Learning Hierarchical Discourse-level Structure for Fake News Detection

Hamid Karimi
Computer Science and Engineering
Michigan State University
karimiha@msu.edu

Jiliang Tang
Computer Science and Engineering
Michigan State University
tangjili@msu.edu

Abstract

On the one hand, nowadays, fake news articles are easily propagated through various online media platforms and have become a grand threat to the trustworthiness of information. On the other hand, our understanding of the language of fake news is still minimal. Incorporating hierarchical discourse-level structure of fake and real news articles is one crucial step toward a better understanding of how these articles are structured. Nevertheless, this has rarely been investigated in the fake news detection domain and faces tremendous challenges. First, existing methods for capturing discourse-level structure rely on annotated corpora which are not available for fake news datasets. Second, how to extract out useful information from such discovered structures is another challenge. To address these challenges, we propose **H**ierarchical **D**iscourse-level **S**tructure for **F**ake news detection. HDSF learns and constructs a discourse-level structure for fake/real news articles in an automated and data-driven manner. Moreover, we identify insightful structure-related properties, which can explain the discovered structures and boost our understating of fake news. Conducted experiments show the effectiveness of the proposed approach. Further structural analysis suggests that real and fake news present substantial differences in the hierarchical discourse-level structures.

1 Introduction

In this work, we focus on detecting fake news articles (hereafter referred to as *documents*) based on their contents. Many existing linguistic approaches for fake news detection (Feng et al., 2012; Pennebaker et al., 2015; Ott et al., 2011) overlook a crucial linguistic aspect of fake/real news documents i.e., the hierarchical discourse-level structure. Usually, in a document, discourse units (e.g., sentences) are organized in a hierarchical structure e.g., a tree. The importance of considering the hierarchical discourse-level structure for fake news detection is three-fold. First, previous studies (Bachenko et al., 2008; Rubin and Lukoianova, 2015) explored discourse-level structure in fake news detection and discovered that the way two discourse units of a document are connected could be quite revealing and insightful about its truthfulness. For instance, (Rubin and Lukoianova, 2015) applied Rhetorical Structure Theory (RST) (Mann and Thompson, 1988) and noted that fake stories lack “evidence” as a defined inter-discourse relation. Second, fake news is typically produced by connecting disjoint pieces of news and unlike well-established journalism (e.g., New York Times) fake news production lacks a meticulous editorial board. Therefore, by incorporating the hierarchical discourse-level structure, we can investigate the coherence of fake/real news documents (we will show this later). Third, a substantial number of studies have shown that using hierarchical structures yields a better document representation in various downstream tasks whose predictions depend on the entire text (Bhatia et al., 2015; Morey et al., 2018; Li et al., 2014b). Since typically fake news detection is considered as a classification problem based on the entire text, applying discourse analysis has the potential to advance fake news detection (this will be verified later).

On the other hand, incorporating the hierarchical structure at the discourse level for fake news detection faces tremendous challenges. First, many existing methods incorporating structural discourse (Li et al., 2014a; Bhatia et al., 2015) (not for fake news detection though) rely on annotated corpora such as Penn Discourse Treebank (Prasad et al., 2007). Constructing and annotating such corpora is an arduous and costly process. Incor-

Proceedings of NAACL-HLT 2019, pages 3432–3442

porating hierarchical structure is even more difficult for fake news detection as there exists virtually no available annotated discourse corpus in this domain. Therefore, we need to learn the discourse-level structure in a data-driven and automated manner. Second, how to use the hierarchical discourse-level structure and extract insightful information that can boost our understanding of fake news is another challenge.

In this study, we embrace the opportunities and challenges and propose **H**ierarchical **D**iscourse-level **S**tructure for **F**ake news detection (HDSF) framework. HDSF in an automated manner learns a hierarchical structure for a given document through an approach based on the dependency parsing at the sentence level (i.e., sentences are discourse units). As an example, Figure 1 illustrates a hierarchical structure of a document i.e., a dependency tree of sentences. Sentences represent discourse units and two sentences are connected by a directed link where a sentence at the *head* of the link *semantically depends* on a sentence at the *tail* of the link e.g., in Figure 1, the sentence s_4 depends on the sentence s_2.

