




# Integrating Machine Learning Techniques in Semantic Fake News Detection

Adrian M. P. Braşoveanu<sup>1,3</sup>  · Răzvan Andonie<sup>2,3</sup>

Accepted: 3 October 2020 / Published online: 29 October 2020  
© Springer Science+Business Media, LLC, part of Springer Nature 2020

## Abstract

The nuances of languages, as well as the varying degrees of truth observed in news items, make fake news detection a difficult problem to solve. A news item is never launched without a purpose, therefore in order to understand its motivation it is best to analyze the relations between the speaker and its subject, as well as different credibility metrics. Inferring details about the various actors involved in a news item is a problem that requires a hybrid approach that mixes machine learning, semantics and natural language processing. This article discusses a semantic fake news detection method built around relational features like sentiment, entities or facts extracted directly from text. Our experiments are focused on short texts with different degrees of truth and show that adding semantic features improves accuracy significantly.

**Keywords** NLP · Semantics · Relation extraction · Deep learning

## 1 Introduction

Detecting fake news is an interdisciplinary problem, as it requires us to examine which methods were used to disseminate the news (e.g., social networks [53]), the links between the various actors involved (e.g., by using the information available in public Knowledge Graphs like Wikipedia), the propaganda tools (e.g., language can often be examined through the lens of semantics [6]) or even the geopolitics (e.g., as proven by the Cambridge Analytica scandal, some news might be targeted to some specific groups who might be more likely to respond to it). At a superficial level it is important to distinguish between satire and political weapons (or any other kind of weapons built on top of deceptive news) [8] or between the various news outlets that spread it, but when analyzing a news item it often helps to deploy a

---

✉ Adrian M. P. Braşoveanu  
adrian.brasoveanu@modul.ac.at

Răzvan Andonie  
andonie@cwu.edu

<sup>1</sup> MODUL Technology GmbH, Vienna, Austria

<sup>2</sup> Computer Science Department, Central Washington University, Ellensburg, WA, USA

<sup>3</sup> Electronics and Computers Department, Transilvania University of Braşov, Braşov, Romania

varied Natural Language Processing (NLP) arsenal that includes sentiment analysis, Named Entity Recognition Linking and Classification (NERLC [26]), n-grams, topic detection, part-of-speech (POS) taggers, query expansion or relation extraction [65]. NLP tools are often supported by large Knowledge Graphs (KGs) like DBpedia [35] which collects data about entities and concepts extracted from Wikipedia. The extracted named entities and relations will be linked to such KGs whenever possible, whereas various sentiment aspects, polarity or subjectivity might be computed according to the detected entities. Features like sentiment, named entities or relations render a set of shallow meaning representations, and are typically called *semantic features*. In contrast, POS or dependency trees render *syntactic features*.

The underlying assumption made by most models used for detecting fake news is that the title and style of an article are sufficient to identify it as fake news. This is mostly true for news that originate from verifiable bad sources, which is rarely the case anymore. Therefore, we think that taking a holistic approach, that includes a machine generated Knowledge Graph (KG) [43] of all the stakeholders involved in the various events we are interested in is absolutely needed. Such a holistic approach includes methods which can generate and learn graphs of entities associated to fake news.

Our contribution is a method used to integrate semantic features in the training of fake news classifiers. The goal is to show how to use semantic features to improve fake news detection. For this, we compute semantic features (sentiment analysis, named entities and relations) which will be added to a set of syntactic features (POS - part-of-speech and NPs—Noun Phrases) and to the features of the original dataset. On the resulted augmented dataset we apply various classifiers, including Deep Learning (DL) models like Long-Short Term Memory (LSTM), Convolutional Neural Network (CNN), and Capsule Networks. For the Liar data set [62], using semantic features improves the fake news recognition accuracy significantly.

The paper is organized as follows. Section 2 presents the problem description. Section 3 surveys the most recent results in fake news recognition. Section 4 introduces our approach for building machine generated KGs for semantic fake news detection. Section 5 describes the experimental results. The paper is concluded in Sect. 6.

## 2 Problem Description

It is difficult to establish with certainty the truthfulness of any kind of declaration. Facts-based declarations might be easier to check as they simply require some fast queries on a Knowledge Graph or search engine, whereas political declarations might be more context-dependent and require more fine-grained semantic information. This is the main reason why this field of study became so important in the last decade, once social media analysis at scale became a reality.

### 2.1 Background

There are various definitions of fake news. Most of them are based on Alcott and Gentzkow's paper [4] about the impact of fake news on the 2016 US Election.

**Definition** (based on [4]). *A news item or a part of a news item will be considered fake if it can be verified that its content is false.*

Fake news detection can be considered a part of a larger class of tasks focused around fact checking [58]. Some related tasks include fact verification under open-world assumption, common sense reasoning (e.g., understanding the arguments required to support certain premises), subjectivity and emotive language detection (e.g., predicting whether the document originates on websites well-known for spreading hoaxes or propaganda), deceptive language detection, rumor detection (e.g., identification and classification of unverified reports), speaker profiling or click bait deception.

From the point of view of Machine Learning, fake news detection is a *binary classification* (e.g., true or false) or *multi-class classification* (e.g., when multiple degrees of truth are taken into account) problem. Input data includes a statement and some information about it (e.g., speaker, location, party affiliation). The expected output is a binary label or a more fine-grained label (e.g., true, mostly true, etc).

In order to perform semantic fake news detection, some additional statements like the past truth history of a speaker or the relations between speakers and publishers should be considered if possible. The idea of using past inaccuracies for each speaker was introduced with the Liar data set [62] and named credit history, but it is rarely used in practice.

**Definition** (based on [62]). *Credit History (CH) is the historical count of false (or provably untrue) statements for an actor.*

A credit history score can also be replaced by a single aggregated count of all the untrue values. Such credit scores allows us to understand diverse perspectives when analyzing news and helps determine which person or group might benefit from spreading certain news. An earlier iteration of this idea was also explored in the context of social media networks: credibility propagation [27].

## 2.2 Problem Statement

Expanding upon the idea of semantic fake news detection, it is important to look closely at two concepts: the ideas of credit score and degree of truthfulness

Generalizing the idea of credit score we will be able to define it as a graph that models all the relations between the various entities present in a statement or document. Such a graph can be seen as a Knowledge Graph [18] if the information is fine-grained and can lead to some good inferences.

**Definition** *A credit history graph is a graph that contains all the entities, their credit histories and links between them as they are available from a Knowledge Graph (KG) or generated from a collection of texts.*

Credit history can be extended to cover all important entities related to a fake news (e.g., speakers, publishers, etc). By doing this, the resulting graph will be similar to the mention-entity graphs used in Named Entity Linking processes which include all the links between the various entities from a text. The main difference will be that the extended credit history scores can be imposed over the mention-entity graphs becoming weights. Such scores can be created automatically through similar processes (e.g., counting, aggregation, rule-based generation) like those applied for the creation of machine generated Knowledge Graphs [19].

Credit history itself however is not enough, as it merely provides us with proxy indicators about the truthfulness of some of the actors involved in a statement. Due to the fact that information about these actors is generally incomplete regardless of the used Knowledge Graph (KG), we can not expect these indicators to be reliable predictors. Adding some fine-

grained semantic information like sentiment, entities or relations should help us build better credit histories. This kind of semantic information can also be considered an alternative to the credit history. Such features can be extracted from both traditional KGs (e.g., DBpedia, Wikidata), as well as from the texts themselves.

