden layer. The output layer provides the probabilities for the target words, using SoftMax as the loss function. The weights of the hidden layer are used as the word embeddings which will be used for vector re-engineering in the follow-up processing modules.

A few preprocessing steps need to be conducted before generating the word embeddings. Figure 1 and the following steps show the main processing modules of the proposed system.

2.1. Data Cleaning

The raw text data contain a lot of irrelevant symbols that are not contributing to the sentiment classification and need to be removed, such as punctuation and some spaces. In this study, these symbols were removed first. As the MATLAB Word2Vec function [15] used in this work is case sensitive by default, all the capitalized characters were also deliberately converted to be lowercase.

2.2. Normalization

A number of customer review databases are employed in this work for evaluation. In this step, each individual review was treated as an independent document for data processing. Text normalization was performed to transform each document into a canonical

(standard) form. For the simplicity, this step (lemmatization) was completed using function from MATLAB [16] to convert the words to their root forms.

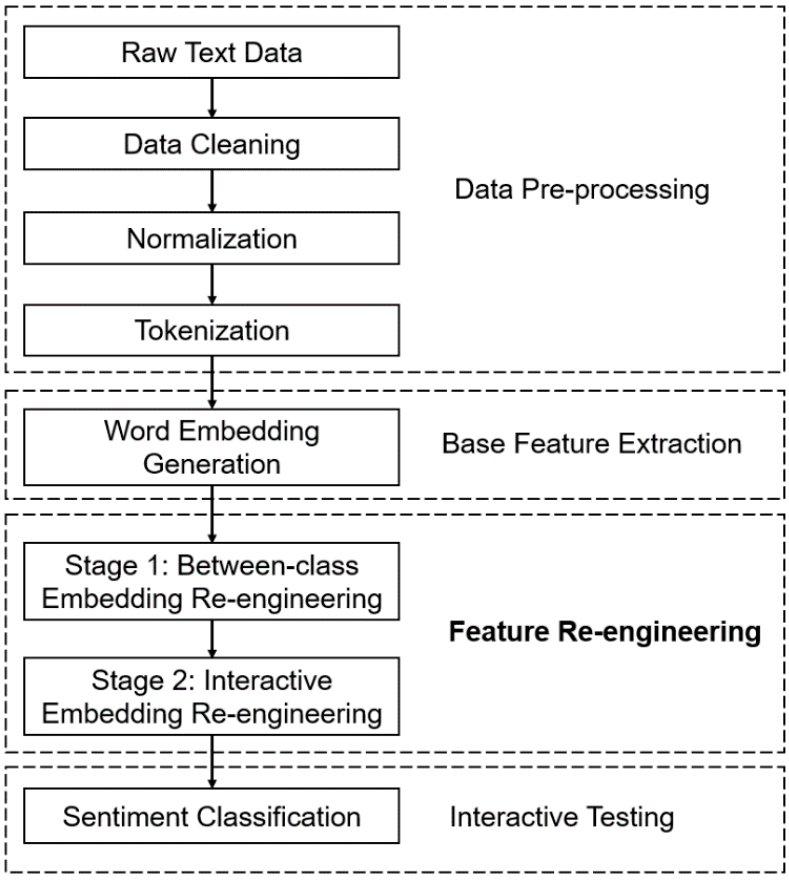

Figure 1. Flow chart diagram of the proposed classification system for sentiment analysis with four main processing modules.

2.3. Tokenization

After performing the basic noise removal and normalization for the raw text data, the remaining texts are tokenized (splitting a phrase, sentence, paragraph, or an entire text document into smaller units, each of these smaller units is called a token).

In this study, the tokenization was based on the whitespace, which was performed separately for each individual review.

2.4. Word Embedding

After the tokenization, data are ready to be vectorized for word embedding generation. It should be noted that in this study, the word embeddings were created based on the available data (i.e., without using a pretrained model), and embeddings for training and test sets were generated separately. Therefore, for the case when a word in the test set is not in the embedding vocabulary, it will return a row of NaNs. It has been found that with the proposed re-engineering mechanism, such situation did not impact the performance.

The Word2Vec algorithm employed in this work uses the continuous skip-gram model architecture to learn the word embeddings, instead of the continuous bag-of-words (CBOW). In the CBOW architecture, the model predicts the current word from a window of surrounding context words. In the continuous skip-gram architecture, in contrast, the model uses the current word to predict the surrounding window of context words. The skip-gram architecture weighs the nearby context words more heavily than more distant context words [17,18]. According to the authors' note in [19], CBOW is faster while skip-gram does a better job for infrequent words. In addition, for a medium-sized corpus (less than 10 million words),

the skip-gram model for Word2Vec tended to perform well with a relatively short feature vector (less than 100 dimensions) [20]. This makes the skip-gram model a good fit for the proposed study in this work.

A number of other important parameters need to be clarified while using the adopted Word2Vec algorithm to create the word embeddings. The dimension of word embedding has been set to 100, i.e., for each unique word, a vector with 100 dimensions will be generated. This parameter was set empirically to balance the computational complexity and performance. The size of the context window is set to 10, which has been found to be effective for the skip-gram model [20]. Other parameter settings, such as loss function, initial learning rate, and number of epochs for training are set with the default values [21].

Once the tokens have been converted into vectors using Word2Vec, ML classifiers can begin to perform the sentiment analysis. In the next section, the novel word embedding re-engineering mechanism is presented in detail.

