Expert Systems With Applications 210 (2022) 118606

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Social media analysis by innovative hybrid algorithms with label propagation

Ayşe Berna Altınel

Department of Computer Engineering, Faculty of Technology, Marmara University, Istanbul, Turkey

ARTICLE INFO

Keywords:
Label propagation algorithm
Social media analysis
Topic-based tweet classification
Sentiment polarity detection

ABSTRACT

Due to the huge size of the data accumulated on microblogging sites, recently, two fundamental questions have become very popular: 1) What percentage of this accumulated data has positive or negative sentiment polarity? 2) How is the distribution of this accumulated data on different topics? Inspired by these motivated necessities, this paper presents several different algorithms which are based on the Label Propagation Algorithm (LPA) in order to handle previously mentioned two fundamentals tasks: sentiment polarity detection task and topic-based text classification task. These algorithms are the Label Propagated- Relevance Frequency Classifier (LP-RFC) and LP-Abstract Frequency Classifier (LP-AFC). These algorithms can be defined as new semantic smoothing classifiers, which take advantage of the semantic connections among terms in the label propagation phase of the LPA. Additionally, another classifier, namely LP-Com$_{RFC+AFC}$, was built. LP-Com$_{RFC+AFC}$ is actually a weighted summation classifier of the individual LP-RFC and LP-AFC. Furthermore, considering the shortage of labeled data in real-world scenarios, a semi-supervised version of LP-RFC and LP-AFC, namely "Merging Unlabeled and Labeled Instances with Semantic Values of Terms" (MULIS), was designed and implemented. For the experiments of the sentiment polarity detection task, three different datasets were use and for the experiments of topic-based text classification task, a self-collected tweet dataset was use. According to the experimental results, the suggested algorithms, and their composite form, LP-Com$_{RFC+AFC}$, generated higher F1 scores than all of the baseline algorithms at nearly all of the training splits on the datasets.

1. Introduction

Microblogging sites are environments in which people are able to document their feelings, ideas, and comments about anything in life. Twitter is one of the most famous among these microblogging sites. At the end of 2020, Twitter announced that there were 321 million active Twitter users monthly. According to Twitter statistics reported, in Turkey, 7 million tweets were posted per day and 59 % of these tweets had positive sentiment polarity and the rest had negative sentiment polarity in 2021. Social media has become an indispensable part of people's daily lives. Shopping, watching news, reading reviews, communicating with other people, and learning something new are just a few of the regular activities that people take part in on social media. Such intense use of social media has brought about some difficulties as well as conveniences. The ease of accessing daily activities and services with one click has certainly brought about a great deal of comfort to people's lives.

On the other hand, this heavy usage has accumulated very big data on web platforms. Due to these accumulated big data, social media analysis is today's one of the important challenges. Consequently, a lot of important questions have derived from these big data: How can we process this big data? What percentage of this accumulated data has positive or negative sentiment polarity? What is the distribution of this accumulated data on different topics such as politics, sports, health, etc.? How can we summarize these data? How can we translate these data into another language? How can we extract meaningful data from these messy data?, and so on. Not only academic platforms but also commercial companies care about the answers to these questions. Inspired by these motivated requirements, this paper presents several different algorithms for two fundamentals tasks: sentiment polarity detection task and topic-based text classification task.

Sentiment Analysis (SA) is a mining variant of a text. It is associated with context mining. It determines, defines, and extracts the properties of it. Natural Language Processing is used for automating SA. SA is used for classifying the polarity of a text as positive, negative, or neutral. It has many use areas, such as marketing, information, and social networks. SA has gained popularity with the increasing usage of social networks. Social networks provide significant sources of data to use in

E-mail addresses: berna.altinel@marmara.edu.tr, berna.altinel@gmail.com.

https://doi.org/10.1016/j.eswa.2022.118606
Received 3 February 2021; Received in revised form 5 August 2022; Accepted 15 August 2022
Available online 18 August 2022

SA studies. SA has four main categories (Altınel & Gümüşçekiçci, 2022a, 2022b; Girgin, 2021). These are Dictionary-based SA, Corpus-based SA, Machine Learning-based SA, and lastly, Hybrid-based SA. SA does not depend on language. For this reason, different languages have different difficulties. SA has different performance levels. These are the aspect level, document level, sentence level, and word level (Altınel & Gümüşçekiçci, 2022a).

In text classification, the meanings of words and the semantic relations between terms and texts are crucial. Another critical issue in text classification is the scarcity of labeled data and the massive quantities of unlabeled data. Semi-supervised learning (SSL) algorithms attempt to make use of such excessive unlabeled data to increase the estimation ability of the classifier, especially when there are inadequate labeled data.

In semantic text classification methods, semantic values of words and semantic connections between words are considered while modeling the classifier. This property of the semantic text classification approach makes it superior to the traditional text classification approach, used to demonstrate terms in a vector space model while isolating those terms from their closer and farther contexts. Several semantic approaches were proposed in the literature within studies that analyzed their advantages over the traditional methods (Altınel & Gümüşçekiçci, 2022a, Altınel & Ganiz, 2018).

These semantic techniques can be grouped into three main types: domain knowledge-based (ontology-based) methods, corpus-based (statistical-based) methods, and deep learning methods (Altınel & Gümüşçekiçci, 2022a).

Recently, deep learning-based algorithms have become very popular and, accordingly, new neural network methods have been developed. One such method is called the paragraph vector model (doc2vec) (Altınel & Ganiz, 2018). For example, Bilgin and Şentürk (2017) performed SA using doc2vec on Turkish and English Twitter datasets. Only the distributed bag of words (DBOW) and distributed memory (DM) algorithms were compared, and it was shown that the DBOW algorithm generated better results than the DM algorithm (Bilgin & Şentürk, 2017). In another recent study performed on Turkish datasets (Şahin, 2017), the usage of word2vec vectors was compared with the BOW text representation technique. It was shown that word2vec vectors produced higher classification accuracy than tf-idf (Şahin, 2017).

Inspired by the many advantages of semantic classifiers over traditional classifiers and the advantages of corpus-based (statistical-based) classifiers, two fresh semantic supervised classifiers, namely the Label Propagated- Relevance Frequency Classifier (LP-RFC) and LP-Abstract Frequency Classifier (LP-AFC), were suggested in this study. These algorithms depend on both relevance and weighting calculations in the context of classes. Moreover, the semantic information gathered from those statistical calculations was input into the classification phase of the LP. Moreover, a composition form of RFC and AFC over LPA was built. This algorithm is called LP-$\text{Com}_{\text{RFC+AFC}}$ and it is a weighted summation classifier of the individual LP-RFC and LP-AFC with a positive real number $(0 < \lambda < 1)$. Different λ values (i.e., 0, 0.25, 0.5, 0.75, 1) were used so that actually-five variants of the LP-$\text{Com}_{\text{RFC+AFC}}$ algorithms were developed.

Additionally, one semantic semi-supervised classifier, namely "Merging Unlabeled and Labeled Instances with Semantic Values of Terms" (MULIS), was also implemented. First, it builds a classifier, which includes the relevance values (Lan et al., 2009) of terms in the training corpus and directly indicates the semantic links between words. It then tries to classify unlabeled samples based on this classifier and move them into the originally labeled set. In the second part of the algorithm, another semantic classifier, which is based on class-based abstract features, is applied. Furthermore, four variants of MULIS, namely $\text{MULIS}_{\text{RFC-AFC}}$, $\text{MULIS}_{\text{RFC-RFC}}$, $\text{MULIS}_{\text{AFC-RFC}}$, and $\text{MULIS}_{\text{AFC-AFC}}$, were also developed. The relevance value calculation was applied as a supervised term weighting methodology in the text classification field (Lan et al., 2009). Class-based term weighting computation is based on abstract features and utilized for several problems of the text classification domain, such as feature extraction and information retrieval (Bilgin & Şentürk, 2017, Biricik et al., 2009). However, to the best of our knowledge, LP-$\text{Com}_{\text{RFC+AFC}}$ is the first attempt at using both relevance and weighting computation in a supervised way. Moreover, MULIS is the first attempt at using both relevance and weighting computation in a semi-supervised way.