**Definition** *Relational features include all the features extracted directly from the texts or the named entities detected in them through the exploitation of Knowledge Graphs.*

While we focus on extracting all the needed features directly from the text, the Tri-Relationship framework described in Shu's paper [54] also deserves a mention here, even though it is focused on the objects involved in distributing the news (e.g., people, organizations). All the mentioned approaches share the idea of enriching the fake news text with a set of annotations, in order to provide some context. This naturally leads to our main research question:

*What are the most useful semantic features that can be used to improve fake news detection?*

Ideally, such features should be integrated into the neural models, whenever possible. Today, due to the cost of developing good semantic systems, some of these features might come from various external tools. The semantic features need to be selected according to the task and dataset at hand. If the task refers to the detection of fake news as spread by people via their statements, then the main entities we will be interested in might include people, organizations, locations and events.

The recent success of language models based on Transformers [59] like BERT [15] or RoBERTa [39] might potentially change the entire field of Natural Language Processing. Such models have been shown to perform well on a variety of tasks including sentiment analysis, Named Entity Recognition (NER), semantic role labelling or dependency parsing. This is partially due to their ability to pick up various language phenomena like direct objects, noun modifiers or coreferents [13]. Examining questions about semantic features or pre-Transformers NLP models might not lead us to the good results obtained with the BERT-inspired language models. However, considering the fact that a lot of these new language models are quite large, expensive to train and potentially damaging to the environment [57], we think this is a task that is worth pursuing. Even more importantly, Transformers are not the only models that are currently showing good results in NLP, as bidirectional LSTMs [33], Capsule Networks [51] or variational autoencoders like Vampyr [21] were also shown to provide good results for a fraction of the cost and time required to train Transformer models.

### 3 Related Work

Several articles per week were published about fake news in the last years, around 40% of them being dedicated to actual models (neural nets or others) developed for classifying, creating or examining fake news. The interest in this topic has actually increased a lot after the mid-2018 boom of NLP language models. The large majority of these articles are simply surveys or political studies in which the propagation of fake news plays a central role. It has to be noted that this type of disinformation does not extend only to political news, but to any kind of news (e.g., sports, entertainment, culture), including scientific publications, therefore many articles simply examine various facets of it. Nevertheless, due to the rapid expansion of the literature on this subject, we have chosen to focus this section on several aspects we considered important, mainly detection methodology and models.

An exploration of the fake news phenomena during more than a decade (2006–2017) was built around Twitter rumor cascade by a series of social scientists [61]. Multiple surveys (e.g., [53,66]) are focused on building various fake news classifications. A good survey of the various fake news types (e.g., visual-, user-, post-, network-, knowledge-, style-, stance-based) and the typical methods that were used for their detection can be found in [44]. A recent survey [66] identifies not only the major types of fake news (fake news, biased/innaccurate and misleading—each with its own subcategories), but also goes on to classify the various actors involved in spreading the news, their motives, as well as the various methods that can be used to combat the propagation of false information. Another survey shows a data mining perspective for the spread of fake news in social media [53]. Rubin [49] classifies deceptive articles into three large classes (serious fabrications, hoaxes, humor or satire) and defines a set of criteria for creating a good text corpora for fake news detection [48], namely the fact that such data sets should only contain verifiable facts that happened in a certain interval and were reported using similar style though with various degrees of cultural influences. Any such corpora should only focus on text-only items as they would be easier to process.

A recent survey on the role of fake news detection in decision making [23] identifies three large areas of interest: taxonomy of false information (e.g., rumor, fake news, hoax, misinformation), Machine Learning (ML) techniques (e.g., supervised, unsupervised, semi-supervised) and Deep Learning (DL) techniques (e.g., supervised, unsupervised, etc).

Most of the time, simply analyzing will not yield good results, therefore some models also incorporate some data about the networks (e.g., social media, organizations) through which the news spread. Ruchansky [50] proposes a CSI model which stands for Capture, Score and Integrate, therefore combining information on the temporal activity of the users, their behavior, and a classifier. Similar ideas, but oriented towards identifying the geography of fake news, were described in [17]. An approach that is closer to verification is presented in [5], the focus being on a set of annotated statements is verified across top news sources (e.g., New York Times, CNN, The Guardian, etc). An approach that checks various parts of articles is also presented in [55] and uses 3HAN, a Hierarchical Attention Network (HAN) network with three layers.

A model for early detection of fake news based on news propagation paths is described in [38] and is based on a hybrid time-series classifier that contains both Recurrent Neural Networks (RNNs) and CNNs. Wu [63] assumes that intentional fake news are typically manipulated to look like real news. He built a classifier based on social media propagation pathways using LSTM-RNN and embeddings. Vo and Lee [60] focus on the story told by fake news URLs and the co-occurrence of various entities through such links. Their GAU model combines Guardian-Guardian SPPMI matrix, Auxiliary information and an URL-URL SPPMI matrix and is shown to outperform a series of baselines inspired from recommendation systems (e.g., Matrix Factorization or CoFactor).

A set of LSTMs is used for performing a multi-source multi-class fake news detection (or MMFD) in [28]. The advantage of this method is the multi-source fusion of the MMFD framework, since it can determine various degrees of fake news. The accuracy of the approach is not very high, but given the fact that it combines three large components (automated feature extraction, multi-source fusion and fakeness discrimination) it is promising. Aghakhani [2] showed that a Generative Adversarial Network (GAN) [20] can perform relatively well for detecting deceptive reviews.

Latest NLP multi-task learning models are usually built around Transformer architectures [59]. An early paper from the WSDM fake news challenge [64] shows that both a number of models that use BERT architecture or embeddings perform quite well. Several of the systems submitted for the SemEval 2019 Task 4 Hyperpartisan News Detection have also

used Transformer architectures [29]. A two-stage model that uses BERT architecture [15] and relies on both information and attention mechanisms is presented in [37] and achieves an accuracy that is higher with more than 10% than the original baseline. Due to the rise of generative language models like BERT or GPT-2 [56], neural fake news took the media and public by storm and caused a lot of discussions about ethics in AI development. A staged release strategy like the one used for GPT-2 was essential in order to extend the time needed for experiments and minimize potential damage. Another strategy to defend against neural fake news is to train the model against itself, similar to how GAN works as described in [67]. While neural fake news generators can be seriously damaging, it is worth considering them in order to increase the size of training data sets.

Including all the various articles that showcase the latest advances in sentiment analysis or Named Entity Linking or relation extraction is beyond the scope of this paper. However, we think it is important to note at least several surveys that cover these topics, as these areas are very important for the analysis of fake news. A good explanation on the relations between Named Entity Linking, Information Extraction and Knowledge Graphs can be found in [3]. Good sentiment analysis engines are difficult to build as they are generally umbrella technologies that are built on top of a chain of tools that might include POS taggers, Named Entity Linkers or relation and aspect extractors as described in [10]. A survey that includes details about the most common Deep Learning algorithms to be used for sentiment analysis or named entities extraction/classification, can be found in [65].

An early version of the technique presented in this article [9] focused on balanced classification. The current article describes the more general unbalanced classification problem while updating all the models in line with current literature and significantly expanding the experimental section (e.g., provides results on multiple data sets, more in-depth explanations of the results).

