Persian Rumor Detection Using a Multi-Classifier Fusion Approach

Alireza Mansouri

Information Technology Faculty

ICT Research Institute

Tehran, Iran

amansuri@itrc.ac.ir

Mojgan Farhoodi

Information Technology Faculty

ICT Research Institute

Tehran, Iran

farhoodi@itrc.ac.ir

Maryam Mahmoudi

Information Technology Faculty

ICT Research Institute

Tehran, Iran

mahmoudy@itrc.ac.ir

Seyed Mohammadreza Mirsarraf

Information Technology Faculty

ICT Research Institute

Tehran, Iran

mirsaraf@itrc.ac.ir

Abstract— During the last few years, rumor and its rapid diffusion via social media have affected public opinions, even in some important such as presidential elections. One of the main approaches for rumor detection methods is based on content and natural language processing. Despite considerable improvement made in this regard in English language, unfortunately, we have not witnessed enough progress in Persian language, mainly due to a lack of datasets in this area. The main novelty of this paper is combining different learning methods to achieve a more performance outcome in comparison with each method. The methods we used in this study include three classifiers based on lingual-based features, word frequency-based, and word embedding-based features. The results show that combined different methods using the weighted majority fusion method, a significant improvement in the results is achieved.

Keywords- Rumor detection, Machine learning, Content-based text classification, Deep learning, Multi-classification

# Introduction

Social media have many advantages and disadvantages. The advantages include low cost, easy access, and rapid information propagation; however, one of the main disadvantages is the possibility of widespread fake news and rumors [1]. Fake news disseminations in online media and social networks have caused erosions to democracy, justice, and public trust; therefore, we face increased demand for fake news and/or rumor detection and intervention. [2].

When a social media user receives a rumor, his opinion may be affected depending on various factors. These factors include the user’s trust in the sender, the number of times he has received it, the social network structure that affects links and segregations [3, 4], and psychological parameters discussed as the opinion formation models, such as the social impact model of opinion formation [5] and Deffuant model of opinion formation [6]. The emotional aspects of the news also may affect the users’ opinions [7]. A viral rumor on social networks may change the opinion of the majority of the society, or in terms of the physicians, it may cause a phase transition [8, 9]. Therefore, a planned opinion phase transition on social media may manipulate and alter public opinion. A recently published study [10] shows that 28 countries have had organized social media manipulation campaigns in 2017, which increased to 48 countries in 2018, and to 70 countries in 2019, mainly using Facebook and Twitter. Among the public opinion manipulations are the 2020 US presidential election [11], the 2019 Portuguese election [12], the 2017 French presidential election [13], the 2016 US presidential election [14-16], and the 2016 UK European Union membership referendum (Brexit) [17].

Detecting rumors and controlling their impacts on society is essential to achieve a more trustable and purified cyber society. Due to the large number of posts exchanged on online social media, using automatic methods to (even roughly) detection of rumors is inevitable. Recently, natural language processing (NLP), machine learning, and social network analysis have been widely used for rumor detection [1, 2, 18].

In this research, we used three classification methods based on 1) lingual-based features, 2) word frequency-based features (Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF)), and 3) word embedding-based features. The results of these classifiers are sometimes different, and some samples are misclassified by some classifiers due to a variety of reasons, such as the use of different features and different parameters to adjust the algorithm. Therefore, as the main contribution of in this study, we used Multi-classification and fusing the classifiers’ results to enhance the overall performance of the proposed multi-classification method.

The rest of this paper is structured as follows: Section 2 introduces some related works, Section 3 explains the proposed method and related basic concepts, Section 4 describes the datasets we used, Section 5 reports the results of the experiments, Section 6 is dedicated to the discussion, and finally Section 7 concludes the paper.

# Related Works

Recently, several studies have been conducted on rumor detection in English and some other languages using various approaches. Some studies have used various lingual features of the news text for rumor detection, as summarized in Table I. However, unfortunately, few studies have been conducted on Persian rumor detection, as among the studies listed in Table I, just one study has focused on Persian news [19], and others have studied English news. Our research focuses on Persian news using the lingual feature-based approach; therefore, our research is also shown in the table, specifying which lingual features we have used.

One of the problems in studies in this field on Persian language is the lack of sufficient datasets. In [20], to solve the lack of Persian datasets problem, a machine translator has been used to translate English tweets into Persian. However, this approach imposes an error on the system. In [21], a Persian rumor detection on Twitter based on context-based features has been reported, and [19] analyzed the content of the original rumor and introduced informative content features to early identifying Persian rumors on Twitter and Telegram, i.e., when it is published on news media but has not yet spread on social media.

No study is reported for Persian rumor detection using N-gram and TF-IDF; however, some studies are reported on other languages. In [31], N-gram and TF-IDF have been used for rumor detection. Wynne et al. [32] have used N-gram and TF-IDF for rumor detection. Oriola has also used some content-based methods for rumor detection, including N-gram and TF-IDF; we are interested in this study.

