Near Real-Time Detection of Misinformation on Online Social Networks

Lennart van de Guchte[1,2](✉), Stephan Raaijmakers[2], Erik Meeuwissen[2], and Jennifer Spenader[1]

[1] University of Groningen, 9747 AG Groningen, The Netherlands
lennartvandeguchte@gmail.com

[2] TNO, 2595 DA The Hague, The Netherlands

Abstract. In this paper, we focus on the automatic detection of misinformation articles on online social networks. We study micro-blog posts that propagate news articles and classify these articles as misinformation or trusted information. We do this by extracting a comprehensive set of network and linguistic features and propose a deep learning model that combines both feature types. Experiments on real data demonstrate that our proposed method detects misinformation with an accuracy of 93% in near-real time. Moreover, we compare network and linguistic features with respect to the earliness of detection and combine these features with temporal information about diffusion patterns. We find that combining both feature types is optimal for the detection of misinformation articles in near-real time.

Keywords: Misinformation · Early detection · Online social network · Deep learning

1 Introduction

The massive usage of online social networks has amplified the negative effects that misinformation has on society. To counter misinformation fact-checkers, such as politifact.com or snopes.com, verify news stories and correct inaccurate or false information. However, manual fact-checking cannot keep up with the quantity or speed at which deceptive information is currently propagated. Further, researchers have concluded that correcting misinformation after dissemination is too late to be fully effective e.g. [18], due in part to the "continued-influence effect" [19]: damage caused by exposure to misinformation is hard to undo.

This is why detecting and verifying misinformation in real-time, as it begins to spread, is crucial. In this work, we focus on micro-blog posts that broadcast hyperlinks to news articles, either misinformation or not. We ignore the actual context of these hyperlinks but focus on linguistic and network properties of these posts. This approach is motivated by the fact that it appears to be difficult and non-trivial to use only the text of an article for detection [27].

Previous efforts to automate misinformation detection have also utilized social context information, such as micro-blog posts, diffusion behaviour and user characteristics, in combination with machine learning methods [27]. Although various studies proved that these features are effective in detecting misinformation after dissemination, only a few studies applied these features to early detection [25]. In this paper, we focus on the effectiveness of network and linguistic features for near real-time detection of misinformation.

Network features are extracted from the information diffusion networks that we deduced from social interactions and include diffusion patterns, user characteristics and social bot indicators, while linguistic features are extracted from micro-blog posts. We study the performance of network and linguistic features when combined with temporal information and propose a deep learning model that combines both feature groups.

The main contribution of this paper consists of tweet volume-independent detection of misinformation in near real-time. Specifically:

- We propose a new method for detecting misinformation articles in near real-time with high accuracy, by combining linguistic and network features that are extracted from an online social network.
- We show the relative strength of network and linguistic features for discriminating misinformation from trusted articles for various detection deadlines, i.e. time after a hyperlink to a news article is broadcast.
- We contribute a novel Twitter dataset that includes tweets related to misinformation and trusted political news articles. The dataset consists of 1300 political related articles and can be used to reconstruct the dissemination of news articles on Twitter.[1]

The rest of this paper is organized as follows. Section 2 presents the relationship to existing work. In Sect. 3 we formulate the problem in detail and Sect. 4 explains our approach. Section 5 describes the experiments we conducted and discusses the results. Finally, in Sect. 6 we draw conclusions and present ideas for future work.

2 Related Work

Online social networks have been investigated extensively for linguistic and network features. Linguistic features are usually extracted from micro-blog posts that propagate misinformation. In [2] a comprehensive set of sentiment words, hashtags, emoticons, orthography and topic related features was successfully used to detect tweets that contain misinformation. In [32], word embeddings techniques were utilized to create linguistic features and combined this with deep learning methods for classification.

[1] The dataset is available at https://github.com/lennartvandeguchte/Near-real-time-misinformation-detection.

Previous research has found that misinformation spreads significantly farther, faster, deeper, and more broadly within networks than truthful information [31], in part because of active propagation by social bots [7]. This information is utilized to model the temporal characteristics of news diffusion using propagation paths or diffusion networks. In [16], structural and temporal features were extracted from diffusion networks to successfully detect misinformation on Twitter. It was found that adding linguistic features improved performances.

In [29], linguistic models were studied to classify suspicious tweets and combined this with network features. Linguistic features consisted of syntax, semantic cues, and document embeddings while the network features represented some simple user interactions. It was found that adding network features outperformed all linguistic models and, besides, utilizing a recurrent or convolutional neural network as classifier was better compared to logistic regression. However, these features were extracted after misinformation was already propagated through the network.