Our key contributions are summarized as follow.

- To the best of our knowledge, we are the first to study automatic document structure learning for fake news detection.
- We propose the framework HDSF that automatically and in an end-to-end manner learns structurally rich representations for fake and real news documents.
- We identify a set of structure-related properties delineating meaningful structural differences between fake and real news documents.

The rest of the paper is organized as follows. In Section 2, we formally define the problem. Section 3 describes the proposed framework. In Section 4, we introduce several structure-related properties to analyze the hierarchical structures. Section 5 presents the experiments and discussions followed by an overview of the related work in Section 6. We conclude the paper in Section 7 and shed light on a few future directions.

2 Notations and Problem Statement

Following the previous work (Allcott and Gentzkow, 2017; Shu et al., 2017), we define the fake news as follows.

A sample document from our corpus

S_1. Massive protests have broken out in Charlotte, North Carolina, after police shot and killed disabled an African-American man named Keith Lamont Scott, who was apparently reading a book in his car, waiting for his son to come home from school when officers shot him dead.
S_2. Details are still sketchy at the moment, but it appears the police were in search of another armed man in the area.
S_3. Scott is the sixth civilian shot by Charlotte police this year.
S_4. Twelve officers have apparently been injured as tear gas canisters were fired into the crowd.
S_5. Scott is the 215th black man shot by police this year.

A hierarchical discourse-level dependency tree

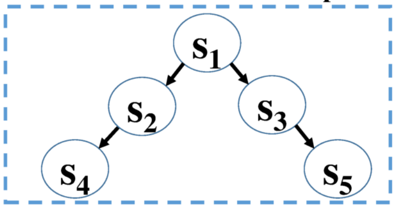


Figure 1: An illustration of the hierarchical discourse-level structure of a document using a dependency tree

Definition. We define a news document *fake* if its content is verified to be false and *real* otherwise.

Let's briefly introduce some notations. Suppose we have a corpus $\mathcal{D}$ of fake and real news documents. Let a document $d \in \mathcal{D}$ contain k sentences $s_1, s_2, \cdots s_k$. Suppose a sentence $s_j \in d$ $(1 \leq j \leq k)$ includes words $W_j = \{w_1, w_2, \cdots, w_{T_j}\}$ where T_j denotes the number of words in sentence s_j. Additionally, binary labels Y (i.e., *fake* or *real* labels) hold ground-truth labels associated with documents in $\mathcal{D}$.

Given the corpus of fake/real news documents (i.e., $\mathcal{D}$), we aim to learn model $\mathcal{M}$ that can automatically learn hierarchical and structurally rich representations for documents in $\mathcal{D}$. Meanwhile, given binary labels Y, the model $\mathcal{M}$ uses the hierarchical representations to automatically predict the labels of unseen news documents.

3 The Proposed Framework

To incorporate the discourse-level structure for fake news detection, we propose the framework HDSF illustrated in Figure 2. It provides three components: the *Hierarchical Discourse-level*

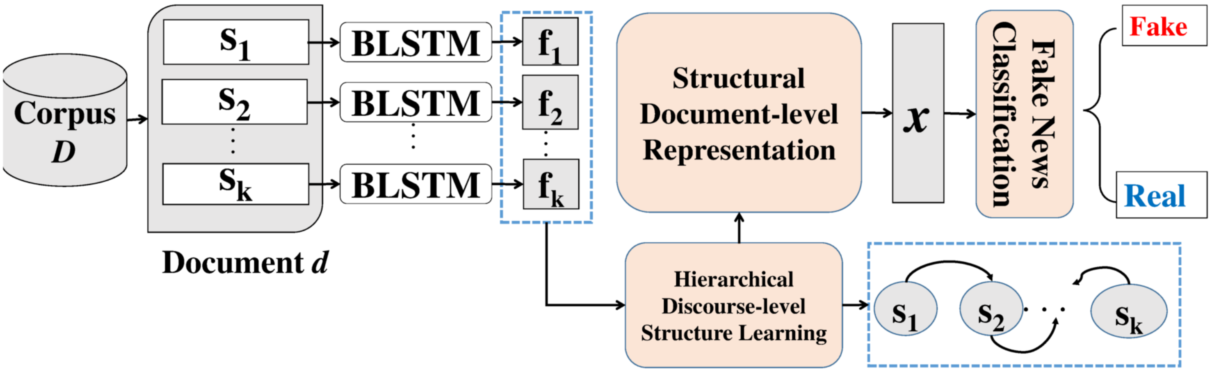


Figure 2: The proposed framework Hierarchical Discourse-level Structure for Fake news detection (HDSF)

