JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY
http://jict.uum.edu.my

How to cite this article:
Nezhad, Z. B., & Deihimi, A. M. (2021). Sarcasm detection in Persian. *Journal of Information and Communication Technology, 20*(1), 1-20. https://doi.org/10.32890/jict.20.1.2021.6249

SARCASM DETECTION IN PERSIAN

[1]Zahra Bokaee Nezhad & [2]Mohammad Ali Deihimi
[1]Department of Computer Engineering, Zand University, Iran
[2]Department of Electronics Engineering, Bahonar University, Iran

Corresponding author: zbokaee@gmail.com
m.a.deihimi@gmail.com

Received: 25/4/2019 Revised: 13/5/2020 Accepted: 14/5/2020 Published: 4/11/2020

ABSTRACT

Sarcasm is a form of communication where the individual states the opposite of what is implied. Therefore, detecting a sarcastic tone is somewhat complicated due to its ambiguous nature. On the other hand, identification of sarcasm is vital to various natural language processing tasks such as sentiment analysis and text summarisation. However, research on sarcasm detection in Persian is very limited. This paper investigated the sarcasm detection technique on Persian tweets by combining deep learning-based and machine learning-based approaches. Four sets of features that cover different types of sarcasm were proposed, namely deep polarity, sentiment, part of speech, and punctuation features. These features were utilised to classify the tweets as sarcastic and non-sarcastic. In this study, the deep polarity feature was proposed by conducting a sentiment analysis using deep neural network architecture. In addition, to extract the sentiment feature, a Persian sentiment dictionary was developed, which consisted of four sentiment categories. The study also used a new Persian proverb dictionary in the preparation step to enhance the accuracy of the proposed model. The performance of the model is analysed using several standard machine learning algorithms. The results of the experiment showed that the method outperformed the baseline method and reached an accuracy of 80.82%. The study also examined the importance of each proposed feature set and evaluated its added value to the classification.

Keywords: Sarcasm detection, natural language processing, machine learning, sentiment analysis, classification.

INTRODUCTION

Twitter has become one of the biggest destinations for people to put forward their opinions. There is a great opportunity for companies and organisations to notice users' ideas (Rajadesingan, Zafarani, & Liu, 2015). Sentiment analysis has emerged as a field of study in identifying opinionative data in the Web and classifying them according to their polarity. Sentiment analysis can present reasonable opportunities for marketers to generate market intelligence on consumer attitudes and help organisations to fulfil their purposes (Al-Otaibi et al., 2018). Nevertheless, there are some problems with using sentiment analysis tools. One of these problems is the existence of sarcasm in the user's view, which can lead to misclassification of sentiment analysis (Maynard & Greenwood, 2014). In fact, sarcasm is one of the considerable challenges in sentiment analysis. It is an indirect way of telling a message and can be conveyed through different ways such as direct conversation, speech, text, etc. (Seyed Sadeqi & Ehsanjou, 2018). In direct conversation, facial expression and body gesture help to recognise sarcasm. In speech, sarcasm can be derived from changes in tone. In text, it is too difficult to identify sarcasm; however, there are some methods that help to reveal it.

Cambridge Dictionary describes sarcasm as "the use of remarks that mean the opposite of what they say made to hurt some one's feeling or to criticise something humorously!" Consider the following tweet, "Yay! It's a holiday weekend, and I'm on call for work! Couldn't be luckier!" Although this tweet includes the words "yay" and "lucky" with positive sentiments, the expression has a negative sentiment (Liu et al., 2014). This shows that detecting the sentiment of a tweet seems complicated when the tweet is sarcastic. Sarcasm may create problems for not only sentiment analysis approaches but also many other natural language processing (NLP) tasks such as review summarisation systems. Consider the following tweet, "I'm very pleased to waste my four hours on such a pathetic movie!". Although this tweet has the word "pleased" with a positive sentiment, the whole emotion of the tweet is negative. Accordingly, if a movie review summarisation system does not employ a sarcasm detection model, it may recognise this tweet as a positive review (Maynard & Greenwood, 2014). It is demonstrated that the state-of-the-art approaches of sentiment analysis can be highly enhanced when there is an ability to detect sarcastic statements (Hazarika et al., 2018). To compound the problem, in Persian, people tend to use sarcasm in their daily conversations for criticizing and censoring especially in political topics (Hokmi, 2017).To the best of the researchers' knowledge, there is no work on sarcasm detection in Persian. Therefore, they aim to present a model that performs the task of sarcasm detection in Persian. The proposed approach considers different types of sarcasm and it is evaluated on the first sarcastic Persian dataset.

The rest of the article is structured as follows. The next section describes related works. It is followed by the architecture of the proposed model. The results are shown in the fourth section, while the fifth section concludes this work and proposes possible directions for future works.

RELATED WORKS

Currently, despite several studies that have been conducted to detect sarcasm in English, there is no attempt to address this problem in Persian. However, there are several research related to the recognition of sarcasm in different languages and this subject has gained rapid attention. Twitter sarcasm detection techniques can be classified into five categories, namely pattern-based approach, context-based approach, deep learning-based approach, machine learning-based approach, and lexicon-based approach (Bharti et al., 2017).