## 4 Methodology

In order to fully exploit the relations between the entities mentioned in a news statement, our procedure includes the following steps:

- **Metadata collection** The first step is to simply collect the sentiment, entities and additional metadata available from third party tools.
- **Relation Extraction** A second pass will collect both (i) the general relations found in a KG, and (ii) those computed from the current texts.
- **Embeddings** Last step refers to the adaptation of various neural models (e.g., by adding a layer of embeddings) for improving fake news detection.

The features included in the last step will be only internal, whereas the features included on the other steps can also be external. The entire process is illustrated in Fig. 1.

The intuition behind the current data modeling that led to the additional semantic features is that by adding extracted entities and making a clear distinction between direct and indirect speech, we can create the premises for more sophisticated analysis that may pinpoint the personal history of a speaker with both the issue at hand (or subject), as well as with all the parties involved in the respective issue. If such an analysis is extended, down the road, it should also be possible to identify more obscure details about a speaker, for example if a named entity labelled as politician follows the party line or not. In other words, it opens up the possibility of using the graphs to peak behind the scenes of various declarations.

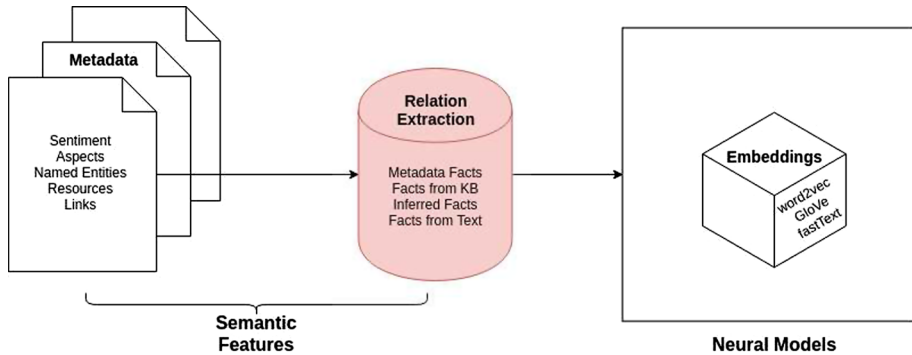


Fig. 1 External and internal semantic features for neural network models

#### 4.1 Metadata Collection Pipeline

Our pipeline for generating metadata has following components:

- *Sentiment Analysis (SA)* Sentiment annotations can exist on multiple levels: (i) document; (ii) sentence; (iii) aspect-based [65]. Current state-of-the-art systems are typically aspect-based, therefore all the aspects of the entity features can get an estimate of the sentiment value. Since our data set (the Liar data) contains short statements, we use aggregated sentence level sentiment polarity and subjectivity values computed with the TextBlob<sup>1</sup> library.
- *Named Entities (NE)* Since the results for NE extractions are typically good enough [26], almost any modern NLP library can be used for this task. Both Spacy and Stanford NLP libraries have good performance.
- *Named Entity Links (NEL)* Generally NERLC (NER+linking and classification) tasks are considered more complicated and typically require dedicated NEL engines [26]. Any good NEL engine can be used for this task. We use a Python wrapper<sup>2</sup> around a DBpedia Spotlight [14] instance that was run locally using the English model.<sup>3</sup>

#### 4.2 Relation Extraction

Instead of using existing solutions, we develop a simple *Relation Extraction (REL)* component that queries DBpedia. Where possible, the existing entities are enriched with additional data obtained via a SPARQL query from DBpedia. This is particularly important in order to discover more relations between a speaker (which we will call *source entity*) and his/her subject (which we will call *target entity*). We consider two types of relations:

- (i) extracted directly from the provided news statements by defining the types of relations we are interested in via POS tags (for example, for extracting relations between two entities we will generally be interested in NP–V–NP chains—a verb between two proper nouns, whereas additional relations for an entity can be added by extracting S–V–O triplets);

<sup>1</sup> <https://github.com/sloria/textblob>.

<sup>2</sup> <https://github.com/aolieman/pyspotlight>.

<sup>3</sup> <https://github.com/dbpedia-spotlight/dbpedia-spotlight-model>.

- (ii) extracted from the DBpedia Knowledge Base (e.g., if `dbr:Donald_Trump` mentions `dbr:Barack_Obama` in a document, all the triples that belong to these entities are extracted from DBpedia and a subset of common links like `dbo:orderInOffice` or `dbo:President` is identified).

The machine generated KG includes all the DBpedia triples that belong to the entities collected from the data set. The relations extracted from text are schemaless, whereas the relations extracted from KG are grounded to a schema (e.g., DBpedia ontology). This component is implemented with the Python libraries `RDFLib`, `SPARQLWrapper` and `Spacy`.<sup>4</sup>

A more complex graph can also be built if we consider a split of the entity links by topics. In such a scenario a speaker can be a friend of his subject (e.g., though only in the context in which the subject is a person or organization) when he is discussing one topic, but also an enemy when discussing another topic. This closely models the real life allegiances from politics or business, but not necessarily from other fields like sports or culture, therefore it will be left outside the scope of this publication as it requires a more complex analysis.

If the data sets include full texts and not just short sentences, the semantic modelling and the associated graphs will quickly become complex. In such scenarios it is best to use a standard like the RDF Data Cube Vocabulary (QB),<sup>5</sup> as it is also important to keep the context in which each statement was made, therefore multiple counts can be modelled as observations with a source (speaker) and target (subject of the speech). It has to be noted that in order to reduce the size of the graph it is better to prune entities according to their type (e.g., keep only people, organizations, locations and events).

### 4.3 Embeddings

Shallow neural architectures that learn word embeddings from distributional semantics (e.g., continuous bag of words architectures like `Word2Vec`, `GloVe` or `fastText` [42]) have been successfully applied to classic NLP problems [65], and should be an integral part of any NLP architecture. Such architectures generally provide fast computation times and lead to good results due to the fact that they capture relational similarities. These embeddings can be applied to a wide-range of tasks (word analogy, human similarity judgements, sentiment analysis, word prediction, hybrid recommendation, etc.).

If the used corpora is clean and large enough (several tens of thousands of examples [42]), embeddings can be an ideal solution for building baselines. Only the most used (`word2vec`, `GloVe`, `fastText`) pre-computed embeddings were included for the top 60k English words. The component that loads them uses negative sampling and `Glove` with a fixed size of 300 (`glove.840B.300d`). The Keras API offers the possibility to add an embeddings layer to a neural network. This layer can be used for: (i) learning and saving the embeddings together with the word vectors; (ii) loading pre-trained embeddings. In all our DL models, we place such a layer after the inputs and use it for loading embeddings. Such a layer is effective especially when the number of training examples is relatively small [45].

<sup>4</sup> <https://spacy.io/>.

<sup>5</sup> <https://www.w3.org/TR/vocab-data-cube/>.



## 5 Experiments

The success of our approach depends on a series of components for extracting sentiment scores, named entities, or relations. Therefore, if those components do not perform well, the whole approach will be flawed. First, we would like to find out if such an approach is valid. Therefore, missing a named entity from a statement might not be extremely important at this stage. If the approach proves to be valid, then further work needs to include additional evaluations for all the components in the pipeline, or at least some of their performance scores (when available).

### 5.1 Data

We use several similar data sets that contain annotated data with multiple degrees of truth for our experiments.