3. Embedding Re-Engineering Mechanism

Word embeddings generated by Word2Vec provide a bridge between the human-readable text and machine-readable vectors, which allows the deployment of a wide range of ML algorithms. However, as was reviewed in Section 1, the state-of-the-art reports indicate that much improvement may still be possible in the field of sentiment analysis. In this section, a novel feature re-engineering mechanism, rooted in the fundamentals of pattern recognition, is presented for the word embedding classification tasks.

All pattern recognition problems may ultimately be reduced to similarity measurements between entities [22]. In the context of sentiment analysis, good semantic classification relies on fulfilling two principles: (1) a training set with good between-class separation, (2) high similarity level between the query text and the correct class from the training set. There are two typical paths to fulfill these principles: to train a mathematical model that is capable of classifying the samples (such as neural nets), or to directly leverage the carefully engineered samples/features for the similarity measurement (such as support vector machines). The proposed method in this work is following the second train of thought in achieving a substantial classification improvement for sentiment analysis.

3.1. Embeddings Re-Engineering

It has been found that using the resulting vectors directly after Word2Vec transform often cannot obtain the best performance. To achieve better between-class separation, an embeddings re-engineering mechanism is proposed to measure the distances between all the available between-class component pairs for each vector dimension, then reconstruct the embedding vectors with better discriminations. The following depicts the main steps of the mechanism in satisfying the 1st principle. Here the polarity (positive and negative) scenario is used as an example, and each vector has N dimensions (N = 100 in this work). Note that according to the illustration in Figure 2, each component may have multiple vector elements originate from different words. In this illustration, it is expected that the training set contains two classes, and each class includes multiple word embedding vectors after performing the Word2Vec transform. Each embedding vector has a number of components, which correspond to the number of dimensionalities of the vector.

(1) For each component of a given embedding vector from one class (positive), $C_{pos_{ii}}$, its distances to all vectors of the same component (position) from the other class (negative), C_{neg} are measured:

$$\begin{aligned} d_{11} &= \left|C_{pos_{11}} - C_{neg_{11}}\right|, \\ d_{12} &= \left|C_{pos_{11}} - C_{neg_{21}}\right|, \ldots, \\ d_{1Q} &= \left|C_{pos_{11}} - C_{neg_{Q1}}\right|. \end{aligned}$$

These distance measurements are iterated for all the components (dimensions) of the vectors between the classes.

(2) The distances between one given component and all the components from the other class are summed, which are used to indicate the separation level of that component with all the other classes as a whole:

$$D_i = \sum_{j=1}^{Q} d_{ij}.$$

(3) To maximize between-class separations, the positions of components C_{ij} are re-arranged according to the descending order of the total distance measurements, D_i (Figure 2).

(4) The resulting components of Step 3 are, therefore, reconcatenated to form re-engineered embedding vectors.

It should be noted that this re-engineering process is conducted for each component (dimension) separately, i.e., no across-dimension rearrangement of the feature component is performed. An illustrative demonstration for this stage of the mechanism is shown in Figure 2.

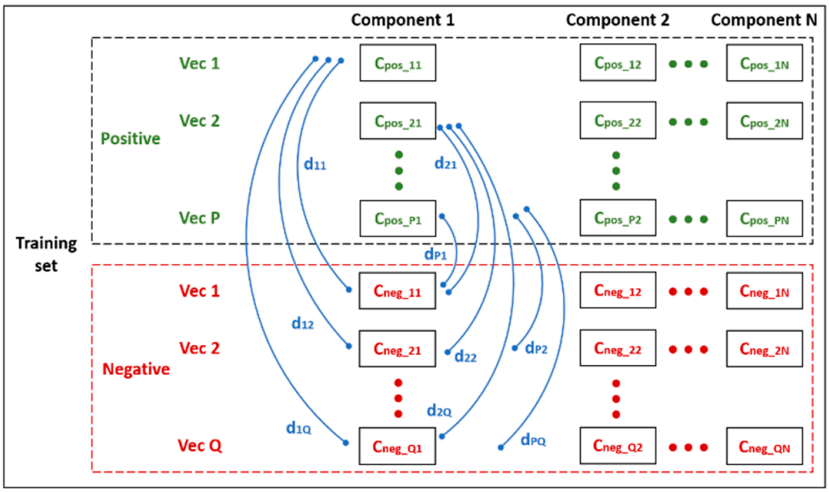

Figure 2. An illustration of the first stage of the proposed mechanism for a binary sentiment analysis scenario. The value of each component from each feature vector is compared with the values of the same component from all the feature vectors of the other classes to make the similarity measurements. The component values are then replaced in descending order of total distance measurements for each value. The new feature vectors are then formed from these reordered values.

3.2. Interactive Testing

To achieve good matches between the text query vectors and the vectors in the training categories (2nd principle), in the interactive test stage it is proposed to minimize the similarity of the two entities by creating re-engineered vectors for both the training and query set. In this stage, the resulting values between the vectors of a given training class and the query set are reordered according to the increasing distance measurements for each component. This is to rank and find the closest possible query–training matches of the set. The second stage mechanism is illustrated by Figure 3.