Few studies have also been conducted on Persian rumor detection using deep learning. Samadi et al. [33] have proposed two different architectures for detecting rumor using the BERT pre-trained model. Jahanbakhsh-Nagadeh et al. [34, 35] have used semantic features and ParsBERT [36], a monolingual BERT for the Persian language, for Persian rumor verification. Mottaghi et al. [37] have used a convolutional neural network (CNN) model for their deep learning approach to detect Persian rumor. Ghayoome et al. [38] have developed a deep cross-lingual contextualized language model for fake news detection. Sadr et al. [39] have used a hybrid of LSTM and bidirectional LSTM (BLSTM) for Persian fake news detection.

Ensembling classifiers have been used in various applications [40]. However, few studies have focused on ensembling rumor classifiers [41].

# Methods

This section describes the methods we used in this research.

## Classifiers Preparation

This research recruits various classifiers and their fusion. Therefore, we first prepared each classifier according to the general two-phased process shown in Fig. 1. As the figure shows, in phase 1 (learning phase), the classifier model is trained with the labeled dataset. Labels indicate whether each post is rumor or not. The learning process depends on the classifier model. After the learning phase, the classifier is ready to predict or label the input samples.

## The Multi-classifier Architecture

Our experiment is conducted based on a simple multi-classifier architecture shown in Fig. 2. This figure shows the components of our proposed multi-classifier architecture composed of “data cleaning/ preprocessing”, the classifiers, “fuser”, and “evaluation” components. The figure also shows the data flow between the components. The following sections explain the components of this architecture.

## Data cleaning/ preprocessing

The “data cleaning/ preprocessing” component prepares the input dataset to be processed in the next steps. This component normalizes the posts by removing unnecessary characters, unifying some Persian characters with more than one form, and unifying the various forms of typewriting, which is one of the Persian typewriting challenges.



1. General two phased learning prediction of classifiers

Furthermore, this component extracts some information such as words stems, segmentation, and sentence splitting. For using the classification methods based on N-gram, TF-IDF, and LSTM, stop words and punctuation marks are also removed from the input dataset, whereas the stop words remain for lingual feature based classifier because some of the features are extracted from the stop words and punctuation marks. This component uses Hazm , a free Python library for Persian language processing based on NLTK library. The learned classifiers receive preprocessed posts from pre-processed dataset without labels and label each post to send to the “fuser” component.