The **Liar** data sets [62] contains politics-related short texts extracted from the Politifact API and classified based on the degree of truth, while also offering credit histories that track the accuracy of the speaker statements. The data set is split into three partitions (**train**, **test** and **validation**) and includes six classes that need to be predicted: **True**, **Mostly True**, **Half True**, **Barely True**, **False** and **Pants-on-fire**. The initial paper about the Liar data set [62] identified SVMs as best classical models and CNNs as the best Deep Learning classifiers. A follow-up paper [40] indicates that LSTMs would be even better. Since our focus is not on credit history (five counts for all the classes that are not True including the score for the current statement) but on the impact of the relational features, we do not reproduce those results and do not compare with them (Table 1).

The **Politifact** data set [46] also contains short texts extracted from the Politifact API, but, as opposed to the Liar data set, does not provide any additional features except for the identity of the speaker. The data set is split into two partitions (**train** and **test**) and includes the following classes that need to be predicted: **True**, **Mostly True**, **Half True**, **Mostly False**, **False** and **Pants-on-fire**. The paper that introduced the data set [46] shows that LSTMs with text and no additional features perform best in both 2-classes (when the data is split only in True and False) and 6-classes (when keeping the original annotations) scenarios. On the associated website, there are an additional five thousand texts from Politifact sister websites that were not annotated and which can be used if there is a need to extend this data set.

As it can be seen the two data sets are quite similar. In fact, except for one class that has a different name, even the labels are exactly the same (**True**, **Mostly True**, **Half True**, **False** and **Pants-on-fire**). The classes with a different label (**Barely True** or **Mostly False**)

**Table 1** Data set statistics

| DS-partition     | True | Mostly true | Half true | Barely true | False | Pants on fire | Totals |
|------------------|------|-------------|-----------|-------------|-------|---------------|--------|
| Liar-train       | 1683 | 1966        | 2123      | 1657        | 1998  | 842           | 10,269 |
| Liar-test        | 211  | 249         | 267       | 214         | 250   | 92            | 1283   |
| Liar-valid       | 169  | 251         | 248       | 237         | 263   | 116           | 1284   |
| Politifact-train | 499  | 527         | 530       | 364         | 463   | 193           | 2756   |
| Politifact-test  | 227  | 252         | 217       | 169         | 160   | 50            | 1075   |

Number of examples for the six major classes in Liar and Politifact data set

however clearly contain similar examples. Due to this, we considered the two data set as being siblings, and ideal for our experiments. Both data sets display a low number of **pants-on-fire** examples in all their partitions. Both data sets contain multiple examples for a set of speakers, but Liar also contains some examples taken directly from political campaign statements were the speakers might not be clear or easy to identify. The length of the examined statements is somewhat similar being around 18 words and 105–110 characters (e.g., Liar has an average of 17.9 words, whereas Politifact has an average of 18.32). Since classic tweet length was around 140 characters, we can easily consider these texts as being somewhat equivalents to old tweets, the main difference being the lack of specific language and shortcuts (e.g., no emojis or RT handles are included in any of these data sets).

Taking into account the various combinations of text, original data set features and relations, we can describe four cases for the experiments. The texts themselves (named **text (T)**) are simply statements that are taken out of their original context. The features included in the original data set (**text+attributes (T+A)**) contain information about the subject, speaker (including his job title, state and party affiliation), as well as credit history, and the context (the speech's location). The set **text+relations (T+R)** has semantic features (sentiment polarity, sentiment subjectivity, entities, links, and relations), syntactic features (NP), and the aggregated score of the credit history counts. The features included in the **T+R** data set are all extracted directly from the statements—there is no need to use the full text of the articles to compute them. This is an important detail, since this operation can always be performed if we have a good set of tools for metadata generation, even when the full articles are no available. The last set of features (identified as **all (ALL)**) includes all available features. Since Politifact does not really have additional features (the only feature beside the text is the speaker name), our experiments discard the **text+attributes (T+A)** case. This makes it easier to compare results on the two data sets.

The classes are unbalanced. In Tables 2 and 3 we report the test set accuracy scores for all considered models and additional features for both data sets (Liar and Politifact).

**Table 2** Accuracy for the test set runs on the Liar dataset

| Model                           | T            | T+R   | ALL          |
|---------------------------------|--------------|-------|--------------|
| <i>Classic ML</i>               |              |       |              |
| Multinomial Naive Bayes         | 0.224        | 0.244 | <b>0.262</b> |
| SGDClassifier                   | 0.239        | 0.235 | <b>0.255</b> |
| Logistic regression (OneVsRest) | 0.240        | 0.260 | <b>0.273</b> |
| Random forest                   | <b>0.215</b> | 0.215 | 0.212        |
| Decision trees                  | 0.226        | 0.249 | <b>0.262</b> |
| SVM                             | 0.255        | 0.275 | <b>0.294</b> |
| <i>Deep learning</i>            |              |       |              |
| CNN                             | 0.241        | 0.270 | <b>0.289</b> |
| BasicLSTM                       | 0.245        | 0.289 | <b>0.326</b> |
| BiLSTM attention                | 0.419        | 0.448 | <b>0.499</b> |
| GRU attention                   | 0.450        | 0.496 | <b>0.539</b> |
| CapsNet                         | 0.565        | 0.598 | <b>0.649</b> |

The best results are presented in **bold**. **T** stands for text and **R** for relations

**Table 3** Accuracy for the test set runs on the Politifact dataset

| Model                           | T            | T+R   | ALL          |
|---------------------------------|--------------|-------|--------------|
| <i>Classic ML</i>               |              |       |              |
| Multinomial Naive Bayes         | 0.263        | 0.295 | <b>0.296</b> |
| SGDClassifier                   | 0.262        | 0.294 | <b>0.295</b> |
| Logistic regression (OneVsRest) | 0.246        | 0.269 | <b>0.269</b> |
| Random forest                   | <b>0.244</b> | 0.229 | 0.229        |
| Decision trees                  | 0.246        | 0.269 | <b>0.270</b> |
| SVM                             | 0.262        | 0.281 | <b>0.282</b> |
| <i>Deep learning</i>            |              |       |              |
| CNN                             | 0.203        | 0.231 | <b>0.244</b> |
| BasicLSTM                       | 0.245        | 0.287 | <b>0.282</b> |
| BiLSTM attention                | 0.371        | 0.422 | <b>0.422</b> |
| GRU attention                   | 0.415        | 0.451 | <b>0.452</b> |
| CapsNet                         | 0.473        | 0.523 | <b>0.524</b> |

The best results are presented in **bold**. **T** stands for text and **R** for relations

## 5.2 Models

We start by testing several “classic” models [24] that were built with scikit-learn (Table 2). For these models, using the relational features (**T+R**) shows some improvements, typically 2–3% above the original features (**T+A**) of the data set. However, the best score are far from optimal. Logistic regression and decision trees scores prove to be quite similar for all the three runs, while simultaneously being the worst scores. We notice a single case (the random forest classifier) in which the added relational features do not yield improvements over a run with only the original text. The best “classic” ML classifier proves to be the SVM, confirming the results from [40].

In the second phase, we test several DL models. The DL models are built with Keras [12] and TensorFlow [1], and use hot encoding of the class labels. For the DL models, the reported evaluation metric is accuracy with Adam optimizer [32].