Similar to the embedding re-engineering stage (Section 3.1 above), the distances between a given query sample and all the training samples from one class are measured and sorted in monotonically increasing order. Hence, each query sample has one set of distance scores to represent the closest matches each class has to offer. As each query set may have multiple samples, this computation is repeated for all the query samples. For each query sample, its distance scores with respect to the training set of one class are summed to show its separation level. The smallest distance summation scores may correspond to a set of vectors originating from different embeddings. Therefore, the re-engineered vectors can be constructed based on these distance indications, which manifest the best matches

between the query sample and the given class. This process is performed between the query set and the other class, and another set of re-engineered embedding vectors as well as its corresponding re-engineered query set are generated, which in turn indicate the best matches between the query sample and the other class. The final classification could be performed by: (1) selecting the class which demonstrates the lowest separation level or (2) feeding these re-engineered embeddings to another classifier for decision making.

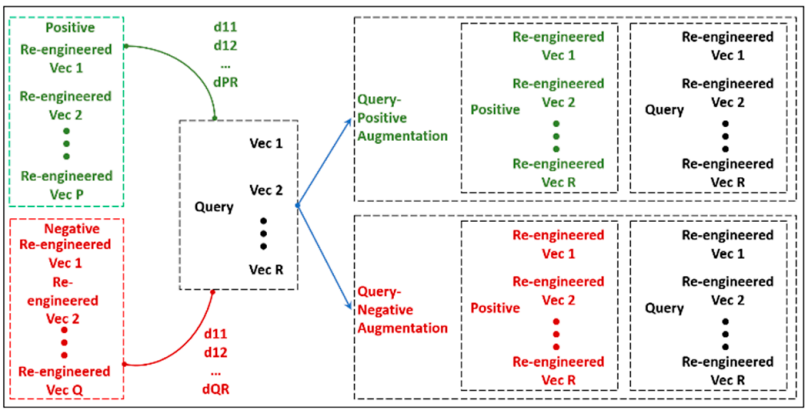

Figure 3. The interactive test stage of the proposed mechanism using a binary sentiment analysis scenario as an example. Each value per component from the vector of one class is compared with the same component from vectors in the query set for similarity measurements. The values of each component in the query set are then reordered according to ascending distance measurements (N = 100). A separately re-engineered query set is produced for each class.

One distinctive highlight of the proposed re-engineering mechanism is the augmentation of the embedding vectors in the interactive test stage. For a given query vector set, its similarity is measured against all the categories in the training set, i.e., each measurement leads to the creation of a pair of re-engineered training–query embedding sets. The query set is, therefore, dilated by K times (K indicates the number of categories in the training set). For example, for a binary classification problem shown in Figure 3 (positive and negative), when one re-engineered query set as well as its corresponding re-engineered positive set are generated, another re-engineered query set and its matching re-engineered negative set are constructed in parallel. The resulting query feature set for evaluation is doubled in size compared to the original query set.

The resulting re-engineered embedding and query set are now ready for sematic classification. Note the augmented feature vectors of the query sets are handled without fusing. The next section is devoted to exploring the effectiveness of the proposed mechanism for sentiment analysis using four publicly available benchmarking datasets.

4. Evaluation Case Studies

In this section, the proposed method is evaluated using four public benchmarking datasets to explore its effectiveness and robustness for sentiment analysis. Depending on the number of categories for classification, i.e., binary or five-way, the section is further divided into two subsections.

4.1. Binary Sematic Classification

The IMDB Dataset of 50 K Movie Reviews [23] was used to explore the impact of the proposed feature re-engineering mechanism for binary classification. This dataset contains a set of 25,000 highly polarized movie reviews for training and 25,000 for testing; additional details of this dataset may be found in [6].

In this study, the original text of each review has been preprocessed, following the four steps presented in the Section 2, i.e., data cleaning, normalization, tokenization, and

preliminary word embedding creation, resulting in the vectorized feature set with positive and negative categories.

Figure 4 shows an illustrative example of the original word feature embeddings of the training set before applying the re-engineering mechanism. It is clear that after the Word2Vec, the resulting word embeddings are still not quite separable, which demands additional feature engineering to improve the discrimination. The feature distributions are plotted from two randomly selected consecutive dimensions (in this case, 50th and 51st). For clear visualization and comparison, the figure displays only the plots of the first 100 observations per class, and the full distribution for these two dimensions is found to overlap even more between the two categories.