Few studies investigated the effectiveness of linguistic and network features over different time windows. In [15], a comprehensive set of linguistic, network, user, and temporal features was evaluated for time windows from 3 till 56 days. They showed that the effectiveness of temporal and network features increases over time while that of linguistic features stayed the same. However, linguistic features outperformed all other feature groups for the smallest time window (3 days). Another interesting finding is that a combination of all features was optimal for the largest time window while for the smallest time window this model was outperformed by a combination of user and linguistic features. The results are evidence that optimal feature selection may depend on the targeted detection time.

Recently, some studies focused on the early detection of misinformation [9]. In [3], linguistic features from a sequence of micro-blog posts are combined with a recurrent neural network that integrates a soft attention mechanism and successfully detects misinformation in an earlier stage. Other research also utilize recurrent neural networks to capture temporal information from propagation paths and combine this with other features such as user characteristics [20] or linguistic content [21]. These deep learning models have shown to outperform competitive methods and detect misinformation in an earlier stage. A limitation of these models is that the earliness of detection depends on the length of the propagation path (e.g. number of retweets). This means that only with abundant data at an early stage of dissemination these models are suitable for early detection. For example, in [20] it has been shown that the proposed model can detect misinformation after 5 min with 92% accuracy, however, to do this they need 40 tweets. Since propagation paths vary in size this approach does not always detect misinformation in 5 min.

A study similar to our current approach where near real-time detection is being investigated along with the relative contribution of different feature sets was carried out in [30]. In this study the detection accuracy was measured as a function of latency for temporal and non-temporal models when using linguistic, user or propagation features. The results showed that when time passed the temporal model and propagation features became stronger while for real-time detection non-temporal and linguistic features slightly outperformed the others, though not very accurate.

3 Problem Statement

We investigate in this paper if near real-time detection is possible by analyzing the linguistic, network and temporal properties of micro-blog posts. To study this we formulate the detection of misinformation as a supervised binary classification problem in which misinformation and trusted articles are being discriminated. An article (A) is represented by the stream of messages (m_t) that post or share this article over time (t): $A(t) = \{m_0, m_1, ..., m_t\}$. For the early classification of an article only a subset of the messages (A_s) is available which depends on the detection deadline (T). A later detection deadline might improve the results since there is more data available but affects the earliness of detection. To find out if there is an optimal moment in time to balance the trade-off between earliness and effectiveness we vary with T.

In contrast with [30], we use different data, features, and classification models and present actual instead of relative detection times. In general, it is difficult to compare different detection approaches because shared datasets are lacking. To overcome this problem, we constructed an up-to-date dataset and make it available to the research community.[2]

4 Approach

4.1 Construction of a Novel Dataset

In line with the majority of research on early misinformation detection we use Twitter data to evaluate our algorithm. The Twitter policy only allows to publish tweet IDs and to reconstruct a data set Twitter's API should be used. However, since Twitter has started to actively remove suspicious accounts and tweets in 2018[3] it has become impossible to fully reconstruct these data sets. Moreover, because the production and dissemination of misinformation is constantly changing detection algorithms should be evaluated using up-to-date data. Therefore we constructed a novel Twitter data set by making use of two publicly available tools:

- Hoaxy [26], for determining whether news content consists of misinformation.
- NewsAnalyzer [1], for scraping trusted news sources.

[2] The dataset is available at https://github.com/lennartvandeguchte/Near-real-time-misinformation-detection.

[3] https://www.nytimes.com/2018/07/11/technology/twitter-fake-followers.html.

Since misinformation detection is topic-dependent and over-represented in political news [31] we decided to validate our research by only collecting political-related misinformation articles.

Hoaxy combines web scraping, web syndication and Twitter APIs to collect and analyse misinformation articles. To do this it makes use of a comprehensive list of 120 low-credibility sources in the U.S which is compiled and published by reputable news and fact-checking organizations. These sources are known for frequently publishing hoaxes, rumors, false news, and conspiracy theories, but may also publish accurate rapports.[4] By utilizing the articles URLs Hoaxy collects all tweets that include these URLs. NewsAnalyzer is used to collect trusted information and works similarly as Hoaxy but is able to use a provided list of news sources. To collect trusted articles we rely on previous work that investigated the trustworthiness of various news sources from a republican, democratic and fact-checker perspective [24]. A combined score from all perspectives was given to generate a list of most trusted news sources in the U.S. from which we used 9 as input for NewsAnalyzer: *CBS News*, *CNN*, *USA Today*, *ABC News*, *The Washington Post*, *The New York Times*, *Fox News*, *NBC News*, and *Huffington Post*. After extracting the data NewsAnalyzer categorized the articles as politics or not-politics if this topic was mentioned in the URL, this was the case for 8 out of 9 news sources. For the last source and for all misinformation articles we build a topic classifier to categorize an article as politics or not-politics.