The following DL classifiers are used, all of them with categorical crossentropy loss function and Adam optimizer:

- CNN—is a simplification of the models described in [31,40]. It contains an embedding layer with dropout set to 0.2, a Convolution1D which learns how to filter groups of words, a GlobalMaxPool layer, as well as a basic hidden layer (dense, dropout set to 0.2 and relu activation function). The result is projected on a single unit output layer squashed with a softmax.
- BasicLSTM—is a simple LSTM with dimension 300, a GlobalMaxPool layer, spatial dropout set at 0.2 and dense layers with softmax activation. Some of the hyperparameters include batch size of 256, epochs set to 20 and learning rate set to 0.001.
- BiLSTM [11]—a bidirectional LSTM (CuDNNLSTM) with attention, dropout and recurring dropout set at 0.25 and a dense layer with softmax activation. We used same hyperparameters like the previous model.
- GRU [25]—a GRU with attention, otherwise similar to the previous BiLSTM model. We used same hyperparameters like the previous model.
- CapsNet model represents a simplified version of the models described in [16,30]. It uses a Capsule layer instead of the GlobalMaxPool layer used in the other models described

here. However, instead of using the Convolution+ReLU technique described in the article, we use a Bidirectional GRU with dimension 128, ReLU activation, dropout and recurrent dropout set to 0.25. The result is projected on a single unit layer squashed with a sigmoid. Some of the hyperparameters include batch size of 256, learning rate of 0.001, number of capsules 10 with dimension 16 and 5 routings. Training took around 5 epochs.

Regardless of the model, some basic preprocessing steps have been performed. These steps included text cleanup, stopwords removal and tokenization. Labels were encoded using LabelEncoder. The DL models have also included additional steps like turning the texts into sequences and padding.

All the DL models, besides the TextCNN and BasicLSTM, use Glove embeddings (more specifically—glove.840B.300d). We did not perform additional tuning of the DL models. We noticed that the embeddings for the most used 60k words from the English language have almost no effect on the results. The input vectors were loaded using Keras's embeddings layers which is defined as the first hidden layer of a network. For the DL experiments, we used whenever possible pre-trained models. Of course, fine-tuning the architectures may improve these results. We used same batch size and learning rate for all models. Whenever possible, we also tried to train for a similar number of epochs.

Since developers quite often stack different techniques, it is sometimes difficult to understand what a certain technique adds to a model. The study in Table 4 offers some explanation on which techniques might help when solving multi class classification problems focused on short texts. The simple models (TextCNN, BasicLSTM) do not include embeddings, therefore their results can easily be explained by the current study. For the more complex models (BiLSTM, GRU with Attention), the better performance is also due to embeddings and attention mechanisms. As it can easily be observed, extracting relational attributes can yield better results than using the original features, but when combining both set of features the results improve significantly.

Sometimes for NLP models it is enough to simply use the text and embeddings. This does not mean that the quality of the embeddings or the quality of the preprocessing will be the only factors affecting the output. In fact the words themselves will play a role in the result in this case. Such contributions can generally be understood through the explanations provided by libraries like Lime [47] or Shap [41] today. After collecting some random training samples that have roughly a quarter of the full training sets from each of the two data sets, we have used Shap to produce a series of graphics with 20 word features that are likely to have an impact on an LSTM model. Interestingly enough, around half of these words were found in multiple samples in both data sets. Such words that carry some weight include *percent*, *president*, *government*, *health*, *taxes* or president names. Figure 2 shows the word features that were rendered as important for the Liar data sets. Numbers were removed from this list, as we considered that they will not necessarily repeat themselves as they will be more context-dependent. Even more interesting, the word *percent* has been the top contender in both data sets. This suggests that fake news are likely to contain not just some information about a

**Table 4** Ablation study

| Model               | Accuracy | $\Delta$ |
|---------------------|----------|----------|
| LSTM with text only | 0.245    | -0.081   |
| + attributes        | 0.284    | -0.42    |
| + semantics         | 0.287    | -0.39    |
| ALL                 | 0.326    | 0        |

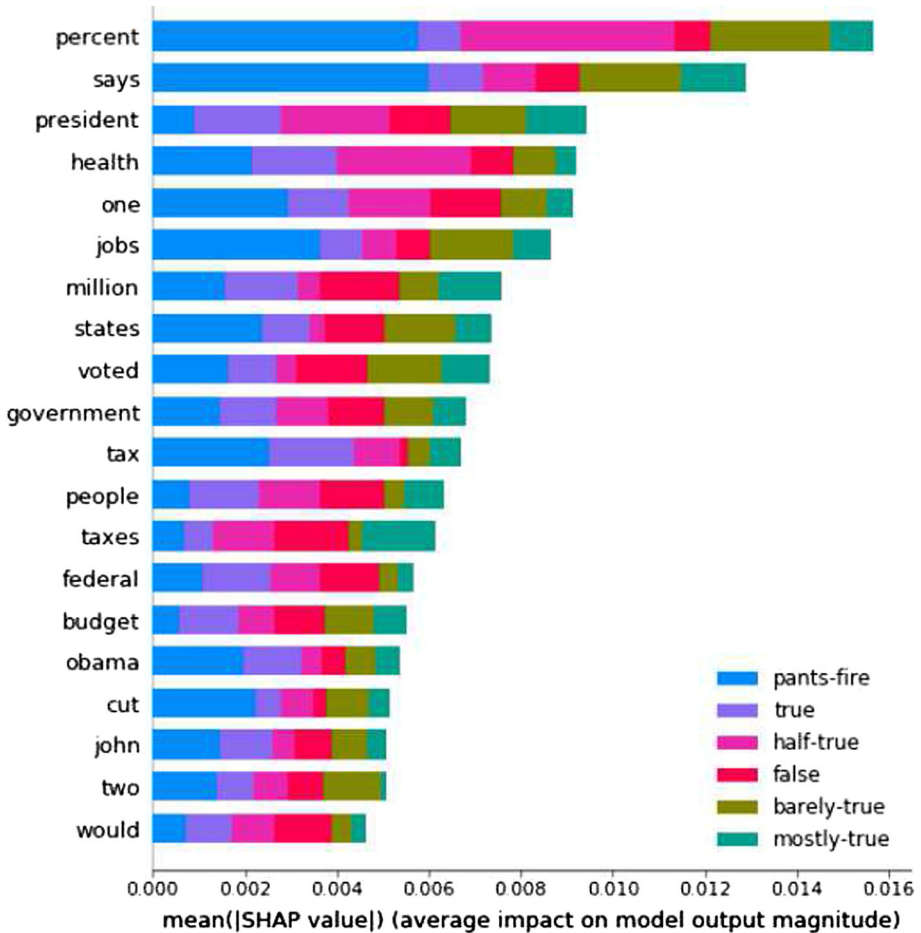


Fig. 2 SHAP explanations for a random sample from the Liar data set showing the top 20 word features with an average impact on model output magnitude

president or government, but also various indicators. This also suggests that it is necessary to use good preprocessing techniques. It is important to note that the various methods to measure feature contributions can lead to different results, therefore we consider the results provided by Shap or Lime more as guidelines to help our future developments.

### 5.3 Discussion

We note that all the DL models obtain better scores than the classic models with the same features. While the current literature is mostly focused on CNNs and basic LSTMs, we observe that attention models and CapsNet models performed best. For all DL models, adding our features results in an accuracy increase of up to 4.2% without any additional techniques like embeddings or attention on unbalanced classes. After adding embeddings and attention (e.g., BiLSTM model) scores again improve significantly (more than 10%, as it can be seen by comparing the basicLSTM score with later models that also contain these techniques).

As noted in [9], in all cases, relational features (**T+R**) perform better than the original features of the data set (**T+A**), which suggests that in some cases it might be enough to simply collect texts and build the rest of the features from metadata. While this paper represents an expansion of the previous one, this observation still stands.