Structure Learning automatically learns a proper structure for a given document, the *Structural Document-level Representation* yields a numerical and unified representation for the entire document, which is used by the *Fake News Classification* component to predict the label of a document.

3.1 Hierarchical Discourse-level Structure Learning

In this component, we aim to construct a hierarchical structure between discourse units (i.e., sentences) without relying on an annotated corpus. To achieve this, we use a dependency parsing approach (Liu and Lapata, 2018; Li et al., 2014a; Kim et al., 2017) and represent the hierarchical structure of a document as a dependency tree (see Figure 1 for an example of a dependency tree). In dependency parsing of discourse units, we mainly need to identify if a discourse unit semantically depends on another one. If so, a *parent-child* (or *tail-head*) link in the dependency tree can be established. Therefore, as long as the semantic dependencies between discourse units are identified, we can construct a discourse-level dependency tree without an annotated corpus. In Section 3.1.1, we describe a method to discover the semantic dependencies between discourse units in an automated manner. Afterward, in Section 3.1.2, we utilize these dependencies to construct the dependency tree of a document.

3.1.1 Learning Inter-Discourse Dependencies

Since discourse units are defined as sentences, we first need to get a fixed representation for each sentence. To this end, we utilize Bi-directional Long Short-Term Memory (BLSTM) network (Schuster and Paliwal, 1997). We represent each word in a sentence s_j by a fixed-size word embedding, and further the BLSTM network at each time step $t \in [1, T_j]$ executes the following functions[1]:

$$\begin{aligned} \overrightarrow{h}_t &= \mathcal{F}(\overrightarrow{h}_{t-1}; w_{t-1}) \\ \overleftarrow{h}_t &= \mathcal{F}(\overleftarrow{h}_{t-1}; w_{T_j-t+1}) \end{aligned} \tag{1}$$

where $\mathcal{F}$ is the LSTM function (Hochreiter and Schmidhuber, 1997), and $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ are outputs of the forward and backward LSTM networks at time step t, respectively. Then, a fixed representation for a sentence s_j, denoted as f_j, is defined as the average of the last output of forward and backward LSTM networks:

$$f_j = \frac{[\overrightarrow{h}_{T_j} + \overleftarrow{h}_{T_j}]}{2} \tag{2}$$

Similarly, we apply the BLSTM network to all sentences of a document and obtain a sequence of sentential representations i.e., $f_1, f_2, \cdots, f_k$ (see Figure 2).

As mentioned before, in dependency parsing, we need to identify the dependency between two discourse units in an automated manner. To do this, the HDSF framework learns and optimizes an inter-sentential attention matrix $\mathbf{A} \in \mathbb{R}^{k \times k}$. The entry (m, n) of $\mathbf{A}$ holds the probability of the sentence s_m being the parent of the sentence s_n where $1 \leq m, n \leq k$ and $m \neq n$. In other words, $\mathbf{A}$ contains parent-child probabilities and is computed as follows.

$$\begin{aligned} u_m &= \mathcal{G}(\mathbf{W} \times f_m + \mathbf{b}) \\ u_n &= \mathcal{G}(\mathbf{W} \times f_n + \mathbf{b}) \\ \mathbf{A}[m, n] &= \frac{e^{(u_m \odot u_n)}}{\sum_{i=1}^{k} e^{\sum (u_i \odot u_n)}} \end{aligned} \tag{3}$$

[1] Note that the backward LSTM (i.e., the lower part of Eq. 1) takes a sequence of words in a reverse order.