As classifier we used a multilayer perceptron (MLP) and as input features we created document embeddings by using the Doc2Vec algorithm [17]. We used the implementation of Doc2Vec from Python's Gensim library[5] and trained the algorithm with its default hyper-parameters, except for the number of epochs (100), window size (10), negative size (5), and sampling threshold (1e−5). This resulted in 300-dimensional vectors to represent the articles. To train the Doc2vec and MLP we used 2335 politics and 2469 not-politics articles we collected using NewsAnalyzer that were already categorized as such according to its original news source. This data was divided into a validation set (20%), for hyper-parameter optimization, and a train/test set (80%) to evaluate our model. Performing a 10-fold cross validation resulted in a average accuracy of 94%. After this we trained the model one more time on all available data and used this to classify the misinformation articles, and the trusted articles which were not yet categorized.

Finally, all articles with less than 20 tweets were thrown away to ensure that all articles in the data set have been exposed to a broad audience. This resulted in a data set of 1300 political related articles equally balanced between misinformation and trusted information. The articles are published in 2019 in the period between January 1 and August 1. Each article consists of multiple related tweets from which the amount can vary between 20 and 5000. Since we did not manually check the quality of the data this data set can be considered as silver standard. As a contribution to the research community we made the data set

[4] We did not verify to what extend these sources publish misinformation and therefore rely entirely on Hoaxy for our misinformation label.

[5] https://radimrehurek.com/gensim/models/doc2vec.html.

available in the form of a data challenge for the International Conference on Military Information and Communication Systems (ICMCIS) 2020[6] and published it on Github.[7]

4.2 Information Diffusion Network

In order to capture temporal information from the dissemination of news articles we deduced information diffusion graphs from the Twitter data. In these networks the nodes represent tweets and the edges show the relationship between an original tweet and a share (retweet, quoted tweet or reply tweet). Each node has a timestamp that corresponds to the time that has passed since the first tweet in the network was posted. By iterating over different timestamps we can now observe how the network evolves over time. Note that these networks consist solely of multiple star networks with a maximum cascade length of 1, as depicted in Fig. 1. In reality, users could retweet other retweets and therefore create longer cascades. The reason for this is that Twitter's API only provides limited data that points all retweets to the original tweet. Though, approaches have been proposed in which cascades are approximated based on tweet timestamps and friend-follower relationships it appears that this process is time-intensive [30] and therefore not suitable for real-time detection.

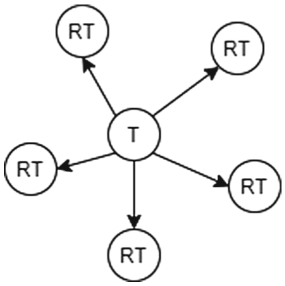

Fig. 1. Example of a star network.

Since the amount of tweets per article can vary a lot (between 20 and 5000 in our data) we transform these variable-length time series into fixed-length time series. This is done by dividing the diffusion network into *snapshots*. Snapshots represent the state of the network at a particular point in time. For example, if the number of snapshots is four ($N_s = 4$) and the detection deadline is four hours ($T = 4$) than a snapshot represents the diffusion network after every hour.

[6] https://www.kaggle.com/c/icmcis2020.
[7] https://github.com/lennartvandeguchte/Near-real-time-misinformation-detection.

4.3 Feature Extraction

Utilizing the previous described diffusion networks and tweets we extract two groups of features: network and linguistic features. Network features have been extracted per snapshot while linguistic features were extracted per tweet. The linguistic feature representation per snapshot is computed by averaging over all tweets that are included in the snapshot.