Herein, it was intended to compare the suggested algorithms with existing state-of-the-art machine learning and deep learning algorithms. In this perspective, linear kernel, RBF, and NB were selected as machine learning algorithms, and word2vec-skip gram (SG) with linear kernel and word2vec (CBOW) with linear kernel and Long Short Term Memory (LSTM) were selected as deep learning algorithms. The reason for selecting these algorithms as baseline algorithms was that they are very popular in the classification domain and were commonly used in recent studies, especially in SemEval tasks (Pontiki et al., 2016). To the best of our knowledge, this is the first study to use relevance values and abstract values of terms in the LP algorithm in a semantic classification setting. For the experiments of the sentiment polarity detection task, the Twitter-3 k dataset, Movie-Review-19 k dataset, and Movie-Review-100 k dataset were used. For the experiments of the topic-based text classification task, a self-collected tweet dataset was used, which had approximately 23 million tweets gathered between February 1st and March 15th, 2021. Using these datasets, the following questions were attempted to be answered in this work: 1) Do the suggested LP-based semantic classifiers generate higher classification performances than the baseline algorithms? 2) When is one superior to another? 3) Is their combination superior to either one alone? 4) Is their semi-supervised setting better than the corresponding baselines (i.e., LP, linear kernel, NB, word2vec)? 5) Do the answers to these questions depend on the size of the training set?

The main contributions of this work are as follows:

- Several different supervised semantic classifiers were developed. In this way, it was possible to see the effect of the usage of class-based semantic values of words in discriminating classes in the text classification field. In addition, to the best of our knowledge, this was the first effort to use relevance values and abstract values of terms in the label propagation phase of the LPA in a semantic classification setting.
- Additionally, several semi-supervised semantic classifiers utilizing LP-AFC and LP-RFC were implemented. In this way, it was possible to take advantage of unlabeled data in the classification process by directly applying LP-AFC and LP-RFC. To the best of our knowledge, this was the first effort to use relevance values and abstract values of terms in a semantic classification within semi-supervised setting. In other words, LP-RFC and LP-AFC algorithms can be defined as new semantic smoothing classifiers, which take advantage of the semantic connections among terms in the label propagation phase of the LPA.
- None of the suggested algorithms use words solely based on their frequencies in a vector space model. Instead, they extract class-specific words by calculating class weights (Bilgin & Şentürk, 2017; Biricik et al., 2009; Lan et al., 2009) for each of them and incorporate this information directly into the classification process, which consequently improves the classification performance.
- Furthermore, this suggested semi-supervised algorithm, MULIS, seemed to produce better classification accuracy than the SSL-linear and linear kernel algorithms.
- Also compared were these suggested supervised algorithms with two of the existing algorithms (Kilimci, 2020; Ozyurt & Akcayol, 2021) in the literature and also the submissions of SemEval-2016 Slot-3 (Pontiki et al., 2016). All of these experiments demonstrated that the proposed semantic algorithms can be configured to achieve higher F1 scores and thus, can be considered appropriate for the text classification task and sentiment polarity task.

The remainder of the paper is organized as follows: The following section provides background including the LPA, binary independent term relevance weights and sentiment polarity detection algorithms. Section 3 then presents the proposed methodologies in detail. The datasets, experimental setup, and baseline algorithms are given in Section 4. Section 5 reports the experimental results and the corresponding discussions on these experimental results. The final section presents future work and conclusions drawn from this study.

2. Related work

2.1. Label propagation algorithm (LPA)

LPA is a semi-supervised learning algorithm first proposed by Zhu and Ghahramani (2002). The LPA uses a graph based transductive approach. The LPA is a fast and efficient graph-based algorithm for labeling data. This algorithm labels the data considering the distribution of the labels in the network. When labeling the desired data, the LPA makes two assumptions about the data: According to the LPA, if two data points are close to each other in the network, these data points are very likely to have the same label or if two data points are in the same cluster on the network, these data points are very likely to have the same label.

Fig. 1 shows how the LPA labels data in a sample network. Thus, since label propagation is a graph based algorithm, it can be used to detect communities in networks. The LPA was first used by Raghavan et al. (2007), to detect communities in their proposed approach. Over the years, the LPA has become a popular algorithm to detect communities in networks. In this study, a graph-based LPA was used, which was introduced by Zhu and Ghahramani (2002).

2.2. Binary independent term relevance weights

Robertson and Sparck Jones (1976) presented the binary independence classifier. The binary independence model is known as a two-class, binary feature naïve Bayes (NB) model. The term k's relevance value is calculated as follows:

$$w_k = log_e\left[\frac{p_k/(1-p_k)}{u_k/(1-u_k)}\right] \quad (1)$$

Here, p_k denotes the probability that the word k appears in a relevant text, u_k shows the probability of word k appearing in a non-relevant text, and w_k presents the relevance weight of term k. According to discussions and analysis in the literature (Shaw Jr, 1995), if term k occurs frequently in relevant texts while it rarely occurs in non-relevant texts, then this means that term k has the capability of discriminating relevant texts from non-relevant texts, which is called the distinguishing characteristic of relevance computation. A positive value of w_k means that k appears in relevant texts, while a negative value of w_k means that k appears in non-relevant documents.

Statistical ways to exploit the relevance information of terms were presented by Robertson and Sparck Jones (1976). It was especially shown that relevance weighting is inferred by a general probabilistic theory of retrieval.

2.3. Sentiment polarity detection

In a study conducted by Kemaloğlu et al. (2021), Turkish SA was performed on social media. In their study, the Logistic Regression and Random Forest algorithms were used as the machine learning algorithm and LSTM was used for the deep learning algorithm. For the dataset, data were collected from five different social media platforms, such as Twitter, etc. In total 28,189 data were collected and these data were manually labeled as positive, negative, or neutral. After the labeling process, this dataset included 5712 positive, 11,567 negative, and 11,247 neutral tweets. According to the experimental results, it was observed that the deep learning models outperformed machine learning approaches with achieving better performance. The model implemented with random forest achieved at most 84.24 % accuracy while the model implemented with logistic regression achieved at most 83.07 % accuracy rate and the model implemented using LSTM achieved 84.46 % accuracy.

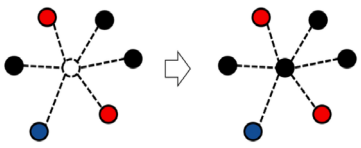


Fig. 1. An example of the LPA in a Sample Network (Bae et al., 2020).

In a study by Aktaş et al. (2022), a Turkish SA was performed on the reviews of an online food order webpage using machine learning models. The proposed model was able to predict if the given review was positive or negative. As a first step, the dataset was collected using the reviews from an online food ordering platform called *yemeksepeti.com*[1]. The collected dataset included a total of approximately 676,000 reviews, of which 338,000 were positive and the remaining 338,000 were negative. For preprocessing, lemmatization and normalization were applied. To implement the classification models, various machine learning algorithms and neural networks were used. These approaches were implemented using KNN, NB, and neural networks. According to the test results, the neural network performed better than the classic machine learning approaches. When performing experiments on the same dataset, the neural network model achieved 86.37 % accuracy for classifying reviews as positive or negative, while the NB model achieved 83.55 % accuracy and the KNN model achieved 81.88 % accuracy.

Kilimci (2020) aimed to predict the direction of the Borsa Istanbul index using SA techniques. Two datasets were built by collecting Turkish and English tweets with BIST100 and XU100 tags on Twitter. The datasets were enriched by using word embedding methods. Furthermore, ensemble techniques were used. The CNN, RNN, and LSTM algorithms, and MV and STCK methods were used. First, ant colony optimization and feature selection were applied to the dataset. Second, the features were embedded. Then, documents were vectorized and classification was performed. Finally, they achieved a classification success of 78.07 % on the Turkish dataset and 78.92 % on the English dataset.