where $\mathcal{G}$ is a non-linear activation function, $\mathbf{W}$ is some weight matrix, $\mathbf{b}$ is a bias vector, and $\odot$ denotes the dot product operator. Further, since we need a root node in a dependency tree, we compute the probability of a sentence s_j being the root node, denoted as r_j, as follows.

$$
\begin{aligned}
u_j &= \mathcal{G}(\mathbf{W} \times f_j + \mathbf{b}) \\
r_j &= \frac{e^{\sum_{\forall y} u_j[y]}}{\sum_{i=1}^{k} e^{\sum_{\forall y} u_i[y]}}
\end{aligned}
\tag{4}
$$

where $u_j[y]$ is the y-th element of vector u_j. Similarly, we calculate the root probabilities for all sentences and obtain the array of root probabilities denoted as $\mathbf{r} = \{r_1, r_2, \cdots, r_k\}$ where $0 \leq r_j \in \mathbf{r} \leq 1$.

3.1.2 Discourse Dependency Tree Construction

We use the learned matrix of inter-sentential parent-child probabilities i.e., $\mathbf{A}$ (Eq. 3) as well as the array of root probabilities i.e., $\mathbf{r}$ (Eq. 4) and propose a greedy algorithm, illustrated in Algorithm 1, to construct the discourse dependency tree of a document. A sentence with the maximum value in $\mathbf{r}$ is considered the root node and is inserted into the tree (line 5). Then, at each iteration, the algorithm finds the maximum entry in a block of the matrix $\mathbf{A}$ whose rows correspond to the rows of current nodes added to the tree (i.e., nodes V in line 7) and its columns correspond to columns of the rest of nodes (i.e., nodes $N \backslash V$ in line 7). Note that the columns of the current nodes are excluded because their parents have already been identified and also each node should have exactly one parent (except the root which has no parent). Assume the search in line 7 results in the entry (p, c) of $\mathbf{A}$ where $1 \leq p, c \leq k$ and $p \neq c$. Then, the sentence s_c is added as the child node of the sentence s_p (line 8). Algorithm 1 continues until all sentences of a document are added to the tree $\mathcal{T}$.

To fix the idea of discourse dependency tree construction algorithm, we present a step-by-step execution of this algorithm demonstrated in Figure 3. In Step 0, the sentence s_1 is added as the root of the tree $\mathcal{T}$ since it has the maximum value in the array of the root probabilities. Next in Step 1, the algorithm searches for the maximum probability value in the row s_1 while the column s_1 is excluded. The maximum value is this block is 0.4 and corresponds to entry (s_1,s_2). Therefore, the sentence s_2 is added as the child node of the sentence s_1. In a similar fashion, the algorithm continues until all 6 sentences are added to the tree.

Algorithm 1: The proposed algorithm for discourse dependency tree construction

Input: $\mathbf{A}$; $\mathbf{r}$
Output: Discourse dependency tree $\mathcal{T}$

1 $\mathcal{T} = empty$
2 $N = \{s_1, s_2, \cdots, s_k\}$ `// All nodes`
3 $V = \{\}$ `// Set of current nodes`
4 Add $N[argmax(\mathbf{r})]$ to V
5 $\mathcal{T}.root = N[argmax(\mathbf{r})]$ `// Adding the root`
6 **while** $|V| \neq k$ **do**
7     $p, c = argmax(\mathbf{A}[V, N \backslash V])$ `// Search block`
8     $\mathcal{T}.link(N[c], N[p])$ `// Child-parent link`
9     $V.add(N[c])$ `// Adding the child node`
10 **end**
11 **return** $\mathcal{T}$

3.2 Structural Document-level Representation

We use a similar method presented in (Liu and Lapata, 2018) to extract a structurally rich representation for the entire document. First, for each sentence (i.e., a discourse unit), we obtain a structurally-aware representation. To achieve this, we take into account the parent-child probabilities as well as the root probabilities as follows.