Network Features. The network features can be categorized into the following three categories: *diffusion patterns*, *followers* and *bots*. *Diffusion patterns* include the number of nodes (all tweets), original tweets, shares, likes, and cascades over time. To adjust these features to the varying network sizes per article we also computed their relative values by dividing them with the network size of the correlating snapshot. The *Followers* features consist of the number of followers, 'well known users' (>10,000 followers), 'superspreaders' (>100,000 followers), and their relative values. Finally, we used Botometer [5], a state-of-the-art bot detection algorithm for Twitter, to compute bot scores of all users in the network. Botometer uses more than 1,200 features which they categorized as network, content, temporal, user, sentiment, and friend features. For each of these categories, and for all features together, a bot score that indicates the likelihood of an account being a bot is computed. Per bot score we computed the average score for all users in a snapshot. Furthermore we computed two average bot scores for the users that posted original tweets/retweet and used a bot threshold of 0.5 to count the total number of bots for accounts that exceeded this threshold. In Table 1 we presented an overview of all network features.

Linguistic Features. We extracted two types of linguistic features: *tweet embeddings* and *handcrafted features*. First, we preprocessed the tweets by removing the URL (link to the article) and @username to prevent the algorithm from becoming biased. The *tweet embeddings* were then computed using the pre-trained word embeddings from Godin et al. [8]. These embeddings were created by training the Word2Vec [22] algorithm on a Twitter corpus of 400 million tweets. For each tweet, we computed a tweet embedding by averaging the word embeddings for each word in the tweet. If a word did not occur in the pre-trained vocabulary we skipped it.

For the handcrafted features we used a variety of feature shown previously to be effective for misinformation detection (see Table 1 for an overview). We used the sentiment classifier TextBlob[8] to compute polarity and subjectivity scores for every tweet [14]. Furthermore, a group of features regarding the orthography of a tweet was extracted. These features include exclamation marks, capital letters, hashtags, mentions, tweet length and emojis. For the emojis we also used a sentiment map that provide a sentiment score for 751 most used emojis on Twitter [23]. The rest of the features were extracted by utilizing several lexicons. Since these lexicons were developed for formal English words, and tweets

[8] https://textblob.readthedocs.io/en/dev/index.html.

Table 1. Overview of all network and handcrafted linguistic features.

Feature	Amount	Representation
Network features		
Number of nodes	1	int
Increase in number of nodes	2	int
Number of original tweets	1	int
Number of shares	1	int
Number of likes	2	int
Number of cascades	2	int
Average like per cascade	2	float
Number of followers	2	int
Number of well known users	2	int
Number of superspreaders	2	int
Average botscores	7	float
Average botscore original tweets	1	float
Average botscore shares	1	float
Percentage of bots	1	float
Handcrafted linguistic features		
Polarity score (TextBlob)	1	float
Subjectivity score (TextBlob)	1	float
Number of exclamation marks	1	int
Percentage exclamation marks	1	float
Number of capital letters	1	int
Number of continuous capital letters	1	int
Hashtags	2	int, binary
Mentions	2	int, binary
Tweet length	1	int
Emojis	1	binary
Emojis sentiment score	1	float
Positive words	2	int, binary
Negative words	2	int, binary
Valence, arousal, dominance	3	float
Weak subjective words	2	int, binary
Strong subjective words	2	int, binary
Hedges	2	int, binary
Assertive verbs	2	int, binary
Factive verbs	2	int, binary
Implicative verbs	2	int, binary
Report verbs	2	int, binary
Verbs of attribution	2	int, binary
Discourse connectives	2	int, binary

contain a lot of informal language, we performed some extra preprocessing. In [6] they studied a variety of preprocessing methods especially used for tweets from which we applied the following in chronological order: replaced slang with formal English, removed integers, punctuation, hashtags and emoticons, replaced contractions by its complete form, and corrected spelling errors/typos by using Norvig's spelling corrector.[9]

To extract features regarding biased and subjective language we use six lexicons that were found to be successful in discriminating suspicious from verified tweets [29]. These lexicons include assertive verbs (assert a level of certainty to the complement cause), factive verbs (presuppose the truth of their complement cause), implicative verbs (implicate the truth or untruth of their complement), reportive verbs (also implicate the truth or untruth but preserve the truth under negation), hedges (introduce uncertainty about the proposition), and subjective words to indicate biased and subjective language. Further, to measure a writer's emotions, we used the Affective Norms for English Words (ANEW) [33]. This is a lexicon of 13,915 English lemmas with related valence, arousal, and dominance