We have not repeated all feature combinations from the original data sets as presented in Wang [62], Long [40] or [46], but rather took the best feature combinations found in those papers and added new combinations based on the relational features proposed by us.

The scores obtained by us for SVMs, basic CNNs and LSTMs confirm their results. Using relational features (sentiment, recognized named entities, named entities links, relations) together with syntactic features (NP), it is already possible to beat the current baselines at a comfortable distance, even without using advanced architectures. It is even possible to use only these semantic and syntactic features, instead of the original ones, and the scores will still be better than the baselines. Taking this into account, we think it would actually be preferable to use the method described in this paper in order to build reliable baselines.

As expected, the results for **T+R** and **ALL** were quite similar for the Politifact data set, as this data set does not contain any extra features besides the speaker. The fact that overall the results for Politifact were also somewhat lower than the results obtained for Liar data set is also in line with current literature results.

We tried to minimize the number of input features. Depending on the length of the text and number of entities involved, the number of additional features can be increased—which may lead to some increase in the overall performance. The most important thing when using our technique is to select the appropriate additional features that can lead to performance improvements. As it can be seen, even selecting several features like relations, sentiment and entities, can lead to significant improvement.

Even without any additional features, it is important to remember that the words included in the news statements themselves will carry some weight, as it can easily be seen in Fig. 1.

## 6 Conclusions

While the literature on fake news detection is increasing at fast pace, the accuracy of the various models greatly varies depending on the data sets and the number of classes involved. In our view, good models should be adaptive and should not require a lot of fine-tuning on data sets. According to our results, by also considering relational features like sentiment, named entities or facts extracted from both structured (e.g., Knowledge Graphs) and unstructured data (e.g., text), we generally obtain better scores on most classifiers. Good text preprocessing techniques, as well as of the associated embeddings, prove to be very important to any model, as even in the case of a simple model with no additional features (e.g., BasicLSTM with no features), the word features themselves will carry certain weights which can be visualized with modern libraries for explaining predictions like Shap.

Currently, most models are based on word embeddings, even though phrases and multi-words expressions perform better for longer texts. This is due to the fact that the language used in a fake news article may differ from the language used in a normal article, as it is often needed to reinforce certain claims.

Since the classes of the two data sets are quite similar, another future work direction is the creation of a large super data set for fake news. Around five thousands statements that were not yet fully annotated are still available on Rashkin's Politifact web page [46]. Several

other data sets could be re-annotated using this scheme. Having a larger data set will help improve the results on long term. This also clarify if the techniques presented here can work on different domains, as the two datasets we have used are rather similar and can even be merged as a single dataset.

Ultimately, we think, that the value of this method does not lie in the fact that it helps us build the best fake news classifier, but rather in helping us quickly build reliable baselines upon which we can improve. Taking this into account, we will continue building models, datasets and experimental pipelines for fake news in the near future.

Some future investigation areas include exploiting these relational features together with graph neural networks, like the recently developed R-GCN [52] or using a single multi-head attention architecture [59] to generate all the semantic features. In the context of Transformers, it would be important to understand what is the best strategies for transfer learning and knowledge distillation, as these can be helpful also for other Deep Learning models. Another interesting direction is to use semantic features for detecting fake reviews. While this is somewhat similar to the fake news detection, the goal there is to detect fake accounts on websites like TripAdvisor or fake authorships.

## References

1. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, Kudlur M, Levenberg J, Monga R, Moore S, Murray DG, Steiner B, Tucker PA, Vasudevan V, Warden P, Wicke M, Yu Y, Zhang X (2016) Tensorflow: a system for large-scale machine learning. CoRR. [arXiv:1605.08695](https://arxiv.org/abs/1605.08695)
2. Aghakhani H, Machiry A, Nilizadeh S, Kruegel C, Vigna G (2018) Detecting deceptive reviews using generative adversarial networks. CoRR. [arXiv:1805.10364](https://arxiv.org/abs/1805.10364)
3. Al-Moslimi T, Ocaña MG, Opdahl AL, Veres C (2020) Named entity extraction for knowledge graphs: a literature overview. IEEE Access 8:32862–32881. <https://doi.org/10.1109/ACCESS.2020.2973928>
4. Allcott H, Gentzkow M (2017) Social media and fake news in the 2016 election. J. Econ. Perspect. 31(2):211–36
5. Atanasova P, Nakov P, Márquez L, Barrón-Cedeño A, Karadzhev G, Mihaylova T, Mohtarami M, Glass JR (2019) Automatic fact-checking using context and discourse information. J Data Inf Qual. <https://doi.org/10.1145/3297722>
6. Barrón-Cedeño A, Martino GDS, Jaradat I, Nakov P (2019) Propopy: a system to unmask propaganda in online news. In: The 33rd AAAI conference on artificial intelligence, AAAI 2019, the thirty-first innovative applications of artificial intelligence conference, IAAI 2019, the ninth AAAI symposium on educational advances in artificial intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27–February 1, 2019, AAAI Press, pp 9847–9848. <https://aaai.org/ojs/index.php/AAAI/article/view/5061>
7. Bender EM, Derczynski L, Isabelle P (eds) (2018) Proceedings of the 27th international conference on computational linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20–26, 2018, association for computational linguistics. <https://www.aclweb.org/anthology/volumes/C18-1/>
8. Berghel H (2017) Lies, damn lies, and fake news. IEEE Comput 50(2):80–85. <https://doi.org/10.1109/MC.2017.56>
9. Brasoveanu AMP, Andonie R (2019) Semantic fake news detection: a machine learning perspective. In: Rojas I, Joya G, Català A (eds) Advances in computational intelligence—15th international work-conference on artificial neural networks, IWANN 2019, Gran Canaria, Spain, June 12–14, 2019, Proceedings, part I, Springer, lecture notes in computer science, vol 11506, pp 656–667. [https://doi.org/10.1007/978-3-030-20521-8\\_54](https://doi.org/10.1007/978-3-030-20521-8_54)
10. Cambria E, Poria S, Gelbukh AF, Thelwall M (2017) Sentiment analysis is a big suitcase. IEEE Intell Syst 32(6):74–80. <https://doi.org/10.1109/MIS.2017.4531228>
11. Chiu JPC, Nichols E (2016) Named entity recognition with bidirectional LSTM-CNNs. TAACL 4:357–370. <https://transacl.org/ojs/index.php/tacl/article/view/792>
12. Chollet F (2017) Deep learning with python. Manning Publications Co
13. Clark K, Khandelwal U, Levy O, Manning CD (2019) What does BERT look at? An analysis of bert’s attention. CoRR. [arXiv:1906.04341](https://arxiv.org/abs/1906.04341)