$$
\begin{aligned}
\mathbf{p}_j &= r_j \times \mathbf{e}_{root} + \sum_{z=1}^{k} \mathbf{A}[z, j] \times f_z \\
\mathbf{c}_j &= \sum_{z=1}^{k} \mathbf{A}[j, z] \times f_j \\
\mathbf{g}_j &= \mathcal{G}(\mathbf{W}[\mathbf{p}_j || \mathbf{c}_j || f_j] + \mathbf{b})
\end{aligned}
\tag{5}
$$

where $\mathbf{p}_j$ and $\mathbf{c}_j$ are two context vectors taking into account possible parents and children of a sentence s_j, respectively, $\mathbf{e}_{root}$ denotes a special root embedding vector, $||$ denotes vector concatenation operator, and $\mathbf{g}_j$ is a *structurally-aware representation for sentence* s_j. Finally, to extract a structurally rich representation for the entire doc-

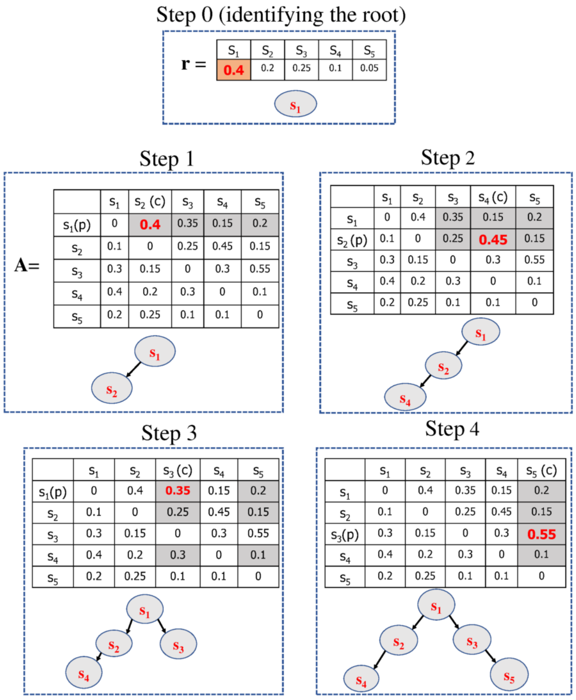


Figure 3: An illustration of a step-by-step execution of Algorithm 1 for the sample document presented in Figure 1. The search block at each iteration has been highlighted. Note the letter 'p' for the *parent node* and 'c' for the *child node*.

ument, we average all of its $\mathbf{g}_j$ vectors:

$$x = \frac{1}{k}\sum_{j=1}^{k} \mathbf{g}_j \tag{6}$$

where x denotes *the structurally rich document-level representation*.

3.3 Fake News Classification

We hypothesize that the document-level structurally rich representations (Eq. 6) offer a discriminatory power to detect fake news documents. Therefore, as shown in Figure 2, we employ a binary classification for fake news detection formulated as follows.

$$\begin{aligned} c_i &= -\big(y_i log(p_f^i) + (1 - y_i) log(p_r^i)\big) \\ \mathcal{L}(\theta) &= \sum_{\forall d_i \in \mathcal{D}} c_i \end{aligned} \tag{7}$$

where y_i is the ground-truth label of a document $d_i \in \mathcal{D}$, c_i is the cross-entropy loss value, p_f^i and p_r^i are probabilities of d_i being fake or real, respectively. p_f^i and p_r^i are obtained from the representation x (Eq. 6) fed to a fully connected layer followed by the softamx function. In Eq. 7, θ denotes the framework's parameters and $\mathcal{L}$ is the total loss of the model. Since the entire framework (Eq. 1 to 7) is fully differentiable, we utilize the error backpropagation (Hecht-Nielsen, 1992) to calculate the gradients followed by the stochastic gradient descent (Bottou, 2010) to update and optimize the parameters. Note that the discourse dependency tree construction algorithm presented in Section 3.1.2 is employed post hoc. Hence, the gradient calculation over $argmax$ operator (line 7 of Algorithm 1) is not performed.