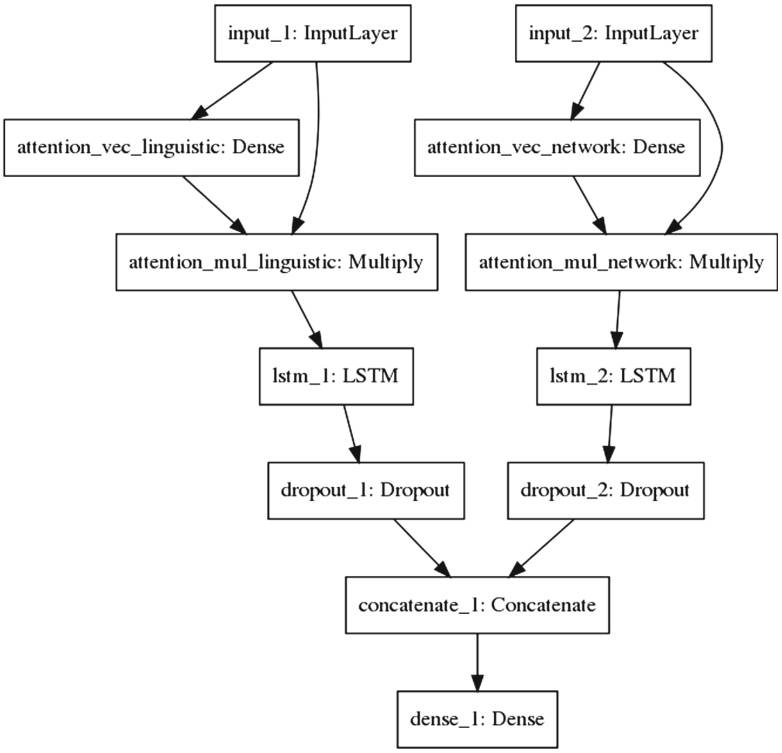
input_1: InputLayer
input_2: InputLayer
attention_vec_linguistic: Dense
attention_vec_network: Dense
attention_mul_linguistic: Multiply
attention_mul_network: Multiply
lstm_1: LSTM
lstm_2: LSTM
dropout_1: Dropout
dropout_2: Dropout
concatenate_1: Concatenate
dense_1: Dense

Fig. 2. Model architecture.

[9] http://norvig.com/spell-correct.html.

norms. Finally, we constructed two new lexicons with verbs of attribution and discourse connectives.[10]

4.4 Classification

We evaluated the discriminative power of both feature classes using a long short-term memory (LSTM) [10], which is a type of recurrent neural networks. It is well-known that RNNs can effectively capture the temporal dynamics of the spread of misinformation [20,21]. Additionally, we use an attention layer in between the input layer and the hidden LSTM layer to function as dynamic feature weighting technique [13]. Thus, unlike conventional attention mechanisms for RNNs, that compute weights for various time steps, this attention layer learns to weight features depending on the input vector. The advantages of using this technique is two-fold. First, it learns the feature importance by linking input values to the target value (misinformation or trusted information). This means that the feature importance is context dependent which results in different features being important for different misinformation articles. Secondly, it can give a deeper insight in which features are useful in general or for some specific cases of misinformation.

In order to combine both feature spaces (network and linguistic features) we rely on a technique called "late fusion". This method learns a combined representation from multiple input streams and has proved to be effective in various vision tasks [11]. In our case this means that we have an LSTM layer for each feature space separately and concatenate these latent feature spaces afterwards using a dense layer. This model was implemented using Keras [4] and is visualized in Fig. 2. The model has two input streams for the network and linguistic features, respectively. In case of evaluating one feature set the model uses only one input stream and discards the concatenation layer.

5 Experiments and Discussion

5.1 Experimental Settings

The various feature groups have resulted in five different models represent by the following acronyms: LSTM-N (network features), LSTM-H (handcrafted linguistic features), LSTM-T (tweet embeddings), LSTM-L (all linguistic features), and LSTM-ALL (combines LSTM-N and LSTM-L). For experimentation we divided the data set into a validation set (20%) and a train/test set (80%). The validation set was used for hyper-parameter optimization based on 10-fold cross-validation with a grid search. The optimal parameters for our models are shown in Table 2. To train the algorithm we applied stochastic gradient descent with the Adam update rule [12] and Dropout [28] was used for regularization. The number of epochs was set to 100 and early stopping was applied when the validation loss saturated for 10 epochs.

[10] The used lexicons can be found at https://github.com/lennartvandeguchte/Near-real-time-misinformation-detection.

Table 2. Model configurations obtained by doing a grid search.