Aytekin and Keskin (2019) proposed a SA method about interest-free finance systems. The analysis was made on Turkish texts. They aimed to detect the positive and negative perceptions of potential customers towards interest-free financial systems. The dataset consisted of Turkish reviews of customers in January 2019 that were collected from various resources. A sentence level sentiment analysis was performed. Initially, they specified the most frequently used concepts, which were related to the interest-free finance systems. Then, they reviewed the data with the specified keywords. Finally, they classified the texts as positive, negative, or neutral and analyzed the results. To conclude, the fact that "participation banks" and the concept of "interest" were mentioned together in the news created a negative prejudice towards these institutions.

Sarıman and Mutaf (2020) aimed to analyze people's feelings about COVID-19 through social media using machine learning methods (Sarıman & Mutaf, 2020). They collected 2 million tweets and processed the data. They obtained the sentence analysis, emotion analysis, and the meaning of the tweets. They evaluated the system and used the area under the ROC curve (AUC) metric as the base metric of success. The labeled classes were Mask, Eba, Curfew, State Support, and Short Working Allowance. They obtained the AUC value, after classifying the tweets as positive and negative. At the end of the study, 97 % classification accuracy was achieved for the Mask class, while it was 94 % for the Eba class, 98 % for the Curfew class, 86 % for the State Support class, and 91 % for the Short Working Allowance class.

3. Methodology

In this paper, a method is introduced for two fundamental tasks: sentiment polarity detection task and topic-based text classification task. Fig. 2 shows the basic flow of the suggested system.

The main steps will be explained in the following respective subsections: preprocessing, sentiment polarity classification and topic-based text classification with the suggested algorithms: Label Propagated-Relevance Frequency Classifier $_{(LP\text{-}RFC)}$, Label Propagated-Abstract Frequency Classifier $_{(LP\text{-}AFC)}$, Composite Classifier ($LP\text{-}Com_{RFC+AFC}$), Merging Unlabeled and Labeled Instances with Semantic Values of Terms (MULIS).

3.1. Preprocessing

Stemming and stop word filtering were applied to all of the datasets in this step. Furthermore, attribute selection was used and the most informative 2000 terms were selected with information gain (IG). In the Twitter dataset, for lemmatization, Zemberek-NLP (Akın & Akın, 2007) was used. In order to create topical clusters in the dataset, Mallet's Latent Dirichlet Allocation (LDA) implementation was used (Graham et al., 2012). LDA is a modeling algorithm that generates topics according to the frequency of words in texts. It helps to find the topics of the text. First, the data were collected and cleaned[1].

Then a suitable topic-label was assigned to the text. The distribution of words over topics and the distribution of topics over documents were calculated and consequently, this topic-labelling process was accomplished.

3.2. Classification

In the classification phase, both supervised and semi-supervised classifiers were built. The supervised classifiers were LP-RFC, LP-AFC, and $LP\text{-}Com_{RFC+AFC}$. Furthermore, four different variants of the semi-supervised MULIS classifier were implemented. They were called $MULIS_{RFC\text{-}AFC}$, $MULIS_{RFC\text{-}RFC}$, $MULIS_{AFC\text{-}RFC}$, and $MULIS_{AFC\text{-}AFC}$. Thus, in this study, three fresh supervised and four fresh semi-supervised classifiers were suggested.

For the classification, the well-known LP algorithm (Zhu & Ghahramani, 2002) was improved by adopting the RFC and AFC. All of the implementations were done in the Scikit-Learn Library[1].

For $l, u \in \mathbb{N}^+$ with $l < u$ and $l+u = n$ we let D$= \{d_1 \cdots d_n\}$ denote the array of documents, and each d_i is a vector of terms, denoted by t. Consequently, D is a *term* $\times$ *document* matrix including the topic-based class labels.

We let class topic-based labels for documents be classified, where $Y_L = \{y_1 \cdots y_l\}$denotes the topic categories, $Y_U = \{y_{l+1} \cdots y_{l+u}\}$ denotes the unknown topics, and$Y = Y_L \cup Y_U$.

Fig. 3 shows the flowchart of LP-RFC and LP-AFC. The first step in the traditional LP algorithm is to build a fully connected graph whose vertices represent the documents. The second step in the LP algorithm is to connect these vertices by edges weighted with a similarity score, which is denoted byw_{ij} and calculated as follows:

$$w_{ij} = e^{-\frac{||d_i - d_j||^2}{2\sigma^2}} \tag{2}$$

Here, the graph's vertices are shown as$(d_1, y_1) \cdots (d_{l+u}, y_{l+u})$ andσ is the scaling parameter. The edges between the vertices, say $, j$, are weighted by using a similarity score between vectors d_i andd_j, which is denoted byw_{ij}. The simplest similarity score is the dot product of the vectors. Another popular way of computing the similarity is the radial basis function (RBF). In this study, for both supervised and semi-supervised classifiers, these similarity scores are computed with relevance frequency calculation and abstract frequency calculation techniques. In other words, two novel similarity score calculation techniques are adopted into the propagation phase of original LPA. For example, in Fig. 3-a the edges between the vertices are weighted by using a similarity score which was calculated according to relevance frequency calculation and abstract frequency calculation. In the second iteration the algorithm propagates the class label of a node in a graph to neighboring nodes by using weighted edges and proximity as shown in Fig. 3-b and in Fig. 3-c.

3.2.1. Label Propagated-Relevance frequency classifier (LP-RFC)

The relevance values of the terms are calculated using only the texts in the training corpus set, as shown in Eq. (3). This calculation provides a term-relevance matrix, which displays the word relevance values for each class. It is clear that these relevance scores are high for the words that have the capability to distinguish between classes. In another way, terms that are specific for only a class will have higher relevance values, where the terms that are not specific for a class will have lower relevance values for that class.

For the classification step, the total relevance score is calculated from an unseen test file. In order to perform this, the relevance value of each class, the P is calculated by summing the relevance scores of terms for that specific class as shown in Eq. (3). Then a test document is labeled with the label of the class P with choosing the highest relevance score. This process is similar to the classification step of NB since the class-conditional document probability is calculated by multiplying the class-conditional term probabilities[2].

The formula for TF-RF is as given below (Lan et al., 2009):

$$TF - RF_{d,P} = \sum_{i}^{|W|} tf_w \times log\left(2 + \frac{a}{\max(1,k)}\right), \quad w \in d \tag{3}$$

Here, tf_w shows the term frequency of word w, a represents the number of documents in the positive category that include word w, and c represents the number of documents in the negative category that include word w. Consequently, this calculation ranks terms according to their discriminating powers for the classes. For instance, for the 23 M−Tweets dataset, there are 5 categories: *Politics*, *Education*, *Health*, *Space and Sports*.

The word cloud images for the highest relevance values for three of these classes *Health*, *Politics*, and *Space* are shown in Fig. 4.

Where (a) represents the word cloud image for the health class and for this class, "*virus*", "*disease*", and "*respiratory*" are the words more at the forefront because these are the most frequent words that appear in the health class. Where (b) represents the word cloud image for the politics class "*Europe*", "*natural*", and "*partners*" are the words at the forefront in this class. And finally, where (c) represents the word cloud image for the space class and "first", "moon", and "explore" are words at the forefront in this class.

3.2.2. Label Propagated-Abstract frequency classifier (LP-AFC)

The LP-AFC was implemented, which is similar to the LP-RFC, but the only difference is that the class-based term values are computed using Eq. (4). The LP-AFC includes all three steps of the LP-RFC, but the LP-AFC uses calculations of the abstract features, which are shown in Eq. (4). The total abstract weight of a test document d is calculated using Eq. (4):

$$W(d,P) = \sum_{i}^{|W|} (\log(tf_{w,P} + 1) \times \log(\frac{N}{N_w})), \ \forall w \in d \tag{4}$$

Here, P is class, W shows the total number of terms in document d, and w represents the term (Biricik et al., 2012).