1. Lingual-based features and related works

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Attribute Type | Feature | [22] | [23] | [24] | [25] | [26] | [27] | [28] | [29] | [30] | [19] | Our Study |
| Quantity | Number of characters |  |  | ✓ |  |  |  |  |  | ✓ | ✓ | ✓ |
| Number of words | ✓ | ✓ | ✓ | ✓ | ✓ |  |  |  | ✓ | ✓ | ✓ |
| Number of noun phrases | ✓ |  |  |  |  |  |  |  |  |  |  |
| Number of sentences | ✓ | ✓ | ✓ | ✓ |  |  |  |  | ✓ | ✓ | ✓ |
| Number of paragraphs |  |  |  |  |  |  | ✓ |  | ✓ |  |  |
| Complexity | Average number of characters per word  | ✓ | ✓ | ✓ | ✓ |  |  |  |  | ✓ |  | ✓ |
| Average number of words per sentence  | ✓ | ✓ | ✓ | ✓ | ✓ |  |  |  | ✓ |  | ✓ |
| Average number of clauses per sentence  | ✓ |  |  | ✓ |  |  |  |  |  |  |  |
| Average number of punctuations per sentence  | ✓ | ✓ | ✓ | ✓ |  |  |  |  |  | ✓ | ✓ |
| Average number of Name Entity per sentence |  |  |  |  |  |  |  |  |  | ✓ |  |
| Uncertainty | #/% Modal verbs (e.g., “shall”)  | ✓ | ✓ | ✓ | ✓ |  |  |  |  |  |  | ✓ |
| #/% Certainty terms (e.g., “never” and “always”)  | ✓ | ✓ | ✓ | ✓ |  | ✓ |  |  | ✓ | ✓ | ✓ |
| #/% Generalizing terms (e.g., “generally” and “all”)  |  | ✓ |  |  |  |  |  |  |  |  | ✓ |
| #/% Tentative terms (e.g., “probably”)  |  | ✓ | ✓ |  |  | ✓ |  |  | ✓ | ✓ | ✓ |
| #/% Numbers and quantifiers  |  |  | ✓ |  |  |  |  |  |  | ✓ | ✓ |
| #/% Question marks  |  |  | ✓ |  |  |  | ✓ |  | ✓ | ✓ | ✓ |
| #/% Inferential words/phrase (e.g., “as a result”) |  |  |  |  |  |  |  |  |  | ✓ | ✓ |
| Subjectivity | #/% Biased lexicons (e.g., “attack”)  |  |  |  |  |  |  |  |  | ✓ |  |  |
| #/% Subjective verbs (e.g., “feel” and “believe”)  | ✓ |  |  |  | ✓ |  |  |  |  | ✓ | ✓ |
| #/% Report verbs (e.g., “announce”)  |  |  |  |  |  |  |  |  | ✓ |  | ✓ |
| #/% Factive verbs (e.g., “observe”)  |  |  |  |  |  |  |  |  | ✓ |  |  |
| #/% Motion verbs (e.g., “fall”, “shake”)  |  |  |  |  |  |  |  |  |  | ✓ |  |
| Non-immediacy | #/% Passive voice  | ✓ | ✓ |  | ✓ |  |  |  |  |  |  |  |
| #/% Self reference: 1st person singular pronouns  | ✓ | ✓ | ✓ | ✓ | ✓ |  |  |  |  |  | ✓ |
| #/% Group reference: 1st person plural pronouns  | ✓ | ✓ | ✓ | ✓ | ✓ |  |  |  |  | ✓ | ✓ |
| #/% Other reference: 2nd and 3rd person pronouns  | ✓ | ✓ | ✓ |  | ✓ |  |  |  |  | ✓ | ✓ |
| #/% Quotations |  |  | ✓ |  |  |  |  |  |  |  |  |
| Sentiment | #/% Positive words  | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |  |  | ✓ | ✓ | ✓ |
| #/% Negative words  | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |  |  | ✓ | ✓ | ✓ |
| #/% Anxiety/angry/sadness words  |  |  |  |  |  | ✓ |  |  | ✓ | ✓ | ✓ |
| #/% Exclamation marks  |  |  | ✓ |  |  |  |  |  | ✓ |  |  |
| Content sentiment polarity  |  |  |  |  |  |  |  |  | ✓ | ✓ |  |
| Emotiveness (The ratio of the sum of adjectives, adverbs and sensory/motion verbs to total words) |  |  |  |  |  |  |  |  |  | ✓ |  |
| Newsworthy (text-enhancing components) |  |  |  |  |  |  |  |  |  | ✓ |  |
| Diversity | Lexical diversity: #/% unique words or terms  | ✓ | ✓ | ✓ | ✓ | ✓ |  |  |  | ✓ |  |  |
| Content word diversity: #/% unique content words  | ✓ | ✓ |  |  | ✓ |  |  |  | ✓ |  |  |
| Redundancy: #/% unique function words  | ✓ | ✓ | ✓ |  | ✓ |  |  |  | ✓ |  |  |
| #/% Unique nouns/verbs/adjectives/adverbs  |  |  |  |  |  |  |  |  | ✓ | ✓\* | ✓\* |
| Informality | #/% Typos (misspelled words)  | ✓ |  |  | ✓ | ✓ |  |  |  |  | ✓ |  |
| #/% Swear words/netspeak/assent/non fluencies/fillers  |  |  |  |  |  |  |  |  | ✓ | ✓ | ✓ |
| Start/End phrase (e.g., “Urgent!”, “Please Share!”) |  |  |  |  |  |  |  |  |  | ✓ | ✓ |
| Consecutive letter/Word (e.g., “very very important!”) |  |  |  |  |  |  |  |  |  | ✓ | ✓ |
| Emoji |  |  |  |  |  |  |  |  |  | ✓ |  |
| Specificity | Temporal/spatial ratio  | ✓ | ✓ |  |  |  | ✓ |  |  |  | ✓ |  |
| Sensory ratio  | ✓ | ✓ |  | ✓ |  | ✓ |  |  | ✓ |  |  |
| Causation terms  |  | ✓ |  |  |  | ✓ |  |  | ✓ |  |  |
| Exclusive terms  |  | ✓ |  |  |  |  |  |  |  |  |  |
| Readablity (e.g., Flesch-Kincaid and Gunning-Fog index) |  |  | ✓ |  |  |  | ✓ | ✓ | ✓ | ✓ |  |

\*: The average of nouns/ verbs/ adjectives/ adverbs



1. Architecture of the experiment

## Lingual features based classification

The prevailing way of characterizing and detecting rumors and fake news based on lingual features relies on the lingual features in various language levels: lexicon, syntax, discourse, and semantics [2]. In this technique, the following feature groups are mostly used:

* Linguistic features: related to the characters, words, and sentences of the post, as well as part of speech (POS) of the words and phrases.
* Psycho-linguistic features: dealing with the sentimental analysis of the post, extracting positive, negative, and neutral sentiments of the sentences of the post.
* Stylometric features: related to writing style, including the percentage of the numbers and the particles of whole the post, punctuation marks, long words, short words, and some other similar features.

This component extracts the features using text processing modules, and normalizes the extracted statistical features to the range [0..1] using min-max normalization method. The feature-set we used in this method are presented in the column “Our Study” of Table I.

In this component, we used different supervised algorithms and compared the results to choose the best one. Table II and Table III show the algorithms we considered for this component and the results obtained from each one on both datasets, respectively. Overally, as the tables show, the Random Forest (RF) algorithm performs better than other algorithms; thus, in the rest of the paper, we report just the results of RF algorithm. We used the free scikit-learn library in python to implement the classification methods.

## N-gram and TF-IDF based classifiers

N-gram and TF-IDF based classifiers have similar fundamentals to the bag of words models. From a content viewpoint, documents have a similar classification result if they have similar content. In addition, much can be learned from the content alone about the document content. The first step in a bag of word implementation is vocabulary management. The length of the document vector is equal to the number of known words. Each document may contain a small number of known words in the vocabulary. These results in a vector with a high number of zeros, called a scattered vector or scattered representation. The scattered vectors require more memory and computational resources when modeling, and a large number of positions or dimensions can make the modeling process very challenging for traditional algorithms. Thus, when using a bag of word model, it is necessary to reduce the size of the words. A simple text cleanning that can be used as the first step, including a) ignoring punctuation, b) ignoring stop words, c) misspelling correction, and d) replacing words to their stems using a stemmer.