Pattern-based approach: Bouazizi and Otsuki (2016) used a pattern-based approach to detect sarcasm on Twitter. They utilised three patterns for sarcasm in tweets: (1) sarcasm as *Wit*, which is used by capital letter words, punctuation marks, or sarcasm-related emoticons to show the sense of humour; (2) sarcasm as *Evasion*, which is used when a person wants to avoid giving a clear answer; and (3) sarcasm as *Whimper*, in which the anger of a person is shown. Then, four sets of features were extracted as sentiment-related features, punctuation-related features, syntactic and semantic features, and pattern features. They reached an accuracy of 83.1%.

Context-based approach: Schifanella et al. (2016) built a complex classification model that worked over an entire tweet sequence rather than one tweet at a time. They deployed features based on the integration between linguistic and contextual features.

Deep learning-based approach: Some studies proposed automated sarcasm detection using deep neural network architecture. Son et al. (2019) put forward a hybrid deep learning model based on soft Attention-Based Bidirectional Long Short-Term Memory (sAtt-BLSTM) and convolutional neural network (ConvNet). They applied Global Vector (GloVe) for word representation. Additionally, they used feature maps generated by sAtt-BLSTM as well as punctuation-based features. Felbo et al. (2017) proposed a deep Moji model based on the occurrence of emoji. They used Long Short-Term Memory (LSTM) as well as attention mechanisms and gained acceptable results. Ghosh and Veale (2016) built a model combing a Recurrent Neural Network (RNN) with a ConvNet. Their model represented better results as compared to recursive support vector machine (SVM).

Machine learning-based approach: Many studies have been conducted to detect sarcasm based on machine learning approach. Rajadesingan et al. (2015) proposed a model to detect sarcasm by behavioural features using users' past tweets. They employed theories from behavioural and psychological studies to construct their features. They used Naïve Bayes and SVM classifiers. Blamey et al. (2012) suggested a new feature to capture properties of a figurative language like emotional scenario and unexpectedness with ambiguity and polarity. Suhaimin et al. (2019) presented a framework to support sentiment analysis by using sarcasm detection and classification. The framework comprised six modules: pre-processing, feature extraction, feature selection, initial sentiment classification, sarcasm detection and classification, and actual sentiment classification. The framework was evaluated using a nonlinear SVM and Malay social media data. The best average F-measure score of 90.5% was recorded using their framework. Suhaimin et al. (2017) proposed a feature extraction process to detect sarcasm using Malay social media data as bilingual texts. They considered four categories of features using NLP, namely lexical, pragmatic, prosodic, and syntactic. They investigated the use of the idiosyncratic feature to capture peculiar and odd comments found in the texts. A nonlinear SVM was utilised for classification. Their results demonstrated that the combination of syntactic, pragmatic, and prosodic features produced the best performance with a F-measure score of 85.2%. Lunando and Purwarianti (2013) presented an Indonesian sarcasm detection model by using unigram, number of interjections, words, negativity, and question words as extracted features. For term recognition, they used translated SentiWordnet, which led to undetected terms and very low accuracy. Rahayu et al. (2018) proposed another method on Indonesian tweets based on two extracted features, namely interjection and punctuation. They also used two different weighting algorithms. Their proposed model outperformed the model proposed by Lunando and Purwarianti (2013). Nevertheless, the paucity of sophisticated features on their work leave more room for improvement.

Lexicon-based approach: Parmar et al. (2018) utilised lexical and hyperbole features to improve the sentiment analysis results. They employed MapReduce to reduce the execution time by a parallel process platform. They suggested five parts for their proposed model. In the first part, different Twitter application programming interfaces (APIs) were performed to retrieve tweets. Afterwards, the results were stored in the hadoop distributed file system (HDFS). Secondly, all data were preprocessed to remove noisy data such as Uniform Resource Locator (URL). The third part was words relationships, which were done by part-of-speech tagging (POS) method. Subsequently, all the phrases were stored in a parsed file. As the fourth step, sentiment analysis was conducted on all phrases. The last step was a feature-based composite approach (FBCA). In this step, the algorithm used hyperbole and lexical with punctuation and negation features to detect sarcastic tweets.

Riloff et al. (2013) developed two bags of lexicons using bootstrap techniques. These lexicons consisted of positive sentiments and negative situations. They attempted to identify sarcasm in tweets for any positive sentiment in a negative situation. However, their method had some limitations since it could not identify sarcasm across multiple sentences.

In the present study, the researchers will investigate the sarcasm detection technique on Persian tweets by combining deep learning-based and machine learning-based approaches. They aim to find both world-level and sentence-level inconsistencies. Four different sets of features are extracted to cover different types of sarcasm in Persian. In addition, this study presents the first Persian proverb dictionary to ignore any misclassification, which will be further explained in the next section.

PROPOSED MODEL

The existence of sarcasm in tweets can lead to a state of ambiguity. It can also decrease the accuracy of some NLP tasks such as Sentiment Analysis and Text Summarisation (Bharti et al., 2017). Consider the following tweet, "گوشیم خورد شد، بهتر از این نمیشه!!!" (My phone is broken, how better can this be???). Although it is a sarcastic tweet, without deploying the sarcasm detection model, the sentiment analysis model may mislead. Therefore, it is crucial to develop a model to detect sarcasm.

Figure 1 represents the architecture of the proposed model. The model consists of four key components, namely (1) Data Preprocessing, (2) Data Preparation, (3) Feature Extraction, and (4) Classification. From each tweet, the researchers extracted a set of features in a way that covered different types of sarcasm. Then, they used several machine learning algorithms to perform the classification.