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

[2] https://scikit-learn.org/

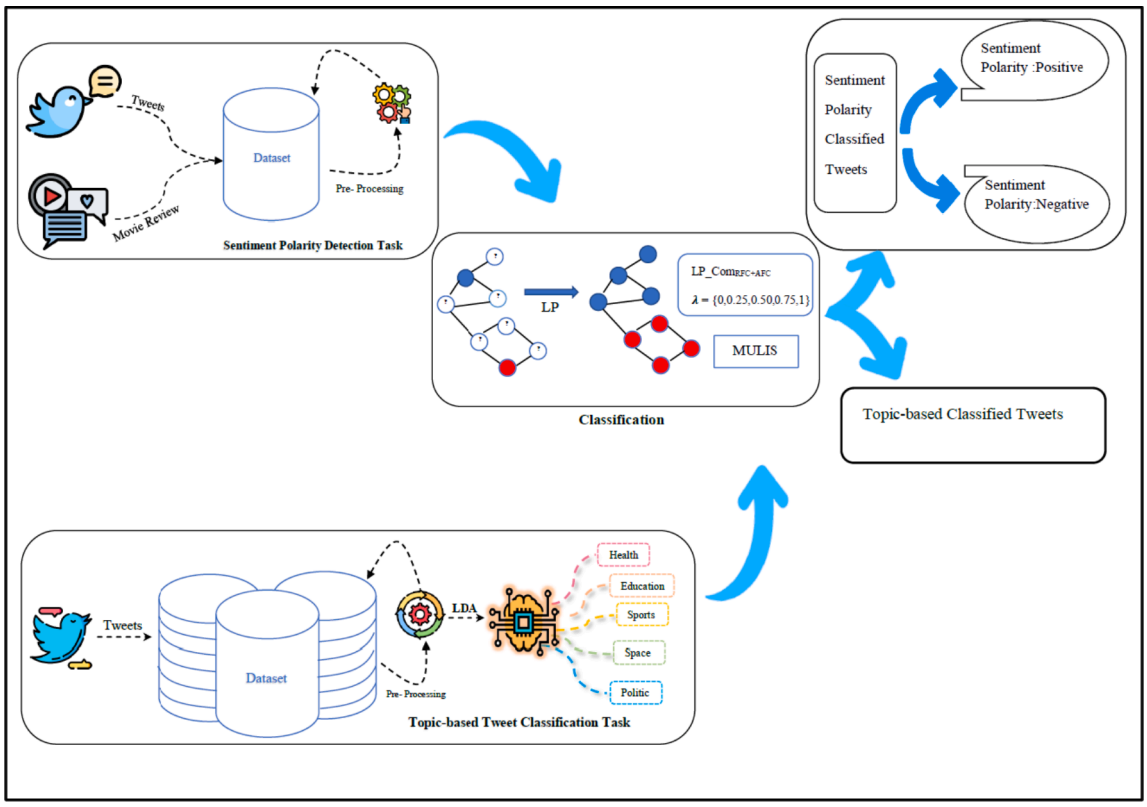
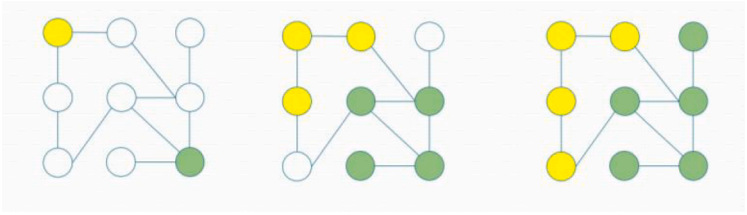


Fig. 2. Flowchart of the suggested system.



a) initial state: edges between the vertices are weighted by using a similarity score which was calculated according to relevance frequency calculation and abstract frequency calculation.

b) first iteration: label propagation is done according to calculated similarity score

c) second iteration: This process is repeated until convergence to a unique solution is achieved.

Fig. 3. Flowchart of the LP-RFC and LP-AFC.

Fig. 4. Word cloud images for the Health, Politics, and Space classes.

3.2.3. Composite classifier (LP-Com$_{RFC+AFC}$)

The composite classifier is actually a weighted summation classifier of the LP-RFC and LP-AFC, as shown in Eq. (5):

$$LP-Com\,(w,P)=\lambda(RF(w,P))+(1-\lambda)(AF(w,P)) \quad (5)$$

Here, λ is a positive real number$(0<\lambda<1)$, $RF(w,P)$ shows the RF value of word w for class P, $AF(w,P)$ shows the AF value of word w for class P, and $Com(w,P)$ is the composite total semantic value of word w for class P. In order to optimize λ, the following values are taken into consideration: {0, 0.25, 0.5, 0.75, 1}.All the experiments are run with all different λ values in order to see its effect. In other words,

These λ values are randomly taken between 0 and 1 so that λ actually shows which algorithm (i.e. LP-RFC or LP-AFC) is best for which case. Because when λ is equal to zero this means that LP-AFC algorithm generates better classification result than LP-RFC algorithm for the corresponding task.

3.2.4. Merging unlabeled and labeled instances with semantic values of terms (MULIS)

MULIS is actually a hybrid semi-supervised algorithm, which takes advantage of the classification capacities of two classifiers: the LP-AFC and LP-RFC. MULIS comprises three steps: preprocessing, labeling, and classification. There are three sets of documents: 1) L_o: originally labeled documents, 2) L_u: unlabeled documents, and 3) L_T: test documents. After the preprocessing step, in the labeling step, the AFC is trained by the L_o and the L_u are labeled by the AFC, and then a document set with their predicted labels is generated (L_p). These labeled instances are then joined to the originally labeled instances, as shown in Eq. (6):

$$L=L_o+L_p \quad (6)$$

Subsequently, the labeled corpus is enlarged. After that, the LP-RFC is trained with this enlarged corpus, L. Finally, test instances (L_T) are classified by the LP-RFC.

Furthermore, four variants of MULIS were implemented, as follows:

1) MULIS$_{RFC\text{-}AFC}$: It uses the LP-RFC in order to label unlabeled instances in the labeling phase. Then it combines these labeled instances with originally labeled instances and it generates an extended labeled set. Finally, it trains the LP-AFC on this extended labeled set and attempts to classify test instances.
2) MULIS$_{RFC\text{-}RFC}$: This uses the LP-RFC in order to label unlabeled instances in the labeling phase. Then it combines these labeled instances with originally labeled instances and it generates an extended labeled set. Finally, it trains the LP-RFC on this extended labeled set and attempts to classify test instances.
3) MULIS$_{AFC\text{-}RFC}$: This uses the LP-AFC in order to label unlabeled instances in the labeling phase. Then it combines these labeled instances with originally labeled instances and it generates an extended labeled set. Finally, it trains the LP-RFC on this extended labeled set and attempts to classify test instances.
4) MULIS$_{AFC\text{-}AFC}$: This uses the LP-AFC in order to label unlabeled instances in the labeling phase. Then it combines these labeled instances with originally labeled instances and it generates an extended labeled set. Finally, it trains the LP-AFC on this extended labeled set and attempts to classify test instances.

4. Experimental setup

4.1. Datasets

In this study, three different datasets were used for the Sentiment Polarity detection task and one dataset was used for the topic-based tweets categorization task. The datasets used for the Sentiment Polarity detection task can be seen in Table 1, where the datasets used in this study are presented according to the dataset size, positive tweet percentage, and negative tweet percentage.

Table 1
Properties of Datasets.

Dataset	Type	Dataset Size	Positive Tweet %	Negative Tweet %
Twitter-3 k	Tweets	3044	49 %	51 %
Movie-Review-19 k	Movie Review	19,206	39 %	61 %
Movie-Review-100 k	Movie Review	100,000	42 %	58 %

Three different datasets were used for Sentiment Polarity detection. These datasets were:

Twitter-3 k dataset: This dataset includes 3044 tweets, and of these, 49 % are sentimentally positive and the remaining 51 % are sentimentally negative.

Movie-Review-19 k dataset: This dataset includes 19,206 movie reviews, and of these, 39 % are sentimentally positive and the remaining 61 % are sentimentally negative.

Movie-Review-100 k dataset: This dataset includes 100,000 movie reviews, and of these, 42 % are sentimentally positive and the remaining 58 % are sentimentally negative.