After selecting the words, the occurrence of the words in the sample documents should be scored. In this study, two scorings are used:

* word frequency for N-gram based classification
* TF-IDF for TF-IDF based classification.
1. Results of various machine learning algorithms on Sepehr\_RumTel01 Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Accuracy | F-Measure | Recall | Precision | Algorithm |
| 0.80 | 0.71 | 0.69 | **0.85** | SVM  |
| 0.80 | 0.73 | 0.71 | 0.84 | Log-Regression |
| 0.83 | **0.81** | **0.80** | 0.84 | Random Forest  |
| 0.77 | 0.70 | 0.69 | 0.76 | Naïve Base |
| 0.79 | 0.75 | 0.73 | 0.81 | KNN |
| 0.77 | 0.74 | 0.73 | 0.75 | Decision Tree |
| **0.84** | 0.79 | 0.77 | **0.85** | LDA (Linear Discriminant Analysis) |

1. Results of various machine learning algorithms on KNTUPT Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Accuracy | F-Measure | Recall | Precision | Algorithm |
| 0.81 | 0.45 | 0.50 | 0.41 | SVM  |
| 0.82 | 0.45 | 0.50 | 0.54 | Log-Regression |
| **0.94** | **0.88** | **0.85** | **0.93** | Random Forest  |
| 0.44 | 0.43 | 0.64 | 0.60 | Naïve Base |
| 0.91 | 0.84 | 0.82 | 0.86 | KNN |
| 0.90 | 0.83 | 0.83 | 0.82 | Decision Tree |
| 0.82 | 0.48 | 0.51 | 0.63 | LDA (Linear Discriminant Analysis) |

For N-gram classifications, we examined uni-gram and bi-gram and the results of uni-gram was better than bi-gram. Thus, our implementation of N-gram classification is indeed based on uni-gram.

Fig. 3 shows the general architecture for bag of words classifications we used in this study. Although we used SVM algorithm as the classification algorithm for our bag of words methods due to its better performance, other classification algorithms could also be used instead.

## Classification based on deep learning

Deep learning is a subcategory of machine learning, a neural network with more than two layers of hidden units or neurons. The deep networks are deep in terms of the number of neuron layers in the network. Generally, deep learning displays relatively high precision and exactness in rumor detection [21]; however, it uses more memory [22].

Long short-term memory (LSTM), a widely used deep learning architecture, is a neural network framework based on the recurrent neural network (RNN). LSTM aims to deal with the vanishing gradient problem present in traditional RNNs. Thus, LSTM is a special kind of recurrent neural network capable of learning long term dependencies in data. This is achieved because the recurring module of the model has a combination of four layers interacting with each other. Since back propagation in RNN takes a while, as a progressed variation of RNN, LSTM overcomes this limitation of traditional RNN with their property of remembering "Short Term Memories" for "Long periods." [23]. Comparing to the RNN neurons, which have two gates, input and output gates, LSTM neurons have an additional gate, forget gate, in the hidden layers. Forget gate makes LSTM capable of having the property of memorization, as shown in Fig. 4.

A common LSTM unit remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. Forget gates decide what information to discard from a previous state by assigning a previous state, compared to a current input, a value between 0 and 1. A (rounded) value of 1 means to keep the information, and a value of 0 means to discard it. Input gates decide which pieces of new information to store in the current state, using the same system as forget gates. Output gates control which pieces of information in the current state to output by assigning a value from 0 to 1 to the information, considering the previous and current states. Selectively outputting relevant information from the current state allows the LSTM network to maintain useful, long-term dependencies to make predictions, both in current and future time-steps.

For implementing LSTM method, we have used the Tensorflow and Keras layer library in Python. Because of variable post sizes of data set, we fixed the size to the average of post sizes plus their standard deviation (146 and 25 for Sepehr\_RumTel01 and KNTUPT datasets, respectively). The shorter input posts were filled with paddings, and the longer input posts were truncated. We have used Word2Vec embedding layer Persian FastText with the dimension of 300 for each input word.



1. Rumor detection using bag of words algorithm



1. Basic structure of the LSTM model

We used bidirectional LSTM with output size of 32. To train the model, we divided the data set to train and test sequence. The size of train sequence was 0.2 of dataset and the size of test sequence was 0.8 of dataset. The batch size of training for LSTM was 256 and the number of epoch was 20. Block diagram of LSTM rumor detector is shown in Fig. 5. As the figure shows, the word presentation in the input of LSTM is Fast text word embedding, developed by Facebook on more than 157 languages, including Persian.

BERT could also be used in this component, similar to [42], in which a pertained BERT base decoder for text feature extractor has been used. The padded and tokenized text is passed into the BERT model to receive word vectors of dimension 768 then they have used LSTM as we did on the top of Word2Vec like our method.

## Fast text embedding algorithm

Word embedding is a way to convert textual information into numeric form, which in turn can be used as input to machine learning algorithms. One major draw-back for word-embedding techniques like Word2Vec was its inability to deal with out of corpus words. These embedding techniques treat words as the minimal entities and try to learn their respective embedding vector. Therefore, if a word does not appear in the corpus, Word2Vec fails to get its vectorized representation. However, FastText follows the same skipgram and CBoW (Continuous Bag of Words) model like Word2Vec. FastText is a modified version of Word2Vec, treating each word as composed of N-grams. In Word2Vec each word is represented as a bag of words, but in FastText each word is represented as a bag of character N-gram. This result in better representation for morphological languages like Persian; thus, we used this embedding algorithm as the input of LSTM.