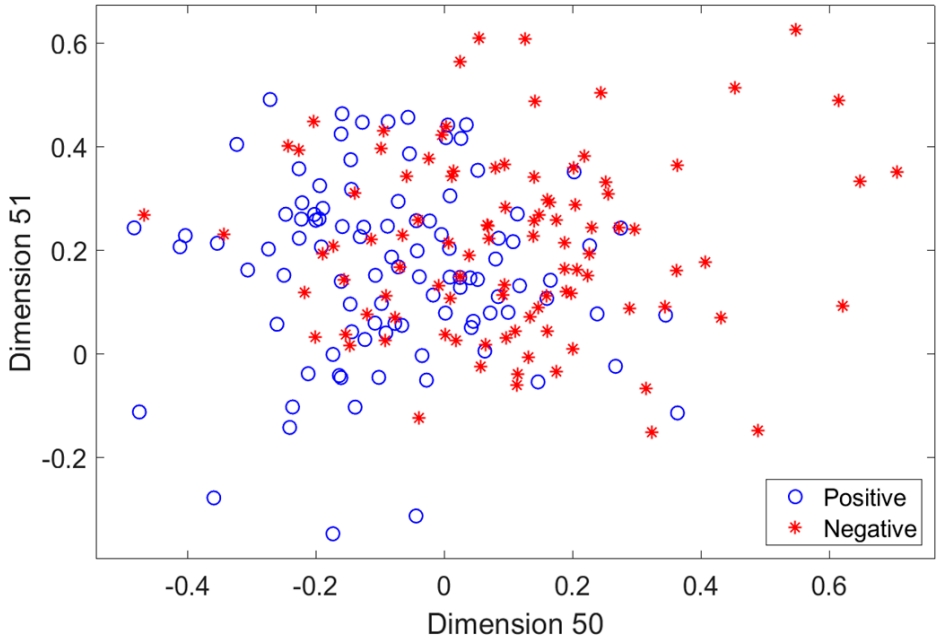

Figure 4. The original training set embedding distribution of the IMDB Dataset for two randomly selected dimensions (100 observations in total). For clear illustration, only the first 100 vectors are plotted.

Figure 5 shows the resulting distribution after the embeddings re-engineering. Thanks to the proposed mechanism, the re-engineered feature vectors have been found with much better separated distribution compared to the original embeddings (Figure 4). The re-engineered distribution can be easily separated using a 3-nearest neighbour classifier successfully. Note that as the between-class distances of the re-engineered embeddings are ranked in a monotonically decreasing order, the embeddings with shorter distances may be removed from the training set to enhance the between-way separation.

The interactive test is similar to the first stage (between-class separation) in its operation, but it is re-engineering the training and query vectors in monotonically increasing order of the distance measurements to find the closest training–query matches possible.

Table 1 provides a comparison of performance with and without using the proposed embedding feature re-engineering mechanism for the IMDB Dataset. A range of classifiers have been tested, including k-NN, logistic linear, support vector machine, linear discriminant analysis, and feed forward neural networks with backpropagation. The best performing algorithm was found to be the naïve Bayes classifier. In line with the established practice in the literature, the precision, recall, and F_1-score have been reported as metrics for classification performance evaluation.

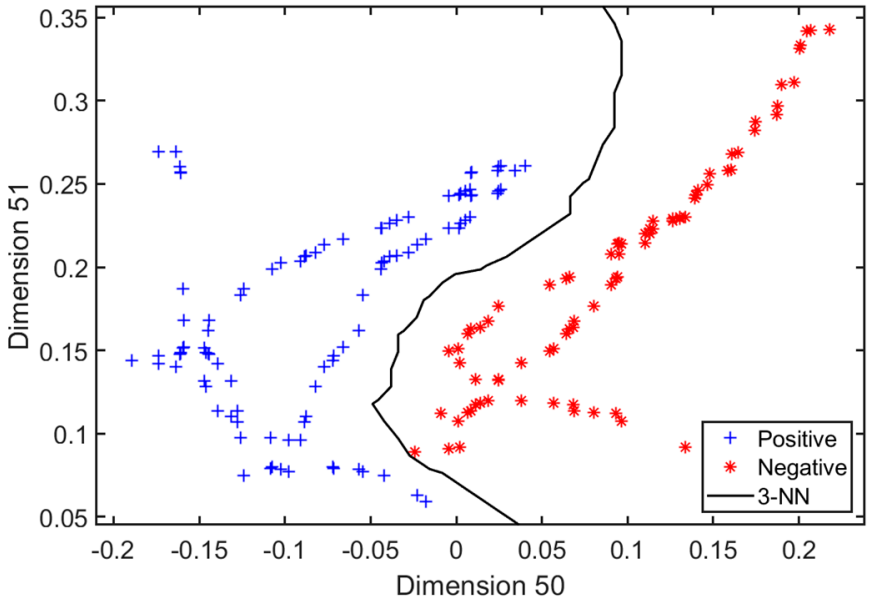

Figure 5. The resulting re-engineered embeddings of the samples from IMDB Dataset. Compared with Figure 3, the distribution is found to be much less overlapping. As an example, using a 3-NN classifier can divide the two categories directly.

Table 1. Results comparison using the IMDB Dataset. The initial classification results using naïve Bayes classifier, as well as the results after applying the re-engineering mechanism. These results are averaged from 100 random tests. The boost of the performance has been found substantial.

Metric	Naïve Bayes	Re-Engineering
Precision	81.29%	**97.50%**
Recall	70.10%	**97.50%**
F_1-score	75.28%	**97.40%**

With each query set comprising 15 vectors (tokens) from a given category, the average results (of 100 random trials) show a substantial improvement when the re-engineering mechanism is used in addition to the employed classifier. These results also seem to be consistent with the distributions in Figures 4 and 5. It may be worth pointing out that the results in the second column of Table 1 are obtained by using only 100 random embedding vectors for training, hence the training time (about 0.5 s, using 11th generation Intel® Core™ i7 eight-core processor) is rather short compared with using the deep learning techniques. This was achieved by leveraging the proposed mechanism to significantly improve the quality of the training set, as well as enhancing the training–query matches.