In this study, one dataset was used for topic-based tweets categorization. This dataset was created in March of 2021 by collecting approximately 23 million tweets within the following categories: Politics, Education, Health, Space, and Sports. A total of 23,214,872 tweets were collected between February 1st and March 15th, 2021. These tweets were accumulated by indexing on MongoDB. The tweets were collected using Twitter Streaming API. Additionally, Apache Spark, which uses a random distributed dataset (RDD) to process data in the memory with many clusters, was used while collecting and processing the data. Since Spark runs in local memory, it is faster than Hadoop, so Spark was used in this phase.

The tweets were categorized using the LDA. A cleaning process was applied to the tweets. With this process, mentions, hashtags, URLs, emojis, and punctuations were removed from the context of the tweets. Using the Zemberek tool (Akın & Akın, 2007), stemming and stop word filtering were applied to the tweets. After cleaning the data, MALLET (McCallum, 2002) was used to apply the LDA algorithm to determine the categories of the first dataset. Since tweets are short documents that are composed of text, one of the pooling methods, as used in (Girgin, 2021), was used to produce more coherent clusters. Human experts reviewed the word clusters generated by the LDA, and some tweets were extracted from their categories. To complete the dataset, topic labels were added to the pooled tweets. As a result, five different topic categories obtained. The distribution of the categories of the tweets in this dataset was as follows:

As can be seen in the pie chart in Fig. 5, with the effects of COVID-19, the majority of the tweets were about health at 39 %, and then this was

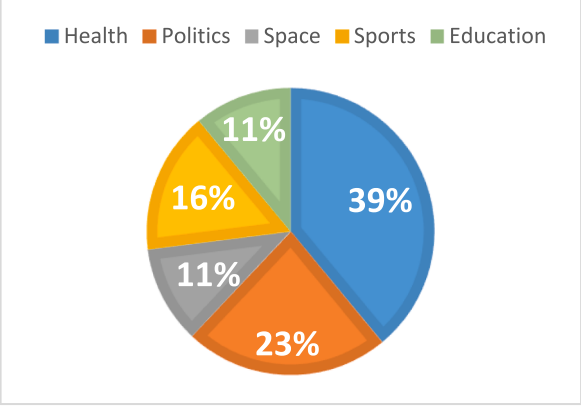


Fig. 5. Distribution of Categories.

followed by tweets about policy at 23 %, tweets about sports at 15 %, and lastly, tweets about both education and space at 11 %.

4.2. Baseline algorithms

SVM with a linear kernel with the LP algorithm was our first baseline algorithm. It should be noted that this algorithm is one of the most important state-of-the-art algorithms in the text classification domain (Joachims, 1998). The second baseline algorithm was RBF with the LP. RBF is the key baseline algorithm for the LP since it results in the highest classification performance. This highest classification accuracy can particularly be observed in the Scikit-Learn Library with optimized parameters (Zhu & Ghahramani, 2002). The third baseline algorithm was NB. The fourth and fifth baseline algorithms were word2vec-skip gram (SG) with linear kernel and word2vec (CBOW) with linear kernel.

4.3. Experimental setup

In the experiments for supervised algorithms, the following percentage values were used for training the set size: 30 %, 50 %, and 70 %. The remaining documents were used for testing.

In the experiments for the semi-supervised algorithms, the following percentage values were used for the labeled, unlabeled, and test data: 30 %, 50 %, and 20 %, respectively. The F1-score (F1) metric is the main evaluation metric applied in the experiments. The conventional 10-fold cross-validation (CV) technique was used for all of the experiments on all of the datasets. Consequently, we report the average of the corresponding F1 scores on the test portions of the datasets. The experimental results of the LP-$Com_{RFC+AFC}$, with different λ values (i.e., 0, 0.25, 0.5, 0.75, 1) and several variants of MULIS (i.e., $MULIS_{RFC\text{-}AFC}$, $MULIS_{RFC\text{-}RFC}$, $MULIS_{AFC\text{-}RFC}$, and $MULIS_{AFC\text{-}AFC}$), and baseline algorithms are given in Tables 2–6.

All of the experiments were run on a machine with Intel i7-8750 @2.20 Ghz CPU and 32 GB RAM. The implementation in this study was done in JAVA and Python programming languages. Additionally, for the experiments on the Twitter dataset, Scikit Learn's SVC class was used, as well as the Apache Spark Ecosystem. Furthermore, the TWINT library from Python 3.6 was used to obtain the Twitter data. All of the data collected on Twitter was stored on a local server by being indexed on MongoDB. The Zemberek library in JAVA was used in order to preprocess the data collected. Moreover, Apache Spark was used in order to make the vector calculations of the SVM in a distributed way, which sped up the process a lot. Moreover, SVM was used with the following default parameters in all of the experiments in this study: 'C' = 1, 'dual' = 'false', 'penalty' = 'l2' parameters of the scikit-learn[3] library. For the Skip-Gram model of the Word2Vec model, the Python 3.6 library named Gensim[1] was used.

In LSTM algorithm the labels are edited using to_categorical from the Keras library,[4] [5]. Then an embedding matrix was created in the embedding layer by Glove.6B.50d. Other deep learning models are used just their default parameter in standard library.

In order to compare our semi-supervised algorithm (MULIS), a semi-supervised baseline algorithm was also used. It is called SSL-linear

Table 2
Experiment results on Twitter-3 k dataset.

Method	Training: 30 % Test: 70 % ***F1%***	Training: 50 % Test: 50 % ***F1%***	Training: 70 % Test: 30 % ***F1%***
LP-Linear kernel	88.81	90.11	91.80
LP-RBF	90.82	91.22	92.23
Naïve Bayes	88.03	88.64	89.44
LP-$Com_{RFC+AFC}$ $\lambda = 0$	91.27	91.55	92.57
LP-$Com_{RFC+AFC}$ $\lambda = 0.25$	91.29	91.76	92.78
LP-$Com_{RFC+AFC}$ $\lambda = 0.5$	91.32	91.81	92.89
LP-$Com_{RFC+AFC}$ $\lambda = 0.75$	91.35	91.90	92.94
LP-$Com_{RFC+AFC}$ $\lambda = 1$	**91.41**	**92.05**	**93.11**
Word2vec-SG + linear kernel	91.09	91.91	91.04
Word2vec(CBOW) + linear kernel	88.9	90.12	86.46

Table 3
Experiment results on Movie-Review-19 k dataset.

Method	Training: 30 % Test: 70 % **F1%**	Training: 50 % Test: 50 % **F1%**	Training: 70 % Test: 30 % **F1%**
LP-Linear kernel	87.12	88.17	89.04
LP-RBF	88.23	89.21	91.08
Naïve Bayes	85.37	86.54	87.97
LP-$Com_{RFC+AFC}$ $\lambda = 0$	90.27	91.06	91.98
LP-$Com_{RFC+AFC}$ $\lambda = 0.25$	91.04	91.79	92.53
LP-$Com_{RFC+AFC}$ $\lambda = 0.5$	91.33	91.82	92.09
LP-$Com_{RFC+AFC}$ $\lambda = 0.75$	**91.56**	**92.99**	**94.34**
LP-$Com_{RFC+AFC}$ $\lambda = 1$	91.40	92.09	92.49
Word2vec-SG + linear kernel	91.45	92.71	94.01
Word2vec(CBOW) + linear kernel	90.93	91.32	92.47

Table 4
Experiment results on Movie-Review-100 k dataset.

Method	Training: 30 % Test: 70 % **F1%**	Training: 50 % Test: 50 % **F1%**	Training: 70 % Test: 30 % **F1%**
LP-Linear kernel	90.94	92.27	94.00
LP-RBF	93.00	93.41	94.44
Naïve Bayes	90.14	90.77	91.59
LP-$Com_{RFC+AFC}$ $\lambda = 0$	93.46	93.75	94.79
LP-$Com_{RFC+AFC}$ $\lambda = 0.25$	93.48	93.96	95.01
LP-$Com_{RFC+AFC}$ $\lambda = 0.5$	**93.61**	**94.26**	95.12
LP-$Com_{RFC+AFC}$ $\lambda = 0.75$	93.54	94.11	95.17
LP-$Com_{RFC+AFC}$ $\lambda = 1$	93.60	94.06	95.34
Word2vec-SG + linear kernel	93.28	94.13	**95.84**
Word2vec(CBOW) + linear kernel	91.03	92.26	93.18

Table 5
Experiment results on the Twitter-23 M dataset.