4 Structure-related Properties

Incorporating discourse-level structure offer a discriminatory power distinguishing fake and real documents, which will be verified in Section 5.3. We expect more than this discriminatory power. We expect to identify insightful and interpretable information from extracted structures which can delineate intrinsic differences between fake and real news documents. To meet this expectation, we define three fundamental properties of constructed discourse dependency trees. Note that we leave the definitions of more advanced properties as one future work. We seek to fulfill two goals by defining the structure-related properties. First, we intend to highlight the way that fake and real news documents are different. Second, we intend to leverage these properties to shed light on the *coherence* of fake and real news documents. Coherence has been the subject of many studies (Barzilay and Lapata, 2008; Lin et al., 2011; Guinaudeau and Strube, 2013; Li and Hovy, 2014) and is concerned with how constituents of a document (e.g., discourse units) are linked together in a way that the entire text creates a clear mental picture to the readers (Storrer, 2002). Notwithstanding its importance, coherence has not been investigated in the fake news domain in a large-scale and systematic fashion. Aiming at filling this gap, we naturally connect the defined structure-related properties with the coherence of news documents.

Property 1 (Number of Leaf Nodes) *This property, denoted as* P_l*, defines the normalized number of leaf nodes in a discourse dependency tree:*

$$\mathsf{P}_l = \frac{l}{log(k)} \tag{8}$$

where l is the number of leaf nodes (i.e., sentences) in the discourse dependency tree of a document. Recall that k is the total number of sentences in a document.

The intuition behind defining Property 1 is as follows. According to the description of the dependency tree in Section 3.1, leaf nodes are isolated discourse units and no other discourse units depend on them. Thus, the more the number of leaf nodes is, the less inter-linked the discourse units will be, and vice versa. Therefore, Property 1 is likely to indicate the coherence of a document – the higher P_l, the more isolated sentences and the less coherent the document. Also, for a document with k sentences $\mathsf{P}_l \in [\frac{1}{log(k)}, \frac{k-1}{log(k)}]$.

Property 2 (Preorder Difference) *This property, denoted as P_t, defines the normalized positional difference between the preorder traversal of a document's discourse dependency tree and its original sentential sequential order:*

$$\mathsf{P}_t = \frac{\sum_{j=1}^{k} |s_j^{position} - j|}{log(k)} \tag{9}$$

where $s_j^{position}$ denotes the position of a sentence s_j in the preorder traversal[1] of dependency tree of a document e.g., $s_3^{position} = 4$ in Figure 1. The position of s_j in the original sequential order is simply j[2]. The preorder traversal of the tree in Figure 1 is the sequence $\{s_1, s_2, s_4, s_3, s_5\}$ and the sentential sequential order is $\{s_1, s_2, s_3, s_4, s_5\}$. Therefore, according to the definition of Property 2:
$\mathsf{P}_t = \frac{|1-1|+|2-2|+|4-3|+|3-4|+|5-5|}{log(5)} \approx 2.86$.

The preorder traversal of a document's discourse dependency tree takes into consideration the organization of a document respect to the dependencies between sentences. Then, the purpose of Property 2 is to measure how much the organization of a document, captured through the preorder traversal, deviates from its sentential sequential order. Sentence order is highly related to the coherence of a document where the displaced order of sentences in a document makes it less coherent (Li and Hovy, 2014). Thus, intuitively, the less the value of Property 2 for a document is, the more coherent that document should be. Also, for a document with k sentences $\mathsf{P}_t \in [\frac{k-1}{log(k)}, \frac{(k^2-1)}{2log(k)}]$ if k is odd and $\mathsf{P}_t \in [\frac{k-1}{log(k)}, \frac{k^2}{2log(k)}]$ if k is even.

[1]The subtrees are ordered based on *when* they are added as the child nodes of a parent node in Algorithm 1. That is why we can compute preorder traversal.

[2]We assume that the sentences in the sequential order are numbered incrementally from 1 to k.

Property 3 (Parent-Child Distance) *This property, denoted as P_c, defines the normalized sum of positional distances between child nodes and their parents when they are considered in the original sequential order:*

$$\mathsf{P}_c = \frac{\sum_{\forall c,p \in \mathcal{T}} |c_{position} - p_{position}|}{log(k)} \tag{10}$$

where $c_{position}$ and $p_{position}$ denote the positions of a child node c and a parent node p, respectively, in the original sentential sequential order. For instance, in our running example, the parent node s_3 has $p_{position} = 3$ (i.e., it is the third sentence) and its child node s_5 has $c_{position} = 5$. Therefore, their parent-child distance is $|5 - 3| = 2$. Following a similar calculation for other parent-child pairs, we have $\mathsf{P}_c = \frac{1+2+2+2}{log(5)} \approx 10$.