To further demonstrate the robustness of the proposed method across different datasets for binary sentiment analysis, another popular dataset was also explored in this work, namely the Sentiment140 dataset with 1.6 million tweets [24]. Tweet records in the public version of this dataset have been annotated into two polarized categories (negative and positive). The additional information on its basic statistics may be found in [24]. Using the proposed method, the averaged F_1-score of 0.9867 was achieved for this dataset, along with precision and recall of 98.50% and 99.00%, respectively. A comparison of the results with the state-of-the-art reports in the literature will be presented in the next section.

4.2. Five-Way Sentiment Analysis

The proposed method has also been tested in a five-way scenario for more challenging sentiment analysis. First, the Yelp academic reviews dataset [10] is adopted for the evaluation, the contents of which have been divided using the 5-star ranking system. It contains data related to businesses, reviews, and user data that have been made publicly

available for personal, educational, and academic purposes, and this dataset contains 6,685,900 reviews in total [25].

The preprocessing of the dataset follows the same steps that were described in Section 2, resulting a set of 100-dimension vectors for the subsequent feature re-engineering. Similar to the binary classification scenarios discussed in Section 4.1, a high average F_1-score of 94.39% was obtained using a small subset (60 embeddings) of the Yelp dataset. In addition, a series of tests have been conducted to explore the effectiveness of the method while further increasing the training set size, and the results of multiple tests are shown in Figure 6. The number of vectors in the training set was increased from 50 to 100 in steps of 10 vectors, while the test set contained 10 vectors (default setting). Each boxplot was generated with 100 trials, randomly sampled across the whole dataset.

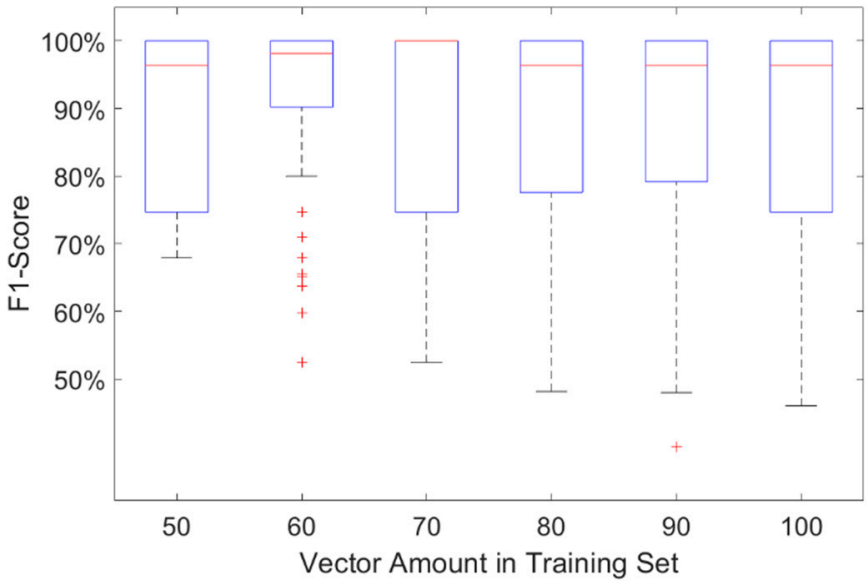

Figure 6. The effectiveness of the proposed mechanism in F_1-score boxplots, tested using the Yelp dataset. The size of the training set is increased from 50 to 100 vectors, while the query set retained only 15 vectors. The best performance is achieved when the training set contains 60 re-engineered vectors, the performance starts to degrade with the addition of training vectors.

The best performance is found when the training set contains only 60 (re-engineered) vectors, with both the highest F_1-score as well as the smallest variance (2nd column of Figure 6). With the size of the training set increasing, the mean performance starts to degrade in terms of both metrics, also inevitably increasing the computational time.

This somewhat counter-intuitive phenomenon stems from the very mechanism proposed in this study. Since the re-engineering mechanism selects and keeps a set of feature component candidates based on the distance measurement, as the size of training set increases, it will likely preserve and generate more irrelevant feature vectors to construct the re-engineered embeddings. For example, embeddings created using similar or the same tokens that appear in the reviews from different categories. After all, the number of keywords in a sentence that indicate its sentiment are usually much fewer than other general words. This potentially makes the addition of data not only useless, but even polluting to the representativeness of the training model. Indeed, if only the useful tokens could be added for training, the performance would improve in proportion to the training data augmentation. Such a limitation, however, may not be easy to overcome since the knowledge of effective keyword tokens is not available in advance.

To explore the robustness of the method, another popular five-way benchmarking dataset, the Amazon Fine Food Reviews dataset [11], was also used for evaluation. This dataset also follows the five-star rating system to signify the sentiment level of each review. Using a similar experiment design and parameter settings, an average F_1-score of 96.47%

was reached using 100 trials and a subset of 80 embedding vectors were randomly selected for training. In the next section, a set of comprehensive comparisons with related works in the literature are presented for each adopted dataset in this study.

5. Comparative Analysis

In this section, we conduct a comparative evaluation of the proposed method with some related techniques. Representative results are listed in Table 2 (the results which were obtained using the proposed method are highlighted in bold). Discussions are also provided to point out the pros and cons of the related works, as well as emphasize the particular characteristics of the proposed system.