	LSTM-N	LSTM-H	LSTM-T	LSTM-L	LSTM-ALL
Learning rate	0.01	0.001	0.001	0.001	0.001
Batch size	20	20	20	20	20
# LSTM cells	50	50	500	600	50 & 600
Dropout rate	0.1	0.1	0.1	0.1	0.1

To find out how different models perform with respect to early detection and the available temporal information we conducted different experiments. First we investigated if our models were able to learn from temporal information by using snapshots of a diffusion network. We did this by choosing two detection deadlines (15 min and 4 h) and varied the amount of snapshots for these time windows. Secondly, we used the optimal number of snapshots to perform the rest of our experiments with detection deadlines between 1 min and 10 days. For every configuration we applied a 10-fold cross validation on the 80% train/test data set and computed the average accuracy plus their standard deviation. A Wilcoxon signed-rank test was performed to measure significance and we reject the null hypothesis when the p-value is lower than 0.05.

5.2 Results

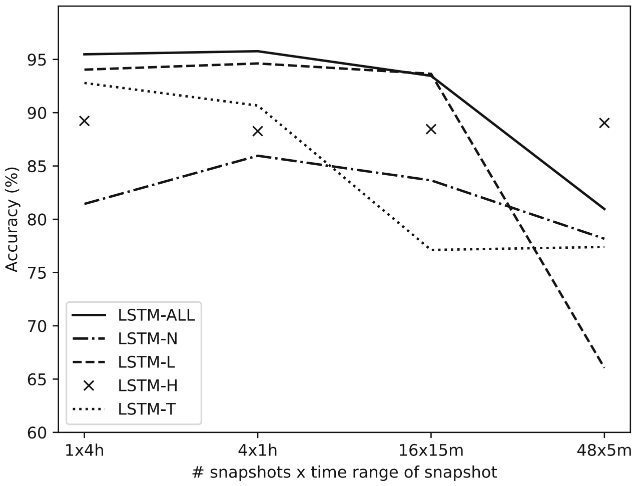

Fig. 3. Model accuracy for varying snapshots and a detection deadline of 4 h.

Snapshots. The performance of all models with a detection deadline of 4 h and varying amounts of snapshots is shown in Fig. 3. We observe that the accuracy of most models decreases with a larger number of snapshots except for LSTM-N. The network features can take advantage of the temporal information and show an increase in accuracy when the number of snapshots is 4 instead of 1, although not significant ($Z = 13.0, p = 0.138$). For higher amounts of snapshots the performance of all models degrades strongly. We repeated this experiment for a detection deadline of 15 min and found similar results, as shown in Table 3. Since no significant improvement was found when using multiple snapshots we performed the remaining experiments using only 1 snapshot.

Detection Deadlines. Figure 4 shows how the different models perform for ascending detection deadlines. We find that LSTM-ALL outperforms all other models for a detection deadline of 1 min ($Z = 3.0, p = 0.036$) indicating that a combination of network and linguistic features is optimal for near real-time detection. Furthermore we observe that each model improves for later detection deadlines. This makes sense because more social context becomes available. However, we see that network features take more advantage from later detection deadlines than linguistic features, a result also found in [30] and [15]. Interestingly, we find a decrease in accuracy between a detection deadline of 1 min and 5 min for some models. We assume that this is due to an increase in noise in the data when we average over multiple feature vectors in a snapshot.

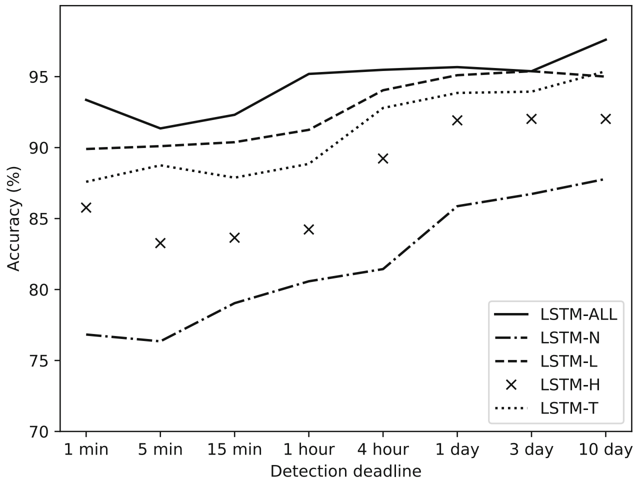

Fig. 4. Model accuracy using 1 snapshot and varying detection deadlines.