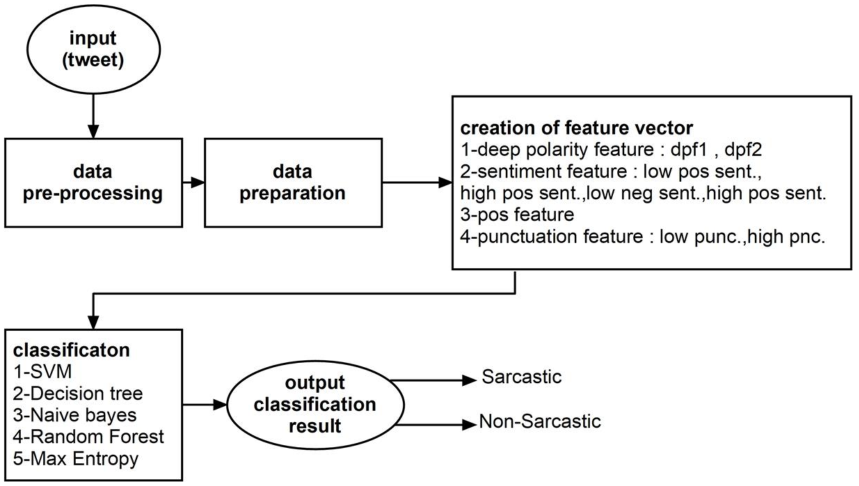
input
(tweet)
data
pre-processing
data
preparation
creation of feature vector
1-deep polarity feature : dpf1 , dpf2
2-sentiment feature : low pos sent.,
high pos sent.,low neg sent.,high pos sent.
3-pos feature
4-punctuation feature : low punc.,high pnc.
classificaton
1-SVM
2-Decision tree
3-Naive bayes
4-Random Forest
5-Max Entropy
output
classification
result
Sarcastic
Non-Sarcastic

Figure 1. *Architecture of Persian sarcasm detection model.*

Data Collection

One problem of NLP tasks in Persian is that there is no standard dataset for sentiment analysis nor sarcasm detection (Gelbukh, 2009). Therefore, the present study created a Persian dataset for sarcasm detection using Tweepy API. To collect sarcastic tweets, the researchers queried the API for tweets containing hashtags #طعنه or #کنایه (both mean sarcasm). Non-sarcastic tweets were retrieved based on political hashtags. Then, these tweets were manually checked and cleaned up by removing noisy and irrelevant tweets. In this step to avoid bias, ten different computer scientists and native speakers were employed for checking labels. Ultimately, the study collected 1,200 sarcastic tweets containing hashtags #طعنه or #کنایه and 1,300 non-sarcastic tweets. In this dataset, sarcastic and non-sarcastic tweets were labelled 0 and 1, respectively.

Data Preprocessing

In this section, tweets were preprocessed to clean and transfer them for feature extraction. This step includes the following process:

Text Filtering: The researchers filtered non-Persian tweets, URLs, retweets, mentions, and some special characters such as '^%#-+'. In addition, all hashtags were removed and replaced informal words with formal words using Dehkhoda – the largest comprehensive Persian dictionary (Dehkhoda, 1931).

Emoji Dictionary: People usually use emoji during their daily conversations in microblogs such as Twitter and Facebook. As a result, the present researchers created an emoji dictionary containing all Twitter emojis. These emojis were manually labelled as happy, sad, and neutral. Based on the dictionary, each emoji in a retrieved tweet was replaced by its relative label.

Proverb Dictionary: Persian speakers mostly use slang expressions as well as several common proverbs in their conversations (Hokmi, 2017). 1,000 common Persian slang and proverbs were collected using Dehkhoda Dictionary. The study also used the website created by Ehsan (2013) to find some other slangs such as:"به جهنم"(The hell with it), "دهنمو سرویس کردی" (bite me), "دلم مثل سیر و سرکه میجوشه" (I have a butterfly in my stomach), etc. Finally, each tweet was scanned to replace its proverb or slang with the direct meaning.

Data Preparation

In this section, four types of preparation were performed on the dataset, namely (1) Normalisation, (2) word Tokenisation, (3) Lemmatisation, and (4) Part-of-Speech (POS) Tagging.

Normalisation: This step converted a list of words into a uniform sequence. With normalisation, the researchers aimed to overcome profound challenges in Persian including: (1) Existence of various prefixes such as "ب" (b), "بر"(bar), "پس"(pas), "فرا" (fara), etc.; (2) Different encoding forms for some characters like "ی" (y) and "ک" (k); (3) Using half-space such as"آن‌ها"and "سخت‌تر" ; and (4) Existence of a wide range of suffixes such as "ها" (ha), "ترین"(tarin), "یان" (yan), etc. (Mohtaj, Roshanfekr, Zafarian, & Asghari, 2018).

Tokenisation: This step was conducted to break each tweet down to its constitutive words. Let us consider an example:"من یک برنامه‌نویس هستم" (I am a programmer . It converts to "هستم" "من"، "یک"، "برنامه‌نویس"،.

Lemmatisation and Stemming: This step was done to decrease the size of the dataset and find the root of all words.

Part-of-Speech (POS) Tagging: It is a process of converting tweets into lists of tuples where each tuple has a form (word, POS tag). The POS tag signifies whether the word is a noun, adjective, verb, and so on.

All preparation steps were done using Hazm, a python library for digesting Persian text.