1. Block diagram of LSTM

## Fusion

Supervised classifiers are trained on training datasets and usually are tested on input data with similar patterns to the training data. Moreover, various classifiers have different performances on test data, depending on both the inherent nature of the classifier and the pattern of the input test data. Therefore, the performance of various classifiers may differ on a particular input data. In such a case, as shown in Fig. 6, ensembling the various classifiers to achieve a fused result could enhance the performance [24].

Several fusion methods could be used for ensembling classifiers, mainly depending on the nature of classifiers' output, e.g., probabilistic, ranking, multilabel or binary. For the case of binary classifiers similar to this research, the foremost fusion method is “weighted majority voting”.

The weighted voting assigns a weight *wj* to each of *N* classifiers, *C1* .. *CN*, and the final label *y* is calculated as:

,

where, *A* is the set of unique class labels, *1*..*m*, and is the characteristic function .

Designing a combined classifier requires special care in the choice of individual classifiers to achieve higher classification performance and have more robust algorithms. A multi-classifier usually consists of two parts:

* A set of individual classifiers
* The method of selection or combination for the final classification.



1. Ensemble classifiers

To achieve a high performance fusion output, in addition to high performance individual classifiers, their diversity is significant. The diversity implies that classifiers consider the subject of classification from various viewpoints or feature sets for classification. In our method, the four rumor classifiers focus on the input posts from two viewpoints. On the one hand, the N-gram, TF-IDF, and LSTM classifiers deal with the words in the posts and consider them as the features for classification; on the other hand, the lingual feature based classifier focuses on the lingual features, the selected features of writing style.

The classifiers in this study are binary classifiers predict whether a post is “rumor” or not. The fusion method we used for these binary labeled classifiers is the weighted majority voting, whose weights are F1-score. The F1-score is an overall classification performance metric, a harmonic mean of the precision and recall metrics. More details of the mentioned metrics will be described in Section “5. Results”.

## Evaluation

The evaluation module of the architecture receives the predictions on every post generated by each trained classifier separately and also from fuser module. Then, it calculates and reports the performance evaluation metrics for each classifier module and fuser module to compare.

# Datasets

In this study, two data sets were used for evaluating rumor detection classification methods, including Sepehr\_RumTel01 [43, 44] and KNTUPT [45].

## Sepehr\_RumTel01 Dataset

The Sepehr\_RumTel01 dataset [43, 44] is taken from the telegram channels of three Iranian websites, Gomaneh (Gomaneh.com), WikiHoax (wikihoax.org), and Anti-Rumor (Shayeaat.ir). This dataset contains 1911 news, comprising 680 rumor and 1231 truthful news.

## KNTUPT Dataset

The KNTUPT dataset [45] is collected from Twitter. It includes 3593704 tweets that were published in the period from November 24 to December 8, 2017. The tweets are based on 60 news about Kermanshah earthquake, in the west of Iran, with a magnitude of 6.3. The new set is mentioned on the Shayeaat website, a fact-checking website. In this dataset, 4343 out of 3593703 news (about one percent) are rumor. Since it was very unbalanced, we extracted all the rumors and randomly selected 2\*4343=8686 real news; therefore, the total size of this dataset was reduced to 13029 for our experience, one-third for the rumors and two-third for the real news.

# Results

For each classification method mentioned in Section 3, we trained a model from labeled data and used it to classify new (unseen) posts whether it is “rumor” or not, using 10-fold stratified cross validation. By 10-fold cross validation, the dataset is partitioned into 10 folds with (roughly) the same number of samples; then, one fold is kept for test, and the other nine folds are used for training the model. After repeating this process for ten times, every sample is chosen once for the test, and the metrics could be measured for all samples. The stratified version of 10-fold cross validation guarantees that train and test folds in each repetition contain (roughly) the same proportion of class labels.

To evaluate the performance classification methods, we used the following commonly used metrics:

* Accuracy: the percentage of correct predictions for the test data, calculated by dividing the number of correct predictions by the number of total predictions.
* Precision: the fraction of relevant examples (true positives) among all of the examples which were predicted to belong in a certain class.
* Recall: the fraction of examples that were predicted to belong to a class with respect to all of the examples that truly belong in the class.
* F-Measure: The adjusted F-Measure allows us to weight precision or recall more highly if it is more important for our use case. Its formula is slightly different:

F1 (when *β*=1) is the harmonic mean of precision and recall we used in this study.

Since we are interested in the classifier’s performance in detecting rumors, we report the measured metrics on the “rumor” class. Table IV and Table V show the classification methods’ results on Sepehr\_RumTel01 and KNTUPT datasets, respectively. The measured values have been rounded to two decimal points. The bar charts of Fig. 7 and Fig. 8 also visualize the same evaluation results of Table IV and Table V, respectively.