Method	Training: 30 % Test: 70 % **F1%**	Training: 50 % Test: 50 % **F1%**	Training: 70 % Test: 30 % **F1%**
LP-Linear kernel	88.21	89.50	91.18
LP-RBF	90.21	90.61	91.61
Naïve Bayes	87.44	88.05	88.84
LP-$Com_{RFC+AFC}$ $\lambda = 0$	91.66	91.94	92.95
LP-$Com_{RFC+AFC}$ $\lambda = 0.25$	91.18	92.14	93.16
LP-$Com_{RFC+AFC}$ $\lambda = 0.5$	**92.80**	**92.93**	93.27
LP-$Com_{RFC+AFC}$ $\lambda = 0.75$	92.73	92.89	93.21
LP-$Com_{RFC+AFC}$ $\lambda = 1$	92.59	93.24	93.18
Word2vec-SG + linear kernel	91.48	92.31	92.96
Word2vec (CBOW) + linear kernel	88.30	89.49	90.38
LSTM	91.98	92.49	**93.08**

[3] https://pypi.org/project/gensim/.
[4] https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM.
[5] https://keras.io/.

Table 6
Experiment results with 30 % labeled, 50 % unlabeled, and 20 % test splits on the Movie-Review-3 k, Movie-Review-19 k, and Movie-Review-3 k datasets.

Method/ Dataset	Movie-Review-3 k **F1%**	Movie-Review-19 k **F1%**	Movie-Review-100 k **F1%**
Baseline-1: SSL-Linear:	70.12	87.82	86.16
Baseline-2: Linear	71.37	87.10	84.15
$MULIS_{RFC-AFC}$	78.98*	94.31*	88.30*
$MULIS_{RFC-RFC}$	76.36*	91.93*	84.36
$MULIS_{AFC-RFC}$	79.14*	93.54*	87.43
$MULIS_{AFC-AFC}$	**80.75***	**95.08***	**91.27***

(Altınel et al., 2017) with the LP and it is actually a semi-supervised version of the customary linear kernel.

All the suggested algorithms in this study can be accessed via a GitHub link.[6]

5. Results and discussion

5.1. Experiments related to the sentiment polarity detection task

The sentiment polarity detection task is considered as a classification task since there are two labels (i.e. positive label and negative label) in the datasets as shown in Table 1. The suggested supervised methods were run on Twitter-3 k, Movie-Review-19 k and Movie-Review-100 k datasets in order to detect sentiment polarity of the test instances. In this study, no additional sentiment-polarity dictionary is used in order to capture the sentiment score of each word; which has some limitations (i.e. the coverage of the dictionary, computational cost) and some dependencies (i.e. language). Instead, this task is fully dealt as a supervised classification problem with the suggested supervised algorithms. This brings some advantages such as being compatible with any language and doing not require the processing of large dictionary resources.

The suggested supervised methods were evaluated and compared with baseline algorithms that are commonly used techniques on the sentiment polarity detection task. The experimental results are shown in Tables 2–4.

Table 2 presents the experimental results of all of the implemented baseline algorithms, including the LP-linear kernel, LP-RBF, word2vec-skip gram (SG) with linear kernel and word2vec (CBOW) with linear kernel, and the suggested supervised methodology $LP\text{-}Com_{RFC+AFC}$ with different λ values, such as 0, 0.25, 0.5, 0.75, and 1 on the Twitter-3 k dataset. At the training level of 70 %, the F1 scores of the LP-linear kernel, LP-RBF, and NB were 91.80 %, 92.23 %, and 89.44 %; respectively. On the other hand, at the same training level for the Twitter-3 k dataset, the F1 scores of the suggested algorithm, $LP\text{-}Com_{RFC+AFC}$, with different λ values, such as 0, 0.25, 0.5, 0.75, and 1 were 92.57 %, 92.78 %, 92.89 %, 92.94 %, and 93.11 %, respectively.

It is also worth noting that at the 30 % training level, all of the proposed algorithms were better than the baseline algorithms. The superiority of the suggested algorithms over the baseline algorithms could be explained by the usage of class-based term weighting techniques, which emphasized specific words for each class by giving them higher weights. Consequently, this process discriminates classes successfully and results in high classification accuracy.

Furthermore, all of the baseline algorithms and the offered algorithms were also run on the Movie-Review-19 k dataset. The experimental results are reported in Table 3. One of the most important points understood from Table 3 is that most of the variants of the $LP\text{-}Com_{RFC+AFC}$ algorithm were better than those of word2vec (CBOW) with linear kernel at all of the training splits. Another important point noticed from Table 3 is that the $LP\text{-}Com_{RFC+AFC}$ algorithm with λ value 0.75 produced higher classification performance than word2vec (SG) with linear kernel at all of the training splits. The classification performance gains of the $LP\text{-}Com_{RFC+AFC}$ algorithm over word2vec (SG) with linear kernel and word2vec (CBOW) with linear kernel is very important since the complexity of the $LP\text{-}Com_{RFC+AFC}$ algorithm is quite low and consequently requires low resources.

According to Table 4, the improvements of the $LP\text{-}COM_{RFC+AFC}$ over the baseline algorithms were noticeable, especially at small training splits, which has great importance as it is very expensive to obtain labeled data in real-world scenarios. According to the experimental results in Table 4, linear kernel generated higher classification accuracy than word2vec (CBOW) with linear kernel at a training split of 70 % on the Movie-Review-100 k dataset. This means that, although word2vec is a more complex algorithm, linear kernel with the BOW feature representation produced higher classification accuracy than linear kernel with vector representations. Moreover, linear kernel generated higher classification accuracy than NB, but it still ranked far behind the $LP\text{-}Com_{RFC+AFC}$ at training splits of 30 %, 50 %, and 70 %. Another critical point that needs to be noted is that although word2vec is a more complex algorithm, the $LP\text{-}COM_{RFC+AFC}$ with the BOW feature representation produced higher classification accuracy than linear kernel with vector representations.

Moreover, the total computation time of the LP-linear kernel, LP-RBF, NB, word2vec (SG) with linear kernel and word2vec (CBOW) with linear kernel, and $LP\text{-}Com_{RFC+AFC}$ with λ value 0.5 were recorded in terms of seconds on the Movie-Review-19 k dataset. All of the experiments reported herein were run on a machine with Intel i7-8750 @2.20 Ghz CPU and 32 GB RAM. The computation time of all of the algorithms in this study on the Movie-Review-19 k dataset is shown in Fig. 6. According to the experiment records, the computation time of the LP-linear kernel, LP-RBF, NB, word2vec (SG) with linear kernel and word2vec (CBOW) with linear kernel, and $LP\text{-}Com_{RFC+AFC}$ with λ value 0.5 were 1180, 1354, 1230, 4402, 4382, and 2201 s.

This computational difference between the LP-linear kernel, LP-RBF, NB and $LP\text{-}Com_{RFC+AFC}$ with λ value 0.5 was appreciated as the statistical based semantic labelling process of the unlabeled data in the $LP\text{-}Com_{RFC+AFC}$. Actually, this might be one of the possible future improvement items of the $LP\text{-}Com_{RFC+AFC}$ in our agenda. On the other hand, the $LP\text{-}Com_{RFC+AFC}$ algorithm predictably seemed to be faster than word2vec (SG) with linear kernel and word2vec (CBOW) with linear kernel since these deep learning algorithms indeed use more complex calculations.

In order to show the effectiveness of the suggested algorithms, the

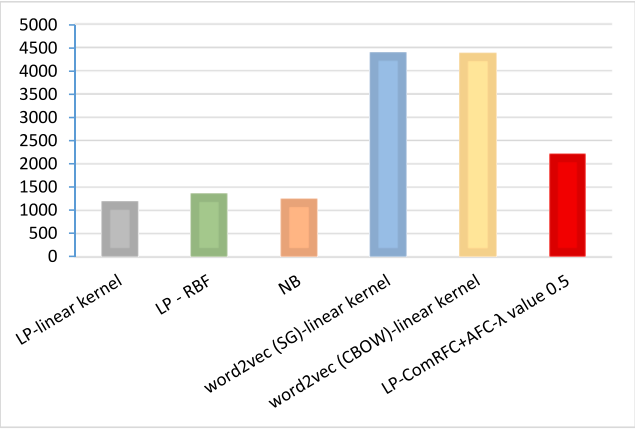


Fig. 6. Total computation time units of the algorithms.