Feature Extraction

In this section, four sets of features were extracted, namely (1) Sentiment Feature, (2) Deep Polarity Feature, (3) POS Feature, and (4) Punctuation Feature.

These features were extracted in a way that covered different types of Persian sarcasm. Based on these features, all of the retrieved tweets were represented by feature vectors. This section explains how to represent a tweet as a feature vector to train a classifier for sarcasm identification.

Deep Polarity Features

This section focuses on sentence-level inconsistency. Let us consider the twe et:“خوشبختم من گوشیم خراب شد چه قدر!!!” (My phone is broken how lucky I am!!!). There is a sentence-level inconsistency between the first and second parts of the tweet (i.e. My phone is broken *and* how lucky I am!!!). We considered each tweet with more than *n* tokens as a multiple sentence tweet (MST). Based on the authors' observation on about 1,000 retrieved tweets, the proper values for *n* are 6, 12, and 18, respectively. The best value for *n* was evaluated in the Analysis and Results section. Each MST was divided into two parts. Thus, for the mentioned tweet, it is as follows:

tweet: My phone is broken how lucky I am!!!
n > 6 or n>12 or n>18? Tweet is MST so divide it into two equal parts: skip it
n =8, Hence we have:
Part1: My phone is broken
Part 2: how lucky I am!!!

If there is any sentiment inconsistency between the first and second parts of the tweet, it will hint about being sarcastic. To fulfil this, first, the present research combined two deep neural network models, LSTM and convolutional neural network, as proposed by Roshanfekr et al. (2017). Then, the combined deep learning model was employed to each MST. Afterwards, the deep model was applied to the first and second parts of each tweet respectively. Consequently, two new binary features, *dpf1* and *dpf2* were introduced (which stand for deep polarity feature).

The feature *dpf1* is activated if the first and second parts of the MST do not have the same sentiment.

The feature *dpf2* is activated if the first or second parts of the MST do not have the same sentiment with the whole MST's sentiment.

For example, in this tweet: "*My phone is broken and how lucky I am*!!!", the sentiment analysis model detected positive sentiment for this tweet and detected negative sentiment for the first part. Thus, the feature *dpf2* was activated.

Sentiment Feature

This section concentrates on word-level inconsistency, which refers to the coexistence of negative and positive words within the same tweet. For example,"چه خوبه که انقدر بدبختم" (I love my misfortune). There is a word-level inconsistency between "خوب" (love) and "بدبخت" (misfortune). To identify such inconsistency, a sentiment dictionary was provided by using two popular Persian dictionaries, Moein (1972) and Dehkhoda (1931). The sentiment dictionary consisted of 2,500 emotional words along with their polarities and scores. There were four different polarities that were considered as positive, high positive, negative, and high negative. The scores were integer from 2 (i.e. high positive) to -2 (i.e. high negative). If any word within the tweet did not exist in the sentiment dictionary, its score was set to 0. Table 1 represents these four polarities with related examples.

Table 1

Four Different Polarities with Examples

Polarity	Example
Positive	خوشحال (happy)،خوب (good)
High positive	عالی (excellent)،شگفت آور(wonderful)
Negative	بدبخت (miserable) ،ناراحت(sad)
High negative	افتضاح (awful)،چندش (disgusting)

Noticeably, due to the lack of rich Persian lexicon, the researchers built a semantic dictionary on their own. First, they selected common emotional words along with their polarity from Dehkhoda (1931) and Moein (1972). Then, for each emotional word, a score between -2 to 2 was considered based on the word's translation in SentiStrength.

Using the semantic dictionary, four auxiliary features were extracted by counting the number of positive, negative, high positive, and high negative words in each tweet. These features were named as pw, nw, PW, and NW. Then, in line with Bouazizi and Ohtsuki's (2015) works, the ratio of the tweet *p(t)* is defined in Equation 1:

$$\boldsymbol{P(t)} = \frac{(\boldsymbol{\delta.PSW+psw})-(\boldsymbol{\delta.NGW+ngw})}{(\boldsymbol{\delta.PSW+psw})+(\boldsymbol{\delta.NGW+ngw})} \tag{1}$$

Where *t* is the tweet and δ is a weight given to the highly emotional words. For neutral words, δ is set to 0; otherwise, is set to 3.

Now, to find word-level inconsistency, the sentiment score for each tweet was calculated with regard to the sentiment dictionary. If each word within the tweet was found in the dictionary, the related sentiment score was assigned to it. Otherwise, the sentimental score was set to 0. After the sentimental scores were set for all words, the researchers calculated the total scores for each tweet using Equations 2 and 3, respectively:

$$\textit{Sum-Of-Pos} = \sum_{n=0}^{n}(\text{Positive and High Positive Score}) \tag{2}$$

$$\textit{Sum-Of-Neg} = \sum_{n=0}^{n}(\text{Negative and High Negative Score}) \tag{3}$$

Sum-Of-Pos and *Sum-Of-Neg* are the summations of the positive and negative scores and *n* refers to a tweet's length.