1. Evaluation Metrics of Rumor Detection for Classifiers on Sepehr\_RumTel01 Dataset with 680 Rumors among 1231 Posts

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Algorithm** | **Feature** | **P** | **R** | **F1** | **Acc.** |
| **Lingual Features-based** | **RF** | **As specified in Table I (“Our Study” column)** | 0.83 | 0.81 | 0.80 | 0.84 |
| **Word Frequency-based** | **SVM** | **BoW (N-gram)** | 0.89 | 0.78 | **0.83** | 0.89 |
| **TF-IDF** | 0.89 | 0.51 | 0.65 | 0.80 |
| **Word Embeding-based** | **LSTM** | **Fast-Text** | 0.78 | **0.85** | 0.81 | 0.86 |
| **WMV\*** | **0.91** | 0.77 | **0.83** | **0.89** |
| \*: Weighted Majority Voting |  |  |

1. Evaluation Metrics of Rumor detection for Classifiers on KNTUPT dataset with 4343 Romors among 8686 Posts

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Algorithm** | **Feature** | **P** | **R** | **F1** | **Acc.** |
| **Lingual Features-based** | **RF** | **As specified in Table I (“Our Study” column)** | 0.93 | 0.85 | 0.88 | 0.94 |
| **Word Frequency-based** | **SVM** | **BoW (N-gram)** | **0.99** | **0.98** | **0.98** | **0.99** |
| **TF-IDF** | 0.94 | 0.92 | 0.93 | 0.96 |
| **Word Embeding-based** | **LSTM** | **Fast-Text** | 0.91 | 0.93 | 0.92 | 0.95 |
| **WMV\*** | **0.98** | 0.96 | 0.97 | 0.98 |
| \*: Weighted Majority Voting |  |  |

1. Measured metrics of classifier methods on Sepehr\_RumTel01 dataset
2. Measured metrics of classifier methods on KNTUPT dataset

# DISCUSSION

The main goal of this study is fusing various Persian rumor classification methods considering post content. Before considering the fusion results, the measured metrics results from methods separately (Table IV, Table V, Fig. 7, and Fig. 8) show that the methods dealing with the words of the news as features perform better than classification based on lingual features. Moreover, the N-gram content-based method outperforms other content-based methods. However, the lingual feature based method, with lower performance than other methods, is time efficient because it deals with a few features determined by the user, but the other three methods (N-gram, TF-IDF, and LSTM) deal with the words as features; therefore, are very time consuming both for training and for testing. Furthermore, the LSTM needs a large memory volume, mainly for loading vectorized words to process them.

Comparing the results for both datasets also shows that the measured metrics for KNTUPT dataset outperform the metrics for Sepehr\_RumTel01 dataset overall. The nature of both datasets could justify this. The KNTUPT posts are derived from 60 main news [45]; therefore, a post may have appeared several times with some repeating words. Thus, as expected, the methods concerning the content of news perform better than the lingual feature based method on the dataset with more similar posts. On the other hand, the lingual feature based method performs relatively the same on both datasets due to its concentration on the writing style features, not the words or the meaning carried by the posts. Thus, it is expected that the a rumor whose content has previously been recognized are more effectively classified by the content-based methods, while the lingual feature based method performs better in classifying a post encountering for the first time.

The weighted majority voting fusion method has achieved better metrics on Sepehr\_RumTel01 than each of the classification methods solely (except than recall measure for the N-gram method with just one percent difference). However, for the KNTUPT dataset, the measured metrics for the N-gram method perform near the ideal performance, and the fused method does not achieve better performance for any of the metrics than the performance of the N-gram method. However, the performance of the fused method using the weighted majority voting (considering F1-score of each classifier as its weight) is better than (or generally speaking, about) each method (Lingual, TF-IDF, and LSTM).

# CONCLUSION

The widespread of rumors on social media may affect society’s opinion; therefore, various methods for rumor detection have been proposed in the literature. In this study, we implemented and experimented three classification methods, including lingual feature-based, word frequency-based (BoW and TF-IDF), and word embedding-based methods. Then, we proposed a multi-classifier model to fuse the results of the four classifiers using the weighted majority voting method whose weights are proportional to the classifiers’ F1-score values.

Regarding the nature of the datasets we used and the results of applying the classification methods on two datasets, we can conclude that:

* For the previously published posts with the same or similar contents, which are in training set for the classifier, the content-based classifiers N-gram, TF-IDF, and LSTM outperform the lingual feature based classifier, and among these three classifiers, N-gram performs better than the others.
* The lingual feature-based classifier has roughly the same performance on the posts, regardless whether the posts (or posts similar to it) has been seen before or not.
* The fusion of the classification results, especially the fusion method based on the weighted majority voting method whose weights are proportional to the F1-score of individual classifiers, could increase the performance of classification, especially in the case of classifying not previously seen news. Therefore, the proposed multi-classifier architecture is recommended to process the posts from various content viewpoints for the newly published rumor, which is more important to be timely detected.
* The training and testing time of the N-gram, TF-IDF, and LSTM is the main deficiency of these classification methods compared with the lingual feature based classifier. Furthermore, N-gram, TF-IDF, and LSTM classifiers suffer from high memory usage, especially for LSTM, which requires a large amount of memory for vectorized words.