Similar to Property 2, Property 3 pertains to the organization of a document and takes into consideration the deviation from sentential sequential order. Intuitively speaking, usually, we expect that a child node and its parent to be close to each other in the original sequential order. Consequently, the less value of this property is, the more coherent a document is likely to be. The range of Property 3 in a document containing k sentences is $\mathsf{P}_c \in [\frac{k-1}{log(k)}, \frac{k(k-1)}{2log(k)}]$.

5 Experiments

To verify the performance of the proposed framework HDSF, we conduct a set of experiments. We seek to answer the following research questions:

1. How does the proposed framework perform on fake news detection?

2. How do the defined structure-related properties describe the fake and real news documents?

In this section, we first describe the datasets followed by presenting the experimental settings. Afterward, we evaluate the performance of HDSF compared to several representative baselines. Finally, we present a structural analysis of the fake/real news documents.

5.1 Datasets

We utilize five available fake news datasets in this study. The first two datasets are collected by (Shu et al., 2017) and include online articles whose veracities have been identified by experts in BuzzFeed[1] and PolitiFact[2]. For the next two datasets, we utilize two available online fake news datasets provided by *kaggle.com*[3] [4]. Finally, we include the dataset constructed and shared by McIntire[5]. Since the proposed framework HDSF is a general-purpose framework investigating discourse-level structures of fake/real news documents based on their textual contents, we do not restrict HDSF to a particular source of data and therefore combine all datasets. Similar to the previous work (Shu et al., 2017), we balance the dataset to avoid a trivial solution as well as ensuring a fair performance comparison. In total, we have 3360 fake and 3360 real documents.

5.2 Experimental Settings

First, we pre-process the documents by removing numbers, non-English characters, stop-words (e.g., 'with'), and converting all characters to lower case. We randomly select 134 documents as the development set, (67 from each class) and 134 documents (67 from each class) as the test set. The remaining 6452 documents are used for training. The development set is used for tuning the hyper-parameters. We initialize the word embeddings from the Google news pre-trained word2vec embeddings (Mikolov et al., 2013)[6]. LeakyReLU (Xu et al., 2015) is used as the non-linear activation function and the number of hidden units in the BLSTM network is set to 100. Each simulation is run for 200 steps with a random mini-batch size of 40 documents. The learning rate starts at 0.01 with the decay rate of 0.9 after every 50 steps. We use the ADAM optimizer (Kingma and Ba, 2014) to optimize the parameters. The PyTorch package[7] is utilized for the implementation and the code and data are publicly available in `https://github.com/hamidkarimi/HDSF`.

[1] `https://www.buzzfeed.com`

[2] `http://www.politifact.com/`

[3] `https://www.kaggle.com/mrisdal/fake-news/data`

[4] `https://www.kaggle.com/jruvika/fake-news-detection`

[5] `https://github.com/GeorgeMcIntire/fake_real_news_dataset`

[6] Out-of-vocabulary words are initialized randomly.

[7] `https://pytorch.org/`

5.3 Comparison Results

To answer the research question (1), we compare the performance of HDSF with the following representative baselines.

N-grams. In this baseline method, we extract and combine unigrams, bigrams, and trigrams features and use SVM (Support Vector Machines) (Scholkopf and Smola, 2001) for classification.

LIWC (Pennebaker et al., 2015). LIWC (Linguistic Inquiry and Word Count) offers a set of rich psycholinguistic features for a written document. We extract 94 features for each document and use SVM for classification.

RST (Rubin and Lukoianova, 2015). We extract a set of RST relations (Mann and Thompson, 1988) using the implementation of the method proposed by (Ji and Eisenstein, 2014). Then, we vectorize the relations and employ SVM for classification. This baseline takes into account the hierarchical structure of documents via RST.