For each tweet, if both *Sum-Of-Pos* and *Sum-Of-Neg* were greater than 0, there might be a word-level inconsistency in the tweet. After conducting a preliminary experiment to find the optimum range of sentiment score, two positive and two negative binary features were extracted. They are defined as follows:

Low-Pos-Sentiment activated if *Sum-Of-Pos* $\leq$ -1
High-Pos-Sentiment activated if *Sum-Of-Pos* $\geq$3
Low-Neg-Sentiment activated if *Sum-Of-Neg* $\leq$ -1
High-Neg-Sentiment activated if *Sum-Of-Neg* $\geq$3

POS Feature

The idea of this section was inspired by the work of Davidov et al. (2010). First, the researchers checked about 800 sarcastic and non-sarcastic tweets' POS tags and noticed some special patterns appearing in most of the sarcastic tweets. Any pattern that appeared in both sarcastic and non-sarcastic tweets was discarded. Then, two most common patterns were selected as sarcastic patterns. These patterns are shown in Table 2. A new binary feature was then created, POS feature. If one of these patterns was recognised in each tweet, the POS feature was activated.

Table 2

POS Feature's Patterns

Pattern	Example
وای (wow) + Pronoun + Adverb + Adjective + Verb	وای من خیلی خوشبختم! (Wow, how lucky I am!)
Noun + Adjective + Adjective + Adjective + Adjective	یک روز ابری دلگیر کوفتی قشنگ دیگه! (a rainy displeased nice awkward day!)

Punctuation Feature

Many studies have shown that punctuation has a huge influence on text classification. In other words, sarcasm not only plays with words and meaning but also translates them into a certain use of punctuation or repetition of words to indicate some special moods such as anger, amazement, exaggeration, etc. (Tungthamthiti, Shirai, & Mohd, 2014). In this study, the number of repetition sequence of exclamation (!) and question marks (?) were counted separately. In addition, the number of repetitive characters such as خخخخ ,هههه, etc. were calculated. Then, two new binary features, *Low-Punc-feature* and *High-Punc-feature* were employed.

After examining a various range of values, the optimum number to activate these features was found as follows:

Low-Punc-feature activated if the number of ? or ! or repetitive characters < 3.
High-Punc-feature activated if the number of ? or ! or repetitive characters ≥ 3.

Consider the following tweet “آره! قیمتش خیلی عالیه!!!!!!!!” (Yah! The cost is really good!!!!!!!!). In this tweet, the sarcasm was revealed by the repetition of the exclamation mark (*High-Punc-feature* was activated).

Classification Algorithms

Based on the features extracted in the previous section, several classification algorithms were used to classify tweets into sarcastic and non-sarcastic. In this article, five standard classifiers were utilised, namely Decision Tree, Random Forest, Naïve Bayes, SVM, and Maximum Entropy. These algorithms were chosen because they have shown good performance in many text classification tasks (Tungthamthiti, Shirai, & Mohd, 2014).

ANALYSIS AND RESULTS

In the experiment, a total of 2,500 political Persian tweets were retrieved using Tweepy API and the first Persian sarcastic dataset was introduced with 1,200 sarcastic and 1,300 non-sarcastic tweets. Then, each tweet's label was manually checked by native speakers to confirm its label. The researchers divided the dataset into 1,600 training and 900 testing data. Python programming language was used to classify the sentiments and conduct the experiment.

In this section, the performance of each set of features was evaluated. The result of the proposed method was also compared against two baseline methods. To form the first baseline, a Naïve Bayes model was built using TF-IDF features. For the second baseline, an SVM model trained with N-gram features was deployed. The current work was also compared with the approach proposed by Tungthamthiti et al. (2014). The key performance indicators (KPIs) used to evaluate the model are as follows:

(1) Accuracy: It shows the fraction of all correctly classified tweets over the total number of tweets.
(2) Precision: It represents the number of tweets that have successfully been classified as sarcastic over the total number of tweets classified as sarcastic.
(3) Recall: It expresses the number of tweets that have successfully been classified as sarcastic over the total number of sarcastic tweets.

Performance of Each Set of Feature

In this section, the performance of the classification of each proposed feature set was checked separately.

During cross-validation: As shown in Figure 2, it was noticed that when n = 18, the performance of deep polarity feature increased, while with n=6, the performance was very low. It showed that with n = 18, there was more chance to have a multiple sentence tweet.

The performance of *dpf1* and *dpf2* was examined separately. It was observed that *dpf2* outperformed *dpf1*. It demonstrated that if the first or second part of the tweet did not have the same sentiment as the whole tweet's sentiment, there was more chance to have a sarcastic tweet. Figure 3 represents these results. On the other hand, it was noticed that the POS feature had very low accuracy and recall. It indicated that two sarcastic patterns seemed to be very inefficient. Figure 4 displays the results of evaluating the POS feature.

The punctuation and sentiment features had a high precision rate. This could be explained by the fact that tweets with contradictory emotional content were likely to be sarcastic. Figure 5 illustrates their performance. However, low accuracy of the sentiment feature was based on many words that did not exist in the sentiment dictionary. From this reason, enrichment of the Persian sentiment dictionary could be applied to boost the performance of sentiment features. It was also noticed that the ratio obtained in Equation 1 was unable be a good sentiment feature for each tweet due to its very poor performance.