14. Daiber J, Jakob M, Hokamp C, Mendes PN (2013) Improving efficiency and accuracy in multilingual entity extraction. In: Sabou M, Blomqvist E, Noia TD, Sack H, Pellegrini T (eds) I-SEMANTICS 2013—9th international conference on semantic systems, ISEM '13, Graz, Austria, September 4–6, 2013, ACM, pp 121–124. <https://doi.org/10.1145/2506182.2506198>
15. Devlin J, Chang M, Lee K, Toutanova K (2019) BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein J, Doran C, Solorio T (eds) Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, vol 1 (long and short papers), Association for Computational Linguistics, pp 4171–4186. <https://doi.org/10.18653/v1/n19-1423>
16. Fentaw HW, Kim TH (2019) Design and investigation of capsule networks for sentence classification. *Appl Sci* 9(11):2200. <https://doi.org/10.3390/app9112200>
17. Fournay A, Rácz MZ, Ranade G, Mobius M, Horvitz E (2017) Geographic and temporal trends in fake news consumption during the 2016 US presidential election. In: [36], pp 2071–2074. <https://doi.org/10.1145/3132847.3133147>
18. Gandon F (2018) A survey of the first 20 years of research on semantic web and linked data. *Ingénierie des Systèmes d'Information* 23(3–4):11–38. <https://doi.org/10.3166/isi.23.3-4.11-38>
19. Gangemi A, Presutti V, Recupero DR, Nuzzolese AG, Draicchio F, Mongiovì M (2017) Semantic web machine reading with FRED. *Semant Web* 8(6):873–893. <https://doi.org/10.3233/SW-160240>
20. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville AC, Bengio Y (2014) Generative adversarial nets. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ (eds) Advances in neural information processing systems 27: annual conference on neural information processing systems 2014, December 8–13 2014, Montreal, Quebec, Canada, pp 2672–2680. <http://papers.nips.cc/paper/5423-generative-adversarial-nets>
21. Gururangan S, Dang T, Card D, Smith NA (2019) Variational pretraining for semi-supervised text classification. In: [34], pp 5880–5894. <https://doi.org/10.18653/v1/p19-1590>
22. Guyon I, von Luxburg U, Bengio S, Wallach HM, Fergus R, Vishwanathan SVN, Garnett R (eds) (2017) Advances in neural information processing systems 30: annual conference on neural information processing systems 2017, 4–9 December 2017, Long Beach, CA, USA
23. Habib A, Asghar MZ, Khan A, Habib A, Khan A (2019) False information detection in online content and its role in decision making: a systematic literature review. *Soc Netw Anal Min* 9(1):50
24. Hastie T, Tibshirani R, Friedman JH (2009) The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer, Berlin. <http://www.worldcat.org/oclc/300478243>
25. Irie K, Tüske Z, Alkhoulí T, Schlüter R, Ney H (2016) LSTM, GRU, highway and a bit of attention: an empirical overview for language modeling in speech recognition. In: Morgan N (ed) Interspeech 2016, 17th annual conference of the international speech communication association, San Francisco, CA, USA, September 8–12, 2016, ISCA, pp 3519–3523. <https://doi.org/10.21437/Interspeech.2016-491>
26. Ji H, Nothman J (2016) Overview of TAC-KBP2016 tri-lingual EDL and its impact on end-to-end KBP. In: Eighth text analysis conference (TAC), NIST. [https://tac.nist.gov/publications/2016/additional\\_papers/](https://tac.nist.gov/publications/2016/additional_papers/)
27. Jin Z, Cao J, Zhang Y, Luo J (2016) News verification by exploiting conflicting social viewpoints in microblogs. In: Schuurmans D, Wellman MP (eds) Proceedings of the thirtieth AAAI conference on artificial intelligence, February 12–17, 2016, Phoenix, Arizona, USA, AAAI Press, pp 2972–2978. <http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12128>
28. Karimi H, Roy P, Saba-Sadiya S, Tang J (2018) Multi-source multi-class fake news detection. In: [7], pp 1546–1557. <https://aclanthology.info/papers/C18-1131/c18-1131>
29. Kiesel J, Mestre M, Shukla R, Vincent E, Adineh P, Corney D, Stein B, Potthast M (2019) Semeval-2019 task 4: hyperpartisan news detection. In: May J, Shutova E, Herbelot A, Zhu X, Apidianaki M, Mohammad SM (eds) Proceedings of the 13th international workshop on semantic evaluation, SemEval@NAACL-HLT 2019, Minneapolis, MN, USA, June 6–7, 2019, Association for Computational Linguistics, pp 829–839. <https://www.aclweb.org/anthology/S19-2145/>
30. Kim J, Jang S, Park EL, Choi S (2020) Text classification using capsules. *Neurocomputing* 376:214–221. <https://doi.org/10.1016/j.neucom.2019.10.033>
31. Kim Y (2014) Convolutional neural networks for sentence classification. In: Moschitti A, Pang B, Daelemans W (eds) Proceedings of the 2014 conference on empirical methods in natural language processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, ACL, pp 1746–1751. <https://www.aclweb.org/anthology/D14-1181/>
32. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. *CoRR*. [arXiv:1412.6980](https://arxiv.org/abs/1412.6980)
33. Kiperavasser E, Goldberg Y (2016) Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL* 4:313–327. <https://transacl.org/ojs/index.php/tacl/article/view/885>



34. Korhonen A, Traum DR, Màrquez L (eds) (2019) Proceedings of the 57th conference of the association for computational linguistics, ACL 2019, Florence, Italy, July 28–August 2, 2019, vol 1, Long Papers, Association for Computational Linguistics. <https://www.aclweb.org/anthology/volumes/P19-1/>
35. Lehmann J, Isele R, Jakob M, Jentzsch A, Kontokostas D, Mendes PN, Hellmann S, Morsey M, van Kleef P, Auer S, Bizer C (2015) DBpedia—a large-scale, multilingual knowledge base extracted from wikipedia. *Semant Web* 6(2):167–195. <https://doi.org/10.3233/SW-140134>
36. Lim E, Winslett M, Sanderson M, Fu AW, Sun J, Culpepper JS, Lo E, Ho JC, Donato D, Agrawal R, Zheng Y, Castillo C, Sun A, Tseng VS, Li C (eds) (2017) Proceedings of the 2017 ACM on conference on information and knowledge management, CIKM 2017, Singapore, November 06–10, 2017, ACM. <http://dl.acm.org/citation.cfm?id=3132847>
37. Liu C, Wu X, Yu M, Li G, Jiang J, Huang W, Lu X (2019) A two-stage model based on bert for short fake news detection. In: International conference on knowledge science, Springer, Engineering and Management, pp 172–183
38. Liu Y, Wu YB (2018) Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: McIlraith SA, Weinberger KQ (eds) Proceedings of the thirty-second AAAI conference on artificial intelligence, New Orleans, Louisiana, USA, February 2–7, 2018, AAAI Press. <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16826>
39. Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019) Roberta: a robustly optimized BERT pretraining approach. *CoRR*. [arXiv:1907.11692](https://arxiv.org/abs/1907.11692)
40. Long Y, Lu Q, Xiang R, Li M, Huang C (2017) Fake news detection through multi-perspective speaker profiles. In: Kondrak G, Watanabe T (eds) Proceedings of the eighth international joint conference on natural language processing, IJCNLP 2017, Taipei, Taiwan, November 27–December 1, 2017, vol 2: short papers, Asian Federation of Natural Language Processing, pp 252–256. <https://aclanthology.info/papers/I17-2043/i17-2043>
41. Lundberg SM, Lee S (2017) A unified approach to interpreting model predictions. In: [22], pp 4765–4774. <http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions>
42. Mikolov T, Grave E, Bojanowski P, Puhresch C, Joulin A (2018) Advances in Pre-Training Distributed Word Representations. In: Calzolari N, Choukri K, Cieri C, Declerck T, Goggi S, Hasida K, Isahara H, Maegaard B, Mariani J, Mazo H, Moreno A, Odijk J, Piperidis S, Tokunaga T (eds) Proceedings of the eleventh international conference on language resources and evaluation, LREC 2018, Miyazaki, Japan, May 7–12, 2018., European Language Resources Association (ELRA). <http://www.lrec-conf.org/lrec2018>
43. Nickel M, Murphy K, Tresp V, Gabrilovich E (2016) A review of relational machine learning for knowledge graphs. *Proc IEEE* 104(1):11–33. <https://doi.org/10.1109/JPROC.2015.2483592>
44. Parikh SB, Atrey PK (2018) Media-rich fake news detection: a survey. In: IEEE 1st conference on multimedia information processing and retrieval, MIPR 2018, Miami, FL, USA, April 10–12, 2018, IEEE, pp 436–441. <http://doi.ieeecomputersociety.org/10.1109/MIPR.2018.00093>
45. Qi Y, Sachan DS, Felix M, Padmanabhan S, Neubig G (2018) When and why are pre-trained word embeddings useful for neural machine translation? In: Walker MA, Ji H, Stent A (eds) Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1–6, 2018, vol 2 (Short Papers), Association for Computational Linguistics, pp 529–535. <https://aclanthology.info/papers/N18-2084/n18-2084>
46. Rashkin H, Choi E, Jang JY, Volkova S, Choi Y (2017) Truth of varying shades: analyzing language in fake news and political fact-checking. In: Palmer M, Hwa R, Riedel S (eds) Proceedings of the 2017 conference on empirical methods in natural language processing, EMNLP 2017, Copenhagen, Denmark, September 9–11, 2017, Association for Computational Linguistics, pp 2931–2937. <https://aclanthology.info/papers/D17-1317/d17-1317>
47. Ribeiro MT, Singh S, Guestrin C (2016) “why should I trust you?”: explaining the predictions of any classifier. In: Krishnapuram B, Shah M, Smola AJ, Aggarwal CC, Shen D, Rastogi R (eds) Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, San Francisco, CA, USA, August 13–17, 2016, ACM, pp 1135–1144. <https://doi.org/10.1145/2939672.2939778>
48. Rubin V, Conroy N, Chen Y, Cornwell S (2016) Fake news or truth? Using satirical cues to detect potentially misleading news. In: Proceedings of the second workshop on computational approaches to deception detection, pp 7–17
49. Rubin VL, Chen Y, Conroy NJ (2015) Deception detection for news: three types of fakes. In: Information science with impact: research in and for the community—proceedings of the 78th ASISand T annual meeting, ASIST 2015, St. Louis, Missouri, Missouri, USA, October 6–10, 2015, Wiley, Proceedings of the association for information science and technology, vol 52, no 1, pp 1–4. <https://doi.org/10.1002/pr2.2015.145052010083>