The main limitation of this research is the lack of enough rumor datasets in Persian, both in the number of datasets and the volume of data in available datasets. Therefore, the results of this study could be more verified by applying our method to more datasets. Thus, one of the future works very helpful for this study and similar studies is developing Persian rumor datasets.

This study could also be extended from other viewpoints. We considered classification methods concerning contents of the posts, both lingual features and semantic features, while other viewpoints of rumor could also be considered, including propagation, news source, and temporal features, as well as automatic fact checking. The fusion of classifiers’ results from various viewpoints may result in more comprehensive and accurate results.

##### References

1. K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD explorations newsletter,* vol. 19, pp. 22-36, 2017.
2. X. Zhou and R. Zafarani, "A survey of fake news: Fundamental theories, detection methods, and opportunities," *ACM Computing Surveys (CSUR),* vol. 53, pp. 1-40, 2020.
3. A. Mansouri and F. Taghiyareh, "Effect of Segregation on Opinion Formation in Scale-Free Social Networks: An Agent-based Approach," *International Journal of Engineering,* vol. 34, pp. 66-74, 2021.
4. A. Mansouri and F. Taghiyareh, "Effect of segregation on the dynamics of noise-free social impact model of opinion formation through agent-based modeling," *International Journal of Web Research,* vol. 2, pp. 36-44, 2019.
5. J. A. Hołyst, K. Kacperski, and F. Schweitzer, "Social impact models of opinion dynamics," *Annual Reviews Of Computational PhysicsIX,* pp. 253-273, 2001.
6. G. Deffuant, D. Neau, F. Amblard, and G. Weisbuch, "Mixing beliefs among interacting agents," *Advances in Complex Systems,* p. 11, 2001.
7. A. Mansouri and F. Taghiyareh, "Toward an emotional opinion formation model through agent-based modeling," in *2017 7th International Conference on Computer and Knowledge Engineering (ICCKE)*, 2017, pp. 14-19.
8. A. Mansouri and F. Taghiyareh, "Phase transition in the social impact model of opinion formation in scale-free networks: The social power effect," *Journal of Artificial Societies and Social Simulation,* vol. 23, 2020.
9. A. Mansouri and F. Taghiyareh, "Phase Transition in the Social Impact Model of Opinion Formation in Log-Normal Networks," *Journal of Information Systems and Telecommunication (JIST),* vol. 1, p. 1, 2021.
10. S. Bradshaw and P. N. Howard, "The global disinformation order: 2019 global inventory of organised social media manipulation," 2019.
11. E. Chen, H. Chang, A. Rao, K. Lerman, G. Cowan, and E. Ferrara, "COVID-19 misinformation and the 2020 US presidential election," *The Harvard Kennedy School Misinformation Review,* 2021.
12. J. P. Baptista and A. Gradim, "Online disinformation on Facebook: the spread of fake news during the Portuguese 2019 election," *Journal of Contemporary European Studies,* pp. 1-16, 2020.
13. S. T. Smith, E. K. Kao, E. D. Mackin, D. C. Shah, O. Simek, and D. B. Rubin, "Automatic detection of influential actors in disinformation networks," *Proceedings of the National Academy of Sciences,* vol. 118, 2021.
14. N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer, "Fake news on Twitter during the 2016 US presidential election," *Science,* vol. 363, pp. 374-378, 2019.
15. A. Bovet and H. A. Makse, "Influence of fake news in Twitter during the 2016 US presidential election," *Nature communications,* vol. 10, pp. 1-14, 2019.
16. H. Allcott and M. Gentzkow, "Social media and fake news in the 2016 election," *Journal of economic perspectives,* vol. 31, pp. 211-36, 2017.
17. M. T. Bastos and D. Mercea, "The Brexit botnet and user-generated hyperpartisan news," *Social science computer review,* vol. 37, pp. 38-54, 2019.
18. A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter, "Detection and resolution of rumours in social media: A survey," *ACM Computing Surveys (CSUR),* vol. 51, pp. 1-36, 2018.
19. Z. Jahanbakhsh-Nagadeh, M.-R. Feizi-Derakhshi, and A. Sharifi, "A Model for Detecting of Persian Rumors based on the Analysis of Contextual Features in the Content of Social Networks," *Signal and Data Processing,* vol. 18, pp. 50-29, 2021.
20. M. H. Saghayan, S. F. Ebrahimi, and M. Bahrani, "Exploring the Impact of Machine Translation on Fake News Detection: A Case Study on Persian Tweets about COVID-19," in *2021 29th Iranian Conference on Electrical Engineering (ICEE)*, 2021, pp. 540-544.
21. S. Zamani, M. Asadpour, and D. Moazzami, "Rumor detection for persian tweets," in *2017 Iranian Conference on Electrical Engineering (ICEE)*, 2017, pp. 1532-1536.
22. L. Zhou, J. K. Burgoon, J. F. Nunamaker, and D. Twitchell, "Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications," *Group decision and negotiation,* vol. 13, pp. 81-106, 2004.