In order to boost the classification performance for sentiment analysis, one intuitive strategy could be combining multiple approaches and fusing the results. Indeed, such a heterogeneous stacking ensemble framework was proposed and tested using a range of databases, and some promising results were reported in [26]. In their work, a series of popular word embedding techniques have been used to extract the features of the text separately; namely, Word2Vec [17], GloVe [27], and BERT [28]. The resulting embeddings were subsequently fed into an ensemble of classifiers, including long short-term memory (LSTM) networks [29], gated recurrent unit (GRU) [30], and their variants. F_1-score performances reported for Sentiment140 [24] and IMDB Review datasets are 0.8435 and 0.9222, respectively (Table 2).

Table 2. Comparative analysis of the proposed method (shown in bold) and a few recent reports using the four adopted datasets. Note that some results were achieved using combined categories.

Dataset/Metric		Precision	Recall	F_1-Score
2-way	IMDB Reviews	91.65% [26] **97.50%**	92.79% [26] **97.50%**	0.9222 [26] **0.9740**
	Sentiment140	85.39% [26] **98.50%**	83.33% [26] **99.00%**	0.8435 [26] **0.9867**
5-way	Amazon Reviews	**96.53%**	**96.46%**	0.8024 [31] **0.9647**
	Yelp Reviews	5-way **93.93%**	5-way **94.96%**	0.7570 (3-way) [32] **0.9439**

In [26], a computational overhead analysis was also reported, which indicated the existence of a trade-off between the classification performance and computational time. It is worth noting, however, that this system required each embedding vector to have 200+ dimensions in order to achieve the reported performance, which took about 40 to 60 min for model training for IMDB Reviews and Sentiment140, respectively. Compared to the proposed system in this work which generated vectors with lengths of less than 100 dimensions, the training and test in total took only a few seconds.

A recent advance towards solving the five-star rating classification problem was reported by Dang et al. [31], in which a chain of complex deep nets were employed to discriminate between the classes of the five-way database. They proposed to firstly use a pretrained BERT model for the feature vector extraction, then connected it with an LSTM network, further followed by a CNN. The ReLU activation function was used in the final stage of the model. Compared to Word2Vec, BERT includes some particular advances over Word2Vec, for example, it supports out-of-vocabulary words. The Amazon Fine Food Review dataset was used to test their algorithm and the F_1-score of about 80% was reported [31] (Table 2). Such performance, however, was achieved using a model with a rather convoluted architecture, and the optimization of the model parameters would be costly [9]. Compared to the proposed method, no pretrained model is needed after the embedding re-engineering.

The 5-way classification is considered a more challenging classification problem and its intermediate ranks are sometimes combined to form more distinctive categories. For example, the results reported in [32] (Table 2) were achieved by grouping ratings 1 and 2 as "negative" sentiments, ratings 4 and 5 as "positive" sentiments, and the rating 3 as "neutral", hence simplifying the classification problem. The bag-of-words and term frequency–inverse document frequency (TF-IDF [32]) techniques were used to construct the word representations. Compared to deep learning methods, an F_1-score of 0.757 was achieved by using a support vector machine, taking only 11 s for the model training [32]. This trade-off between accuracy and training time is also in line with our observation in this study: both the classification performance and the computational load are important factors to consider in developing effective systems for sentiment analysis.

Results of the five-way sentimental analysis in this work were obtained using the logistic linear classifier working together with the re-engineered feature embeddings, while the binary sematic classifications used the 3-NN algorithm. The fact that many state-of-the-art reports simplified the five-way classification into three- or two-way problems [13,14,32] seems to indicate that five-way sentiment analysis is still a rather challenging classification problem. By actively re-engineering the embedding vectors, the proposed method has demonstrated a significant leap in performance for document-level sentiment analysis using small subsets of the databases.

6. Discussion

A novel mechanism for re-engineering the word embeddings has been proposed in this paper, and its effectiveness and robustness are demonstrated using four public benchmarking datasets. Analyses for two-way, as well as five-way, semantic classifications were conducted, and substantial improvements were observed compared to the published results in the literature.

In this section, a conceptual comparison between the proposed mechanism and mainstream neural network (NN) methods is conducted. The re-engineering mechanism as a generic framework is further explored to highlight its characteristics which differentiate it from NN-related methods. Main components of the proposed system and conventional NN systems are illustrated in Figure 7.

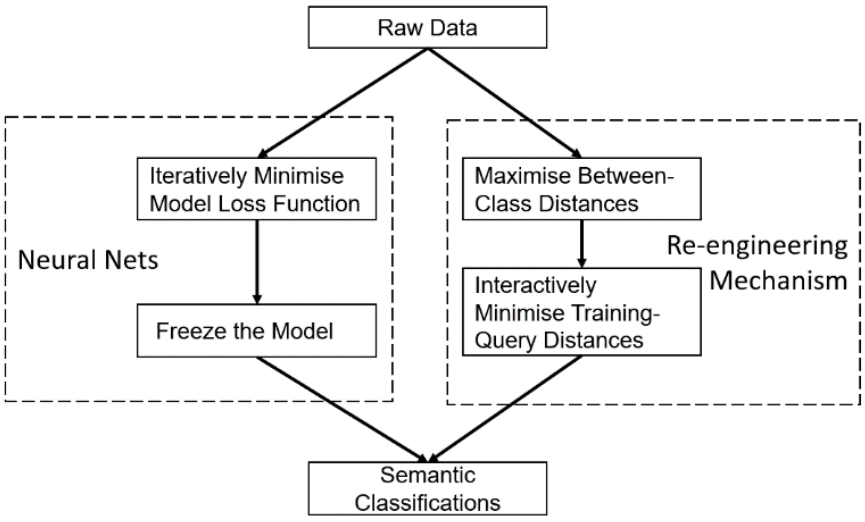

Figure 7. System flowchart diagram of the comparison between the neural nets and the proposed re-engineering framework for embedding feature generation. The system adaptively re-engineers the word embeddings for each given query sample to provide the best possible match, as opposed to fitting different query samples with a static model.