50. Ruchansky N, Seo S, Liu Y (2017) CSI: a hybrid deep model for fake news detection. In: [36], pp 797–806
51. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. In: [22], pp 3859–3869. <http://papers.nips.cc/paper/6975-dynamic-routing-between-capsules>
52. Schlichtkrull MS, Kipf TN, Bloem P, van den Berg R, Titov I, Welling M (2018) Modeling relational data with graph convolutional networks. In: Gangemi A, Navigli R, Vidal M, Hitzler P, Troncy R, Hollink L, Tordai A, Alam M (eds) The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings, Springer, lecture notes in computer science, vol 10843, pp 593–607. [https://doi.org/10.1007/978-3-319-93417-4\\_38](https://doi.org/10.1007/978-3-319-93417-4_38)
53. Shu K, Sliva A, Wang S, Tang J, Liu H (2017) Fake news detection on social media: a data mining perspective. SIGKDD Explor 19(1):22–36. <https://doi.org/10.1145/3137597.3137600>
54. Shu K, Wang S, Liu H (2017) Exploiting tri-relationship for fake news detection. CoRR. [arXiv:1712.07709](https://arxiv.org/abs/1712.07709)
55. Singhanian S, Fernandez N, Rao S (2017) 3HAN: a deep neural network for fake news detection. In: Liu D, Xie S, Li Y, Zhao D, El-Alfy EM (eds) Neural information processing: 24th international conference, ICONIP 2017, Guangzhou, China, November 14–18, 2017, Proceedings, part II, Springer, lecture notes in computer science, vol 10635, pp 572–581. [https://doi.org/10.1007/978-3-319-70096-0\\_59](https://doi.org/10.1007/978-3-319-70096-0_59)
56. Solaiman I, Brundage M, Clark J, Askell A, Herbert-Voss A, Wu J, Radford A, Wang J (2019) Release strategies and the social impacts of language models. CoRR. [arXiv:1908.09203](https://arxiv.org/abs/1908.09203)
57. Strubell E, Ganesh A, McCallum A (2019) Energy and policy considerations for deep learning in NLP. In: [34], pp 3645–3650. <https://doi.org/10.18653/v1/p19-1355>
58. Thorne J, Vlachos A (2018) Automated fact checking: Task formulations, methods and future directions. In: [7], pp 3346–3359. <https://www.aclweb.org/anthology/C18-1283/>
59. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: [22], pp 6000–6010. <http://papers.nips.cc/paper/7181-attention-is-all-you-need>
60. Vo N, Lee K (2018) The rise of guardians: fact-checking URL recommendation to combat fake news. In: Collins-Thompson K, Mei Q, Davison BD, Liu Y, Yilmaz E (eds) The 41st international ACM SIGIR conference on research and development in information retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08–12, 2018, ACM, pp 275–284. <https://doi.org/10.1145/3209978.3210037>
61. Vosoughi S, Roy D, Aral S (2018) The spread of true and false news online. Science 359(6380):1146–1151
62. Wang WY (2017) “Liar, liar pants on fire”: A new benchmark dataset for fake news detection. CoRR. [arXiv:1705.00648](https://arxiv.org/abs/1705.00648)
63. Wu L, Liu H (2018) Tracing fake-news footprints: characterizing social media messages by how they propagate. In: Chang Y, Zhai C, Liu Y, Maarek Y (eds) Proceedings of the eleventh ACM international conference on web search and data mining, WSDM 2018, Marina Del Rey, CA, USA, February 5–9, 2018, ACM, pp 637–645. <https://doi.org/10.1145/3159652.3159677>
64. Yang K, Niven T, Kao H (2019) Fake news detection as natural language inference. CoRR. [arXiv:1907.07347](https://arxiv.org/abs/1907.07347)
65. Young T, Hazarika D, Poria S, Cambria E (2018) Recent trends in deep learning based natural language processing [review article]. IEEE Comp Int Mag 13(3):55–75. <https://doi.org/10.1109/MCI.2018.2840738>
66. Zannettou S, Sirivianos M, Blackburn J, Kourtellis N (2018) The web of false information: rumors, fake news, Hoaxes, Clickbait, and various other shenanigans. CoRR. [arXiv:1804.03461](https://arxiv.org/abs/1804.03461)
67. Zellers R, Holtzman A, Rashkin H, Bisk Y, Farhadi A, Roesner F, Choi Y (2019) Defending against neural fake news. In: Wallach HM, Larochelle H, Beygelzimer A, d’Alché-Buc F, Fox EB, Garnett R (eds) Advances in neural information processing systems 32: annual conference on neural information processing systems 2019, NeurIPS 2019, 8–14 December 2019, Vancouver, BC, Canada, pp 9051–9062. <http://papers.nips.cc/paper/9106-defending-against-neural-fake-news>