[6] https://github.com/bilgesipal/Tubitak3500/tree/batikan-main.

algorithms were also compared with two other recent methods:

The first comparison was with the study of Kilimci (2020), which presented the experimental results of deep learning algorithms on a Turkish dataset that comprised Turkish tweets of the BIST100 tag on Twitter. The deep learning algorithms used in this study were the CNN, RNN, and LSTM algorithms, and they were used in ensemble form. It was attempted to build the same experimental environment by collecting tweets with the BIST100 tag on Twitter and use the same feature selection method as mentioned in (Kilimci, 2020) in order to compare the experimental results. At the end, the suggested algorithms were run on this dataset. According to the experimental results, the LP-$Com_{RFC+AFC}$ with λ value 0.5 achieved 83.21 % classification accuracy while the ensemble methodology in (2020) achieved 78.07 % classification accuracy on the Twitter-BIST100 dataset. The performance superiority of the L.

P-$Com_{RFC+AFC}$ with λ value 0.5 over this ensemble method may be explained with the enrichment of the propagation process of the LPA with semantic values.

The LP-$Com_{RFC+AFC}$ with λ value 0.5 was also run on the SemEval-2016 Slot-3[1] (Pontiki et al., 2016) Turkish sentiment polarity task. In this task, the REST dataset was used, which consists of Turkish restaurant reviews. The classification accuracy of the LP-$Com_{RFC+AFC}$ with λ value 0.5 was 84.75 %. On the other hand, according to the submissions reported in (Pontiki et al., 2016), the classification accuracy of the baseline algorithm was 72.32 % and the other teams submitted 84.27 % and 74.20 % accuracy. This experiment showed that the 84.75 % accuracy score was slightly better than the best score, which was 84.27 %. This performance accuracy also seemed better that the results reported in (Ozyurt & Akcayol, 2021), in which it was attempted to build a semi-supervised form of the LDA with an aspect level and use the REST dataset in SemEval-2016 Slot-3.

5.2. Experiments related to Topic-Based classification Task:

The suggested supervised methods were compared with baseline algorithms that are commonly used techniques on the topic-based text classification task. The experimental results are shown in[7] Tables 5 and 6.

Table 5 presents the experimental results of all of the algorithms on the Twitter-23 M dataset. According to the results presented in Table 5, the difference between the F1 score of the LP-$Com_{RFC+AFC}$ with λ value 0.5 and the F1 score of the LP-linear kernel was 4.59 %, which is a very important difference since in real-world cases, it is very difficult to find labeled data. Hence, even a slight performance gain is very significant, especially in application domains such as scenarios related to health, country national defense systems, etc. While word2vec-SG with linear kernel achieves the highest F1 score of 92.96 at a 70 % training split among the baseline algorithms LP-linear kernel, LP-RBF, NB, and word2vec (CBOW) with linear, it performed remarkably worse than the LP-$Com_{RFC+AFC}$ with different λ values, such as 0.25, 0.5, 0.75, and 1 in terms of the F1 score. Moreover, at all of the training levels, all of the proposed algorithms were significantly better than at least one of the baseline algorithms.

Also compared were the F1 scores of the LP-$Com_{RFC+AFC}$ algorithm with the LSI-kNN algorithm on the Twitter-23 M dataset. The LSI-kNN algorithm applies the standard kNN algorithm over LSI semantic space. Experiments were performed with k parameters of {1, 5, 10, 50, 100} and it was observed that the best results were achieved with k values of 50 and 100. Fig. 7 shows the F1 scores of the LSI-kNN with k values of 50 and 100 and the LP-$Com_{RFC+AFC}$ with λ value 0.5. According to Fig. 7, at a 30 % training split, the F1 score of the LP-$Com_{RFC+AFC}$ was 92.8 %, while the F1 scores of the LSI-kNN with k values of 50 and 100 were 81.52 % and 88.23 %, respectively.

[7] https://alt.qcri.org/semeval2016/task5/.

In order to compare the algorithms herein with one of the state-of-the-art algorithms, such as LSTM, the LSTM algorithm was also run on the Twitter-23 M dataset. The experimental results are presented in Table 5, where it is seen that the LP-$Com_{RFC+AFC}$ significantly boosted the classification performance. The classification success of the algorithms at a 30 % training split on the Twitter-23 M dataset can be ordered as the LP-$Com_{RFC+AFC}$ algorithm with λ value 0.5 > LP-$Com_{RFC+AFC}$ algorithm with λ value 0.75 > LP-$Com_{RFC+AFC}$ algorithm with λ value 1 > LSTM > LP-$Com_{RFC+AFC}$ algorithm with λ value 0 > Word2vec-SG + linear kernel > LP-$Com_{RFC+AFC}$ algorithm with λ value 0.25 > LP-RBF > Word2vec (CBOW) + linear kernel> > LP-Linear kernel > Naïve Bayes. In fact, on the Twitter-23 M dataset, at all of the training splits, at least one of the proposed algorithms was always superior to LSTM.

The variants of the proposed semi-supervised method were also evaluated and compared with two baseline algorithms, the linear kernel and SSL-linear. Table 6 presents the experimental results of the linear kernel and SSL-linear algorithms, and the variants of the proposed semi-supervised method, $MULIS_{RFC\text{-}AFC}$, $MULIS_{RFC\text{-}RFC}$, $MULIS_{AFC\text{-}RFC}$, and $MULIS_{AFC\text{-}AFC}$ on the Movie-Review-3 k, Movie-Review-19 k, and Movie-Review-3 k datasets. Additionally, student t-tests were conducted with a significance level of $\alpha = 0.05$. Moreover, the algorithms that were significantly better than the baseline algorithms are shown with "*" in Table 6.

According to Table 6, the SSL-linear algorithm achieved an F1 score of 70.12 %, while the linear kernel achieved an F1 score of 71.37 % on the Movie-Review-3 k dataset. On the other hand, the F1 scores of $MULIS_{RFC\text{-}AFC}$, $MULIS_{RFC\text{-}RFC}$, $MULIS_{AFC\text{-}RFC}$, and $MULIS_{AFC\text{-}AFC}$ were 78.98 %, 76.36 %, 79.14 %, and 80.75 %, respectively, on the Movie-Review-3 k dataset. Furthermore, as can be observed from Table 6, on the Movie-Review-19 k and Movie-Review-100 k datasets $MULIS_{AFC\text{-}AFC}$ was superior to the baseline algorithms.

6. Conclusion and future directions

This paper presented several different algorithms that were based on the LPA in order to handle the two previously mentioned fundamentals tasks: the sentiment polarity detection task and topic-based text classification task. These algorithms were the LP-RFC and LP-AFC. Furthermore, also designed and implemented were the LP-AFC and LP-RFC in a semi-supervised setting, namely MULIS.

According to the experimental results, the LP-RFC, LP-AFC, and LP-$Com_{RFC+AFC}$ generated higher F1 scores than all of the baseline algorithms (i.e., linear kernel, NB, word2vec (CBOW), and word2vec-SG) at nearly all of the training splits on the MovieReview3k, MovieReview19k, MovieReview100k, and Twitter23M datasets. Consequently, an important point derived from the experiments that the suggested novel algorithms are superior to the baseline algorithms on all of the datasets on both two tasks: sentiment polarity detection task and topic-based tweet classification task. This shows that relevance frequency calculation and abstract frequency calculation improved the propagation phase of LPA.

Two comparisons were also made in order to show the effectiveness of the suggested algorithms.

The LSTM algorithm was also run on the Twitter23M dataset. The classification success of the algorithms on the Twitter23M dataset can be ordered as LP-$Com_{RFC+AFC}$ > LSTM > LP-RBF > Word2vec-SG + linear kernel > LP-Linear kernel > Word2vec (CBOW) + linear kernel > Naïve Bayes. This is very important since the proposed algorithms were very simple to design and implement in comparison to complex word embedding models, such as word2vec and LSTM.