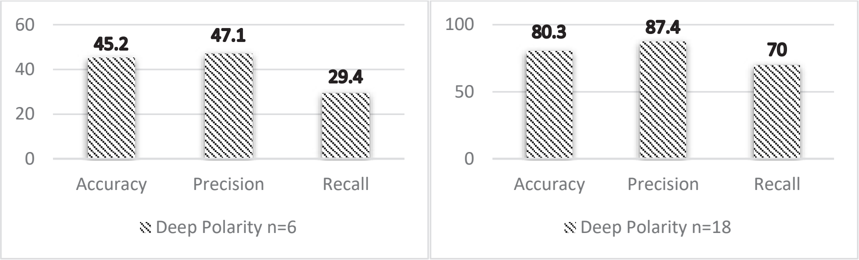

Figure 2. *Performance of classification using deep polarity feature (dpf1 and dpf2) with n=6, 18.*

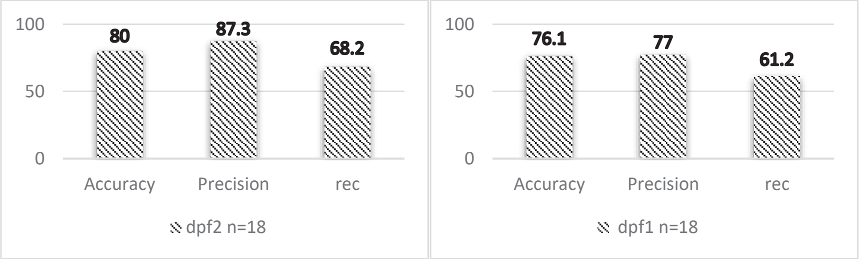

Figure 3. *Performance of dpf1 and dpf2 with n = 18.*

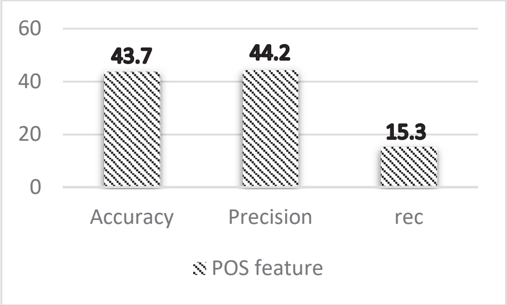

Figure 4. *Performance of classification using POS feature.*

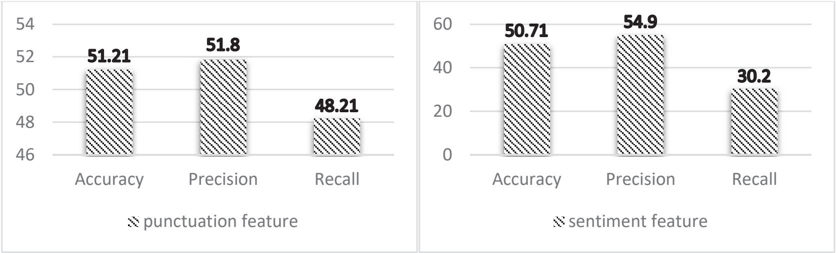

Figure 5. *Performance of classification using deep polarity feature with n=6, 18.*

On a test set: It was observed that the set of features that had better performance during cross-validation acted the same on the test. Nevertheless, it was clear that the performance on a test set was remarkably lower than that during cross-validation.

Overall performance: Table 3 shows the overall performances of the proposed model when all the features were used. This demonstrated that the combination of different sets of features performed better.

As illustrated in Figure 6, the best algorithm as SVM with an accuracy of 80.82%. According to the obtained results, SVM as capable of detecting sarcasm with high precision. Random Forest also represented good results. The worse accuracy was observed for Maximum Entropy with an accuracy of 68.43%.

Moreover, to evaluate the efficiency of the proverb dictionary, the researchers ignored using it in the preprocessing step. It was noticed that the accuracy of the classification decreased exponentially. The precision was also decreased as compared to when using the proverb dictionary. It demonstrated that most parts of Persian tweets needed to be translated into their direct meaning to prevent misclassification. Figure 7 illustrates the related results.

Table 3

Results Obtained by Five Classifiers

Classifiers	Accuracy	Precision	Recall
Decision Tree	74.60%	82.90	68.44%
Random Forest	79.44%	84.28%	65.50%
Naïve Bayes	71.24%	78.38%	59.98%
SVM	80.82%	88.30%	70.80%
Max-Entropy	68.43%	73.65%	54.92%

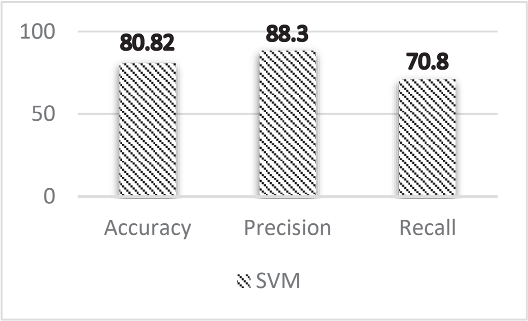

Figure 6. *Performance of the best classification algorithm.*

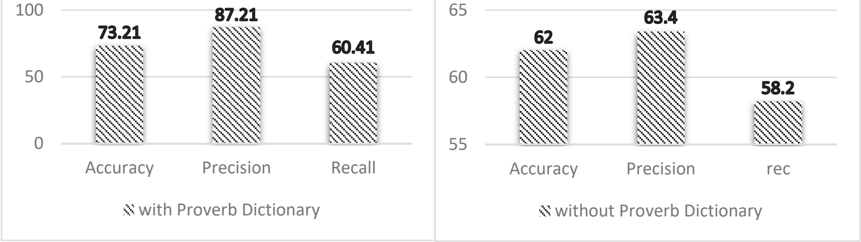

Figure 7. *Performance of classification regarding Proverb Dictionary.*

Baseline: To evaluate the potential of the proposed model, two baseline methods were considered as well as the approach proposed by Tungthamthiti et al. (2014).