The core recognition module of typical neural nets is the loss function of the model; as the iterative training of the model progresses, the measured loss is minimized. Such

algorithms usually have three major drawbacks: (1) the training process can be quite time consuming, especially for challenging classification tasks when the model must be trained using a large database. (2) These methods are based on model parameter optimization, which is inherently subject to overfitting if there are insufficient training data. (3) For some application scenarios, the robustness of the model across different databases may suffer great degradation without careful fine-tuning. Indeed, to tackle these problems, some mechanisms based on pretrained models, such as transfer learning and reinforcement learning, were developed, hoping to better adapt specific implementation scenarios with minimal parameter adjustment (retraining). These approaches, however, may not reduce the overall training time; in addition, as the pretrained model was not specifically designed to fit for the problem at hand, retraining may not guarantee a high level of performance.

The proposed feature re-engineering mechanism, benefiting from its instance-based data-driven design, is able to adaptively optimize the between-class separation of the training set, while achieving the best possible match between the training embeddings and the test samples. Compared to the general methods based on deep nets, this recreation of the word embeddings results in a good performance using only a small amount of data with much faster training.

To better clarify and highlight the novelty of the proposed method, it is worth pointing out three distinctive characteristics of this re-engineering mechanism, i.e., component-wise vector reconstruction, transformation irreversibility, and the analysis efficiency.

6.1. Vector Reconstruction

The major difference between the proposed method and other feature/instance selection methods lies in the idea of vector reconstruction. The re-engineering mechanism measures and ranks the similarity between the vectors for each component (dimension), resulting in a series of newly reconstructed vector embeddings. It does not passively select the vectors, but actively re-engineers new vectors using the existing components of the word embeddings. In addition, to boost the performance, as well as reduce the training duration, often only a subset of these ranked and re-engineered vectors may be preserved for the sentiment classification. The optimal number of retained vectors depends on the quality of the training dataset.

6.2. Transform Irreversibility

Another feature of the proposed method is that this re-engineering process for word embeddings is irreversible. Conventionally, transforms can be operated bidirectionally between word token and vector, which allows a one-to-one match for a given word and its vector. The proposed method, on the other hand, leverages the availability of a set of embeddings as a whole from this pool of vectors to generate a new set of embeddings. The components of each re-engineered embedding vector may stem from different original embedding vectors; therefore, it is impossible to inverse the process and find the original token. However, the document-level sentiment analysis is almost always dealing with sentences, each sentence usually corresponding to a set of word embeddings. From this perspective, the proposed mechanism considers a sentence or a set of sentences as a whole for sematic analysis, instead of looking at individual key tokens.

6.3. Analysis Efficiency

Closely related to the previous two highlights, the proposed method significantly benefits from the mechanism of feature reconstruction to efficiently leverage the available data. As was indicated in Section 4, the proposed method can achieve superior performance using a small amount of data, by re-engineering the embedding feature vectors. Therefore, the computational load is reduced drastically and, in the meantime, maintained a high classification performance for the sentiment analysis. This could be an appealing characteristic for the scenarios where the annotated data may be insufficient for deep learning approaches.

7. Conclusions and Future Work

This work presents an approach to sentiment analysis using a novel mechanism for word embeddings. Four popular datasets were used to analyze its performance in both two-way and five-way classifications. The proposed method demonstrates substantial improvements across these datasets when compared to previous reports, indicating its effectiveness and robustness for sentiment analysis. One potential limitation of the proposed method is the need for exhaustive calculation of the distances between the embedding elements which may result in a significant computational load and speed limitations. However, this may be alleviated by not using all the available instances and instead sampling the embedding instances, hence reducing the number of candidates for the distance measurement. The implication of such a strategy for performance and computation load would a be subject for further work.

The performance achieved using the proposed method in this study was based on the conventional Euclidean distance for similarity measurements, and further work may also be directed towards exploring other distance measures that may better capture the distinctiveness of the word embeddings. Another direction for future work is to explore the application of the proposed method to other data types, such as image and time series analysis.