Model Comparison. The classification results of all experiments are presented in Table 3. We find that linguistic features (both handcrafted and tweet embeddings) outperform the network features for all detection deadlines. For the linguistic models we observe that the tweet embeddings are slightly better than the handcrafted features for a detection deadline of 1 min but no significant difference was found ($Z = 14.0, p = 0.169$). Between models LSTM-T and LSTM-L also no significant difference was found ($Z = 11.0$, $p = 0.171$). Finally, we find that the combination of linguistic and network features outperforms all other models for near real-time detection ($p < 0.05$). This model can classify articles as misinformation with an accuracy of 93.36% after 1 min.

Table 3. Misinformation detection accuracy and their standard deviation by doing 10-fold cross validation.

Detection deadline	# Snapshots	LSTM-N	LSTM-H	LSTM-T	LSTM-L	LSTM-ALL
1 min	1	76.83 ± 5.55	85.77 ± 4.40	87.59 ± 4.26	89.90 ± 4.35	93.36 ± 2.52
5 min	1	76.35 ± 4.79	83.27 ± 5.53	88.75 ± 2.95	90.10 ± 4.67	91.35 ± 3.65
15 min	1	79.04 ± 5.64	83.65 ± 4.32	87.88 ± 3.45	90.38 ± 3.57	92.31 ± 3.80
	3	81.35 ± 6.45	85.10 ± 3.95	88.17 ± 2.95	90.38 ± 6.94	94.04 ± 3.64
	15	81.63 ± 4.50	84.62 ± 3.01	86.92 ± 2.92	91.63 ± 4.26	93.27 ± 2.39
1 h	1	80.58 ± 3.79	84.23 ± 4.75	88.85 ± 3.45	91.25 ± 4.29	95.19 ± 2.47
4 h	1	81.44 ± 4.37	89.23 ± 3.18	92.79 ± 2.25	94.04 ± 3.07	95.48 ± 2.24
	4	85.96 ± 4.17	88.27 ± 3.35	90.67 ± 3.01	94.62 ± 3.14	95.77 ± 2.76
	16	83.65 ± 4.30	88.46 ± 4.39	77.12 ± 14.49	93.65 ± 3.47	93.46 ± 2.10
	48	78.17 ± 7.52	89.04 ± 3.92	77.40 ± 15.71	66.06 ± 13.31	80.96 ± 6.46
1 day	1	85.87 ± 4.87	91.92 ± 3.17	93.85 ± 3.47	95.10 ± 2.52	95.67 ± 2.29
3 days	1	86.73 ± 4.97	92.02 ± 3.50	93.94 ± 2.32	95.38 ± 2.75	95.38 ± 2.94
10 days	1	87.79 ± 5.64	92.02 ± 3.10	95.38 ± 1.71	95.00 ± 2.75	97.60 ± 1.44
	60	86.83 ± 4.15	81.15 ± 10.26	78.37 ± 13.95	69.62 ± 11.62	92.02 ± 5.57

6 Conclusion and Future Work

In this paper, we studied the effectiveness of network and linguistic features for the early detection of misinformation articles. We proposed a model that combines both feature spaces by utilizing a recurrent neural network for classification. Experiments demonstrated that this model can detect misinformation articles in near-real time with an accuracy of 93%. This, for example, could help fact-checkers to increase the efficiency and effectiveness in which they filter and verify the massive amount of articles posted on online social networks. Moreover, we showed that linguistic features outperform network features for early detection.

To substantiate the performance of our model we plan to perform experiments with different datasets and compare our model with other state-of-the-art detection methods. Furthermore, we have the following suggestions for future work:

– Design models that use dynamic detection deadlines so that a desired trade-off between accuracy and latency can be learned.
– Compare tweet volume-independent with volume-dependent detection models for near real-time detection.