In this section, the aforementioned KPIs with F1 score were utilised. F1 score combined precision and recall to represent a more reliable comparison (Bouazizi & Otsuki, 2016). It is defined in the following Equation 4:

$$F1 = \frac{2.\text{Precision}.\text{Recall}}{\text{Precision}+\text{Recall}} \quad (4)$$

The first baseline was a Naïve Bayes model using TF-IDF features. The results are given in Table 4. In line with the results, the model outperformed the first baseline.

The second baseline was an SVM model using N-gram (unigram, bi-gram, tri-gram). According to the obtained results, it was found that although the proposed approach outperformed the second baseline, the N-gram feature was still powerful for classification. Therefore, the combination of N-gram features and the proposed set of features could improve the overall results. On the other hand, based on the authors' observation, SVM trained with N-gram features itself failed to classify words that did not appear in the training data frequently. Nevertheless, the proposed features appeared in the training set several times, which could lead to better classification.

Moreover, the proposed model was compared with the approach conducted by Tungthamthiti et al. (2014), which suggested a sentence coherence feature. Although the effectiveness of this feature was completely proved in their study, the experimental results showed that the present method had better accuracy. However, the authors believed that considering some modifications to their sentence coherence feature based on Persian language structures could improve the efficiency of the method.

Table 4

Performance Comparison of the Proposed Approach and the Baseline

Methods	Accuracy	Precision	Recall	F-Score
Baseline 1	56.8%	56%	55.8%	57.14%
Baseline 2	79%	78.3%	76%	79%
Tungthamthiti et al. (2014)	60.8%	65%	41%	51.20%
Proposed Model	80.82%	88.30%	70.80%	80.82%

CONCLUSION

Since sarcastic text does not have any fixed structure, detecting sarcasm in textual data is a complicated task. This study proposed a method to detect Persian sarcasm based on deep learning and machine learning for the first time. Four sets of features were extracted, namely (1) Sentiment, (2) Deep Polarity, (3) POS, and (4) Punctuation. These features were extracted in a way that covered different types of Persian sarcasm. The performances of the classification of each proposed feature set were examined separately and it was noticed that the combination of different sets of features performed better. The results of the experiment showed that the proposed method had acceptable performance and reached an accuracy of 80.82% using the SVM algorithm. According to the obtained results, SVM is capable of detecting sarcasm with high precision. Moreover, the first proverb dictionary was created to translate several common expressions into their direct meaning. A total of 1,000 common Persian slangs and proverbs were collected using Dehkhoda Dictionary. This has shown remarkable improvement in the study's results, though a better result might be achieved if a bigger proverb dictionary was used, which could be suggested for future works. Furthermore, future works could add some new features to improve the classification results and examine on ways to use the output of the current research to boost the performance of the Persian sentiment analysis.

ACKNOWLEDGMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

REFERENCES

Al-Otaibi, S. T., Alnassar, A., Alshahrani, A., Al-Mubarak, A., Albugami, S., Almutiri, N., & Albugami, A. (2018). Customer satisfaction measurement using sentiment analysis. Retrieved May 4, 2020, from ResearchGate website:https://www.researchgate.net/publication/323536432_Customer_Satisfaction_Measurement_using_Sentiment_Analysis

Bharti, S. K., Naidu, R., & Babu, K. S. (2017). Hyperbolic feature-based sarcasm detection in tweets: A Machine Learning Approach. Retrieved May 4, 2020, from undefined website: https://www.semanticscholar.org/paper/Hyperbolic-Feature-based-Sarcasm-Detection-in-A-Bharti-Naidu/d46fa4117b009fe3128c496da3dc5c6f3f446791

Blamey, B., Crick, T., & Oatley, G. (2012). R U :-) or :-(? Character- vs. word-gram feature selection for sentiment classification of OSN Corpora. *Research and Development in Intelligent Systems XXIX*, 207–212. https://doi.org/10.1007/978-1-4471-4739-8_16

Bouazizi, M., & Ohtsuki, T. (2015, December 1). Sarcasm detection in Twitter: "All your products are incredibly amazing!!!" - Are they really? https://doi.org/10.1109/GLOCOM.2015.7417640

Bouazizi, M., & Otsuki Ohtsuki, T. (2016). A pattern-based approach for sarcasm detection on Twitter. *IEEE Access*, *4*, 5477–5488. https://doi.org/10.1109/access.2016.2594194

Davidov, D., Tsur, O., & Rappoport, A. (2010). *Semi-supervised recognition of sarcastic sentences in Twitter and Amazon* (pp. 15–16). Retrieved from Association for Computational Linguistics website: https://www.aclweb.org/anthology/W10-2914.pdf

Dehkhoda, A. (1931). *Dehkhoda dictionary* (2020 ed.). Retrieved from https://icps.ut.ac.ir/

Ehsan, E. (2013). *Persian slang and modern terms*. Retrieved from http://www.weare.ir/interesting/%D8%A7%D8%B5%D8%B7%D9%84%D8%A7%D8%AD%D8%A7%D8%AA-%D8%B9%D8%A7%D9%85%DB%8C%D8%A7%D9%86%D9%87-%D9%88-%D8%A7%D9%85%D8%B1%D9%88%D8%B2%DB%8C-%D8%B2%D8%A8%D8%A7%D9%86-%D9%81%D8%A7%D8%B1%D8%B3%DB%8C/

Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., & Lehmann, S. (2017). Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. https://doi.org/10.18653/v1/d17-1169

Gelbukh, A. (2009). Computational linguistics and intelligent text processing. In A. Gelbukh (Ed.), *Lecture notes in computer science*. https://doi.org/10.1007/978-3-642-00382-0

Ghosh, A., & Veale, T. (2016). *Fracking sarcasm using neural network* (pp. 161–169). Retrieved from Association for Computational Linguistics website: https://www.aclweb.org/anthology/W16-0425.pdf

Hazarika, D., Poria, S., Gorantla, S., Cambria, E., Zimmermann, R., & Mihalcea, R. (2018). CASCADE: Contextual sarcasm detection in online discussion forums. *ArXiv:1805.06413 [Cs]*. Retrieved from https://arxiv.org/abs/1805.06413

Hokmi, K. (2018). Improving the learning skills of paradoxical elements, simile and irony in high school students. *Journal of Human Sciences Research*, *4*(9), 1–20. Retrieved from https://www.civilica.com/Paper-JR_JHSR-JR_JHSR-4-9_013.html

Liu, P., Chen, W., Ou, G., Wang, T., Yang, D., & Lei, K. (2014). Sarcasm detection in social media based on imbalanced classification. *Web-Age Information Management*, *8485*, 459–471. https://doi.org/10.1007/978-3-319-08010-9_49

Lunando, E., & Purwarianti, A. (2013). Indonesian social media sentiment analysis with sarcasm detection. 2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS), 195–198. https://doi.org/10.1109/icacsis.2013.6761575

Maynard, D., & Greenwood, M. (2014, May 1). Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis. Retrieved May 4, 2020, from ACLWeb website: https://www.aclweb.org/anthology/L14-1527/

Moein, M. (1972). *Moin dictionary* (2020 ed.). Retrieved from https://icps.ut.ac.ir/

Mohtaj, S., Roshanfekr, B., Zafarian, A., & Asghari, H. (2018, May 1). Parsivar: A language processing toolkit for Persian. Retrieved May 4, 2020, from ACLWeb website: https://www.aclweb.org/anthology/L18-1179/

Parmar, K., Nivid Limbasiya, & Dhamecha, M. V. (2018). Feature based composite approach for sarcasm detection using MapReduce. Retrieved May 4, 2020, from undefined website: https://www.semanticscholar.org/paper/Feature-based-Composite-Approach-for-Sarcasm-using-Parmar-Limbasiya/608a5b3cacbf64a29ad67e0c7b66a378bb4e50d5

Rahayu, D. A. P., Kuntur, S., & Hayatin, N. (2018). Sarcasm detection on Indonesian Twitter feeds. *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, *5*, 137–141. https://doi.org/10.1109/eecsi.2018.8752913

Rajadesingan, A., Zafarani, R., & Liu, H. (2015). Sarcasm detection on Twitter. *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining - WSDM '15*. https://doi.org/10.1145/2684822.2685316

Riloff, E., Qadir, A., Surve, P., De Silva, L., Gilbert, N., & Huang, R. (2013). *Sarcasm as contrast between a positive sentiment and negative situation* (pp. 704–714). Retrieved from Association for Computational Linguistics website: https://www.aclweb.org/anthology/D13-1066.pdf

Roshanfekr, B., Khadivi, S., & Rahmati, M. (2017). Sentiment analysis using deep learning on Persian texts. *2017 Iranian Conference on Electrical Engineering (ICEE)*, 1503–1508. https://doi.org/10.1109/iraniancee.2017.7985281

Schifanella, R., de Juan, P., Tetreault, J., & Cao, L. (2016). Detecting sarcasm in multimodal social platforms. *Proceedings of the 2016 ACM on Multimedia Conference - MM '16*. https://doi.org/10.1145/2964284.2964321

Seyed Sadeqi, S. M., & Ehsanjou, M. (2018). Examining and explaining proverbs in Moqaddame va Yazdah maqale from: Elahi Name Attar. *Ourmazd Journal*, *9*(29), 1–35. Retrieved from https://www.civilica.com/Paper-JR_OURMAZD-JR_OURMAZD-9-29_003.html

Son, L. H., Kumar, A., Sangwan, S. R., Arora, A., Nayyar, A., & Abdel-Basset, M. (2019). Sarcasm detection using soft attention-based bidirectional long short-term memory model with convolution network. *IEEE Access*, *7*, 23319–23328. https://doi.org/10.1109/access.2019.2899260

Suhaimin, M. S. M., Hijazi, M. H. A., Alfred, R., & Coenen, F. (2017). Natural language processing based features for sarcasm detection: An investigation using bilingual social media texts. *2017 8th International Conference on Information Technology (ICIT)*, 703–709. https://doi.org/10.1109/icitech.2017.8079931

Suhaimin, M. S. M., Hijazi, M. H. A., Alfred, R., & Coenen, F. (2019). Modified framework for sarcasm detection and classification in sentiment analysis. *Indonesian Journal of Electrical Engineering and Computer Science*, *13*(3), 1175–1183. https://doi.org/10.11591/ijeecs.v13.i3.pp1175-1183

Tungthamthiti, P., Shirai, K., & Mohd, M. (2014, December 1). *Recognition of sarcasms in tweets based on concept level sentiment analysis and supervised learning approaches*. Retrieved May 4, 2020, from ACLWeb website: https://www.aclweb.org/anthology/Y14-1047/