Author Contributions: Conceptualization, S.Y.; Formal analysis, S.Y.; Investigation, S.Y.; Methodology, S.Y.; Project administration, F.D.; Software, S.Y.; Supervision, F.D.; Writing—original draft, S.Y.; Writing—review & editing, F.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data supporting this study are openly available from Kaggle at the the following datasets: IMDB Dataset of 50K Movie Reviews [6]; Sentiment140 dataset with 1.6 million tweets; Yelp Dataset [10]; Amazon Fine Food Reviews [11]; Sentiment140 dataset with 1.6 million tweets [24].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Feldman, R. Techniques and applications for sentiment analysis. *Commun. ACM* **2013**, *56*, 82–89. [CrossRef]
2. Doan, T.; Kalita, J. Sentiment analysis of restaurant reviews on yelp with incremental learning. In Proceedings of the 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, USA, 18–20 December 2016; pp. 697–700.
3. Shivaprasad, T.K.; Shetty, J. Sentiment analysis of product reviews: A review. In Proceedings of the 2017 International conference on inventive communication and computational technologies (ICICCT), Coimbatore, India, 10–11 March 2017; pp. 298–303.
4. Vashishtha, S.; Susan, S. Highlighting keyphrases using senti-scoring and fuzzy entropy for unsupervised sentiment analysis. *Expert Syst. Appl.* **2021**, *169*, 114323. [CrossRef]
5. Pang, B.; Lee, L. Dataset Movie Reviews | Kaggle. Available online: https://www.kaggle.com/nltkdata/movie-review (accessed on 8 January 2022).
6. IMDB Dataset of 50K Movie Reviews | Kaggle. Available online: https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews (accessed on 8 January 2022).
7. Leung, J.K.; Griva, I.; Kennedy, W.G. Text-based Emotion Aware Recommender. In Proceedings of the Computer Science & Information Technology (CS & IT), Zurich, Switzerland, 21–22 November 2020; pp. 101–114.
8. Tang, D.; Wei, F.; Yang, N.; Zhou, M.; Liu, T.; Qin, B. Learning Sentiment-Specific Word Embedding. In Proceedings of the ACL, Baltimore, MD, USA, 22–27 June 2014; pp. 1555–1565.
9. Zhong, W.; Tang, D.; Wang, J.; Yin, J.; Duan, N. UserAdapter: Few-Shot User Learning in Sentiment Analysis. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online Event, 1–6 August 2021; pp. 1484–1488.
10. Yelp Dataset | Kaggle. Available online: https://www.kaggle.com/yelp-dataset/yelp-dataset?select=yelp_academic_dataset_review.json (accessed on 8 January 2022).

11. Amazon Fine Food Reviews | Kaggle. Available online: https://www.kaggle.com/snap/amazon-fine-food-reviews (accessed on 8 January 2022).
12. Xu, Y.; Wu, X.; Wang, Q. Sentiment Analysis of Yelp's Ratings Based on Text Reviews. In Proceedings of the IEEE 2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Timisoara, Romania, 21–24 September 2015.
13. Yu, B.; Zhou, J.; Zhang, Y.; Cao, Y. Identifying Restaurant Features via Sentiment Analysis on Yelp Reviews. 2017, pp. 1–6. [Online]. Available online: http://arxiv.org/abs/1709.08698 (accessed on 8 January 2022).
14. Kazmaier, J.; van Vuuren, J.H. The power of ensemble learning in sentiment analysis. *Expert Syst. Appl.* **2022**, *187*, 115819. [CrossRef]
15. Map Word to Embedding Vector—MATLAB word2vec—MathWorks United Kingdom. Available online: https://uk.mathworks.com/help/textanalytics/ref/wordembedding.word2vec.html (accessed on 9 January 2022).
16. Stem or Lemmatize Words—MATLAB normalizeWords. Available online: https://uk.mathworks.com/help/textanalytics/ref/normalizewords.html (accessed on 9 January 2022).
17. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. In Proceedings of the 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, AZ, USA, 2–4 May 2013; pp. 1–12.
18. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Stateline, NV, USA, 5–10 December 2013; pp. 3111–3119. [CrossRef]
19. Google Code Archive—Long-Term Storage for Google Code Project Hosting. Available online: https://code.google.com/archive/p/word2vec/ (accessed on 9 January 2022).
20. Altszyler, E.; Ribeiro, S.; Sigman, M.; Slezak, D.F. The interpretation of dream meaning: Resolving ambiguity using Latent Semantic Analysis in a small corpus of text. *Conscious. Cogn.* **2017**, *56*, 178–187. [CrossRef] [PubMed]
21. Train Word Embedding—MATLAB trainWordEmbedding—MathWorks United Kingdom. Available online: https://uk.mathworks.com/help/textanalytics/ref/trainwordembedding.html (accessed on 9 January 2022).
22. Pedro, D. A Few Useful Things to Know About Machine Learning. *Commun. ACM* **2012**, *55*, 78–87. Available online: https://dl.acm.org/citation.cfm?id=2347755 (accessed on 8 January 2022).
23. Maas, A.; Daly, R.E.; Pham, P.T.; Huang, D.; Ng, A.Y.; Potts, C. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 142–150.
24. Sentiment140 Dataset with 1.6 Million Tweets | Kaggle. Available online: https://www.kaggle.com/kazanova/sentiment140 (accessed on 8 January 2022).
25. Yelp Dataset. Available online: https://www.yelp.com/dataset (accessed on 14 January 2022).
26. Subba, B.; Kumari, S. A heterogeneous stacking ensemble based sentiment analysis framework using multiple word embeddings. *Comput. Intell.* **2021**, *38*, 530–559. [CrossRef]
27. 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, 26–28 October 2014; pp. 1532–1543.
28. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, pp. 4171–4186.
29. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
30. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
31. Dang, C.N.; Moreno-García, M.N.; Prieta, F.D.L. An approach to integrating sentiment analysis into recommender systems. *Sensors* **2021**, *21*, 5666. [CrossRef] [PubMed]
32. Liu, S. Sentiment Analysis of Yelp Reviews: A Comparison of Techniques and Models. *arXiv* **2020**, arXiv:2004.13851.