The advantages of the suggested algorithms can be listed as follows:

- The proposed algorithms do not need any dictionaries to generate the semantic values; the semantic similarity calculations are done simply using some statistical computations. Therefore, training the

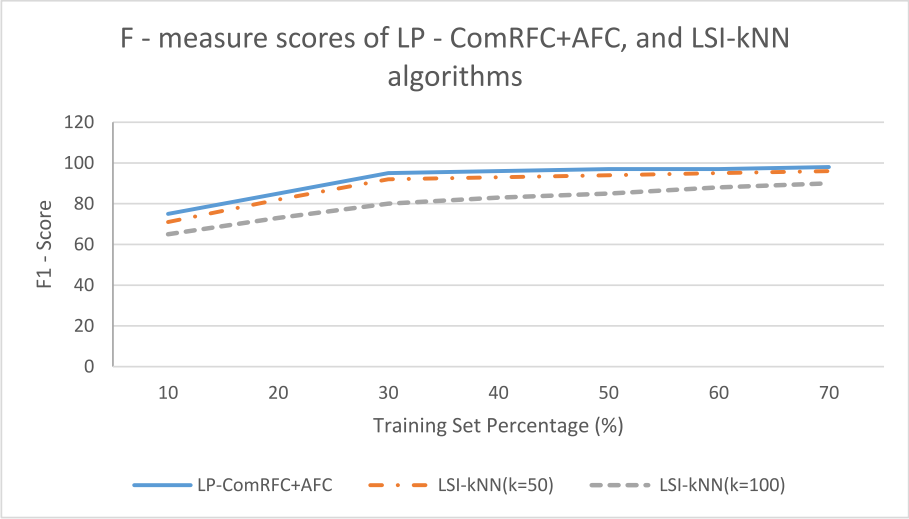


Fig. 7. F-measure scores of the LP-Com$_{RFC+AFC}$ and LSI-kNN algorithms on the Twitter-23 M dataset.

suggested approaches is easy and fast, and it does not require any language-dependent dictionaries, such as Wikipedia, WordNet, etc.

- The experimental results showed that the suggested algorithms are superior to their baselines, such as LP-linear kernel, LP-RBF, and NB. This superiority could be explained by the usage of class-based term weighting techniques, which emphasize specific words for each class by giving them higher weights.
- It is very easy to develop, update, and apply the suggested algorithms; they are not complex algorithms such as word embedding models like word2vec, LSTM. Therefore, they do not require huge resources to carry out experiments, especially in big data environments.
- Word embedding models require necessary optimizations of hyper parameters while the suggested approaches do not require such optimization processes.

It might be a good idea to apply ensemble techniques in order to benefit more from the classification capacities of the LP-RFC, LP-AFC, LP-Com$_{RFC+AFC}$, and MULIS. Applying current and new-generation word embedding models, such as CNN, RNN, LSTM, and BERT, in the propagation phase of the LP algorithm for text classification problems can be considered among future studies.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by The Scientific and Technological Research Council of Turkey (TÜBİTAK) under grant number 118E315 and with grant number 120E187. The points of view in this document are those of the authors and do not necessarily represent the official position or policies of TÜBİTAK.

References

Akın, A. A., & Akın, M. D. (2007). Zemberek, an open source nlp framework for turkic languages. *Structure, 10*, 1–5.

Aktaş, Ö., Coşkuner, B., & Soner, İ. (2022). Turkish Sentiment Analysis Using Machine Learning Methods: Application on Online Food Order Site Reviews. arXiv preprint arXiv:2201.03848.

Altınel, B., Ganiz, M. C., & Diri, B. (2017). Instance Labelling in Semi-Supervised Learning using Meaning Values of Terms. *Engineering Applications of Artificial Intelligence*. https://doi.org/10.1016/j.engappai.2017.04.003

Altınel, B., & Ganiz, M. C. (2018). Semantic text classification: A survey of past and recent advances. *Information Processing & Management, 54*(6), 1129–1153.

Altınel, B., & Gümüşçekiçci, G. (2022a). Turkish Sentiment Analysis: A Survey of Past and Current Improvements. *Turkish Journal of Electrical Engineering and Computer Sciences*.

Altınel, B., & Gümüşçekiçci, G. (2022b). From Past to Present: Spam Detection and Identifying Opinion Leaders in Social Networks. *Sigma Journal of Engineering and Natural Sciences, 40*(2), 441–463.

Aytekin, Y. E., & Keskin, Ö. (2019). The Evaluation of Interest-Free Finance System in Turkey within the Context of Sentiment Analysis. *International Journal of Islamic Economics and Finance Studies, 5*(3), 87–112.

Bae, M., Jeong, M., & Oh, S. (2020). Label Propagation-Based Parallel Graph Partitioning for Large-Scale Graph Data. *IEEE Access, 8*, 72801–72813.

Bilgin, M., & Şentürk, İ. F., 2017. Sentiment analysis on Twitter data with semi-supervised Doc2Vec. In Computer Science and Engineering (UBMK), 2017 International Conference on (pp. 661-666). IEEE.

Biricik, G., Diri, B., & Sönmez, A. C. (2009). A New Method for Attribute Extraction with Application on Text Classification, Soft Computing, Computing with Words and Perceptions in System Analysis. In *Decision and Control (ICSCCW), Fifth International Conference on IEEE* (pp. 1–4).

Biricik, G., Diri, B., & Sönmez, A. C. (2012). Abstract Feature Extraction for Text Classification. *Turkish Journal of Electrical Engineering & Computer Sciences, 20*(1), 1137–1159.

Girgin, B. A. (2021). Ranking influencers of social networks by semantic kernels and sentiment information. *Expert Systems with Applications, 171*, Article 114599.

Graham, S., Weingart, S., & Milligan, I., 2012. Getting started with topic modeling and MALLET. The Editorial Board of the Programming Historian.

Joachims, T. (1998). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features* (pp. 137–142). Berlin Heidelberg: Springer.

Kemaloğlu, N., Küçüksille, E., & Özgünsever, M. E. (2021). Turkish Sentiment Analysis on Social Media. *Sakarya University Journal of Science, 25*(3), 629–638.

Kilimci, Z. H. (2020). Financial Sentiment Analysis with Deep Ensemble Models (DEMs) for Stock Market Prediction. *Journal of The Faculty of Engineering and Architecture of Gazi University, 35*(2), 635–650.

Lan, M., Tan, C. L., Su, J., & Lu, Y. (2009). Supervised and Traditional Term Weighting Methods for Automatic Text Categorization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31*(4), 721–735.

McCallum, A. K., 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Ozyurt, B., & Akcayol, M. A. (2021). A new topic modeling based approach for aspect extraction in aspect based sentiment analysis: SS-LDA. *Expert Systems with Applications, 168*, Article 114231.

Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., ALSmadi, M., … Eryigit, G., 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In Proceedings of the 10th International Workshop on Semantic Evaluation (Semeval-2016), Jan 2016, San Diego, United States. pp.19 - 30, ff10.18653/v1/S16-1002ff. ffhal-01838537.

Raghavan, U. N., Albert, R., & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical review E, 76*(3), Article 036106.

Robertson, S. E., & Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the Association for Information Science and Technology, 27*(3), 129–146.

Sarıman, G., & Mutaf, E. (2020). Sentiment Analysis of Twitter Messages in Covid-19 Process. *Euroasia Journal of Mathematics, Engineering, Natural & Medical Sciences, 7* (10), 137–148.

Shaw, W. M., Jr (1995). Term-relevance computations and perfect retrieval performance. *Information Processing & Management, 31*(4), 491–498.

Şahin, G., 2017. Turkish document classification based on Word2Vec and SVM classifier. In Signal Processing and Communications Applications Conference (SIU), 2017 25th (pp. 1-4). IEEE.

Zhu, X. and Ghahramani, Z., 2002. Learning from labeled and unlabeled data with label propagation. School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CALD-02-107, 2002.