23. C. M. Fuller, D. P. Biros, and R. L. Wilson, "Decision support for determining veracity via linguistic-based cues," *Decision Support Systems,* vol. 46, pp. 695-703, 2009.
24. S. Afroz, M. Brennan, and R. Greenstadt, "Detecting hoaxes, frauds, and deception in writing style online," in *2012 IEEE Symposium on Security and Privacy*, 2012, pp. 461-475.
25. M. Siering, J.-A. Koch, and A. V. Deokar, "Detecting fraudulent behavior on crowdfunding platforms: The role of linguistic and content-based cues in static and dynamic contexts," *Journal of Management Information Systems,* vol. 33, pp. 421-455, 2016.
26. D. Zhang, L. Zhou, J. L. Kehoe, and I. Y. Kilic, "What online reviewer behaviors really matter? Effects of verbal and nonverbal behaviors on detection of fake online reviews," *Journal of Management Information Systems,* vol. 33, pp. 456-481, 2016.
27. G. D. Bond, R. D. Holman, J. A. L. Eggert, L. F. Speller, O. N. Garcia, S. C. Mejia*, et al.*, "‘Lyin'Ted’,‘Crooked Hillary’, and ‘Deceptive Donald’: Language of Lies in the 2016 US Presidential Debates," *Applied Cognitive Psychology,* vol. 31, pp. 668-677, 2017.
28. M. Potthast, J. Kiesel, K. Reinartz, J. Bevendorff, and B. Stein, "A stylometric inquiry into hyperpartisan and fake news," *arXiv preprint arXiv:1702.05638,* 2017.
29. V. Pérez-Rosas, B. Kleinberg, A. Lefevre, and R. Mihalcea, "Automatic detection of fake news," *arXiv preprint arXiv:1708.07104,* 2017.
30. X. Zhou, A. Jain, V. V. Phoha, and R. Zafarani, "Fake news early detection: A theory-driven model," *Digital Threats: Research and Practice,* vol. 1, pp. 1-25, 2020.
31. B. Al Asaad and M. Erascu, "A tool for fake news detection," in *2018 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, 2018, pp. 379-386.
32. H. E. Wynne and Z. Z. Wint, "Content based fake news detection using n-gram models," in *Proceedings of the 21st International Conference on Information Integration and Web-based Applications & Services*, 2019, pp. 669-673.
33. M. Samadi, M. Mousavian, and S. Momtazi, "Persian fake news detection: Neural representation and classification at word and text levels," *Transactions on Asian and Low-Resource Language Information Processing,* vol. 21, pp. 1-11, 2021.
34. Z. Jahanbakhsh-Nagadeh, M.-R. Feizi-Derakhshi, and A. Sharifi, "A Deep Content-Based Model for Persian Rumor Verification," *Transactions on Asian and Low-Resource Language Information Processing,* vol. 21, pp. 1-29, 2021.
35. Z. Jahanbakhsh-Nagadeh, M.-R. Feizi-Derakhshi, and A. Sharifi, "A semi-supervised model for Persian rumor verification based on content information," *Multimedia Tools and Applications,* vol. 80, pp. 35267-35295, 2021.
36. M. Farahani, M. Gharachorloo, M. Farahani, and M. Manthouri, "Parsbert: Transformer-based model for persian language understanding," *Neural Processing Letters,* vol. 53, pp. 3831-3847, 2021.
37. V. Mottaghi, M. Esmaeili, G. A. Bazaee, and M. Afshar Kazemi, "A decision-making system for detecting fake persian news by improving deep learning algorithms–case study of Covid-19 news," *Journal of applied research on industrial engineering,* vol. 8, pp. 1-17, 2021.
38. [M. Ghayoomi and M. Mousavian, "Deep transfer learning for COVID‐19 fake news detection in Persian," *Expert Systems,* p. e13008, 2022.
39. M. M. Sadr, "The Use of LSTM Neural Network to Detect Fake News on Persian Twitter," *Turkish Journal of Computer and Mathematics Education (TURCOMAT),* vol. 12, pp. 6658-6668, 2021.
40. A. Rahman and S. Tasnim, "Ensemble classifiers and their applications: A review," *International Journal of Computer Trends and Technology (IJCTT),* vol. 10, pp. 31-35, 2014.
41. L. Hasimi and A. Poniszewska-Marańda, "Ensemble Learning-based Fake News and Disinformation Detection System," in *2021 IEEE International Conference on Services Computing (SCC)*, 2021, pp. 145-153.
42. S. Singhal, A. Kabra, M. Sharma, R. R. Shah, T. Chakraborty, and P. Kumaraguru, "Spotfake+: A multimodal framework for fake news detection via transfer learning (student abstract)," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, pp. 13915-13916.
43. Z. Jahanbakhsh-Nagadeh, M.-R. Feizi-Derakhshi, and A. Sharifi, "A speech act classifier for persian texts and its application in identifying rumors," *Journal of Soft Computing and Information Technology,* vol. 9, pp. 18-27, 2020.
44. A. R. Feizi Derakhshi, M. R. Feizi Derakhshi, M. Ranjbar-Khadivi, N. Nikzad Khasmakhi, M. Ramezani, T. Rahkar Farshi*, et al.*, "“Sepehr\_RumTel01”, Mendeley Data, doi: 10.17632/jw3zwf8rdp," ed, 2020.
45. S. D. Mahmoodabad, S. Farzi, and D. B. Bakhtiarvand, "Persian rumor detection on twitter," in *2018 9th International Symposium on Telecommunications (IST)*, 2018, pp. 597-602.