References

1. Brena, G., Brambilla, M., Ceri, S., Di Giovanni, M., Pierri, F., Ramponi, G.: News sharing user behaviour on Twitter: a comprehensive data collection of news articles and social interactions. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 13, pp. 592–597 (2019)
2. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on Twitter. In: Proceedings of the 20th International Conference on World Wide Web, pp. 675–684 (2011)
3. Chen, T., Li, X., Yin, H., Zhang, J.: Call attention to rumors: deep attention based recurrent neural networks for early rumor detection. In: Ganji, M., Rashidi, L., Fung, B.C.M., Wang, C. (eds.) PAKDD 2018. LNCS (LNAI), vol. 11154, pp. 40–52. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04503-6_4
4. Chollet, F., et al.: Keras (2015)
5. Davis, C.A., Varol, O., Ferrara, E., Flammini, A., Menczer, F.: BotOrNot: a system to evaluate social bots. In: Proceedings of the 25th International Conference Companion on World Wide Web, pp. 273–274 (2016)
6. Effrosynidis, D., Symeonidis, S., Arampatzis, A.: A comparison of pre-processing techniques for Twitter sentiment analysis. In: Kamps, J., Tsakonas, G., Manolopoulos, Y., Iliadis, L., Karydis, I. (eds.) TPDL 2017. LNCS, vol. 10450, pp. 394–406. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67008-9_31
7. Ferrara, E., Varol, O., Davis, C., Menczer, F., Flammini, A.: The rise of social bots. Commun. ACM **59**(7), 96–104 (2016)
8. Godin, F., Vandersmissen, B., De Neve, W., Van de Walle, R.: Multimedia lab@ ACL WNUT NER shared task: named entity recognition for Twitter microposts using distributed word representations. In: Proceedings of the Workshop on Noisy User-Generated Text, pp. 146–153 (2015)
9. Guo, B., Ding, Y., Yao, L., Liang, Y., Yu, Z.: The future of misinformation detection: new perspectives and trends. arXiv preprint arXiv:1909.03654 (2019)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1725–1732 (2014)
12. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Kohita, R., Noji, H., Matsumoto, Y.: Dynamic feature selection with attention in incremental parsing. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 785–794 (2018)
14. Krishnan, S., Chen, M.: Identifying tweets with fake news. In: 2018 IEEE International Conference on Information Reuse and Integration (IRI), pp. 460–464. IEEE (2018)

15. Kwon, S., Cha, M., Jung, K.: Rumor detection over varying time windows. PloS One **12**(1), 1–19 (2017)
16. Kwon, S., Cha, M., Jung, K., Chen, W., Wang, Y.: Prominent features of rumor propagation in online social media. In: 2013 IEEE 13th International Conference on Data Mining, pp. 1103–1108. IEEE (2013)
17. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
18. Lewandowsky, S., Ecker, U.K., Cook, J.: Beyond misinformation: understanding and coping with the "post-truth" era. J. Appl. Res. Mem. Cogn. **6**(4), 353–369 (2017)
19. Lewandowsky, S., Ecker, U.K., Seifert, C.M., Schwarz, N., Cook, J.: Misinformation and its correction: continued influence and successful debiasing. Psychol. Sci. Public Interest **13**(3), 106–131 (2012)
20. Liu, Y., Wu, Y.F.B.: Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
21. Ma, J., et al.: Detecting rumors from microblogs with recurrent neural networks (2016)
22. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
23. Novak, P.K., Smailović, J., Sluban, B., Mozetič, I.: Sentiment of emojis. PloS One **10**(12), e0144296 (2015)
24. Pennycook, G., Rand, D.G.: Fighting misinformation on social media using crowdsourced judgments of news source quality. Proc. Natl. Acad. Sci. **116**(7), 2521–2526 (2019)
25. Pierri, F., Ceri, S.: False news on social media: a data-driven survey. ACM SIGMOD Rec. **48**(2), 18–27 (2019)
26. Shao, C., Ciampaglia, G.L., Flammini, A., Menczer, F.: Hoaxy: a platform for tracking online misinformation. In: Proceedings of the 25th International Conference Companion on World Wide Web, pp. 745–750 (2016)
27. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: a data mining perspective. ACM SIGKDD Explor. Newslett. **19**(1), 22–36 (2017)
28. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
29. Volkova, S., Shaffer, K., Jang, J.Y., Hodas, N.: Separating facts from fiction: linguistic models to classify suspicious and trusted news posts on Twitter. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 647–653 (2017)
30. Vosoughi, S., Mohsenvand, M., Roy, D.: Rumor gauge: predicting the veracity of rumors on Twitter. ACM Trans. Knowl. Discov. Data (TKDD) **11**(4), 1–36 (2017)
31. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
32. Wang, Y., et al.: EANN: event adversarial neural networks for multi-modal fake news detection. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 849–857 (2018)
33. Warriner, A.B., Kuperman, V., Brysbaert, M.: Norms of valence, arousal, and dominance for 13,915 English lemmas. Behav. Res. Methods **45**(4), 1191–1207 (2013). https://doi.org/10.3758/s13428-012-0314-x