BiGRNN-CNN (Ren and Zhang, 2016). A CNN (Convolutional Neural Network) is applied at the sentence-level on word embeddings and a BiGRNN (Bi-Directional Gated Recurrent Neural Network) extracts features from a sequence of extracted sentential features. This baseline takes into consideration a two-level sequential structure for a document.

LSTM[w+s]. In this baseline, we apply an LSTM network on a sequence of word embeddings belonging to a sentence and then apply another LSTM on a sequence of extracted sentential features. LSTM[w+s] also considers a two-level sequential structure for a document.

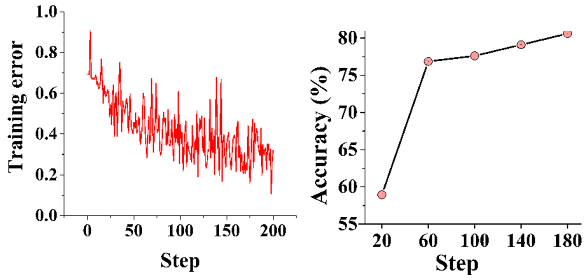
LSTM[s]. This method is similar to LSTM[w+s] except that the mean of word embeddings in a sentence is used instead of applying an LSTM network. LSTM[s] considers a single sequential structure for a document.

We use accuracy as the metric of performance evaluation given that the dataset is fully balanced. Table 1 shows the comparison results on the test set and we make the following observations:

- N-grams achieve a better performance than LIWC. In line with the previous study (Ott et al., 2011), this shows that for fake news detection, taking into account the context of a document as n-grams do is more effective than employing the existing pre-defined dictionaries as LIWC does.

Method	Accuracy (%)
N-grams	72.37
LIWC	70.26
RST	67.68
BiGRNN-CNN	77.06
LSTM[w+s]	80.54
LSTM[s]	73.63
HDSF	**82.19**

Table 1: Comparison results



(a) The training error (b) The accuracy on dev set

Figure 4: The Inspection of HDSF

- Most of the time, methods wherein a document's structure is somehow taken into account outperform n-grams and LIWC. This observation shows that for fake news detection, the content's structure plays an important role.

- The poor performance of RST is because of the following reasons: a) using RST without an annotated corpus is not very effective, and b) RST relations are extracted using auxiliary tools optimized for other corpora which cannot be applied effectively to the fake news corpus in hand. Note that annotating RST for our corpus is extremely unscalable and time-consuming.

- The proposed framework HDSF significantly outperforms all other methods. This observation shows that hierarchical discourse-level representations are effectively rich for fake news prediction.

5.4 The Inspection of HDSF

To further verify the working of the HDSF framework, we inspect HDSF in more detail. Figure 4a demonstrates the training error during model optimization. As we can observe from this figure, the error is decreasing as the training process proceeds. Furthermore, Figure 4b demonstrates the accuracy on the development set during the training and it is monotonically increasing as the training goes on. Hence, based on these figures, we can ensure the framework is getting optimized and *learns* to classify fake news documents correctly.

5.5 Structural Analysis for Fake/Real Documents

In this section, we compute the average values of structure-related properties, presented in Section 4, for the fake/real news documents belonging to the test set. Figure 5 shows the results. We make the two key observations based on this figure:

- There is a significant difference in all three properties for fake news documents vs. real news documents. *This observation shows the fact that structures of fake news documents at the discourse-level are substantially different from those of real ones.*

- Noticeably, in all three properties, the real news documents show less value than the fake news documents. As described in Section 4, all three properties are closely connected to the coherence of a document. *Therefore, real news documents indicate more degree of coherence.*

6 Related Work

Content-based fake news detection has been the subject of many linguistic research endeavors. (DePaulo et al., 2003) investigated fake stories introduced insightful cues in fake stories and highlighted 'unusual' language in such stories. N-grams and Part-of-Speech (POS) tags are fundamental features of a text which have been utilized for fake news detection (Ahmed et al., 2017; Ott et al., 2013). Also, LIWC (Pennebaker et al., 2015) has been employed to investigate the role of individual words in a document for deception detection (Ott et al., 2011). Since POS tags, n-grams, and LIWC features are considered as 'shallow' and hand-crafted features, deep neural networks have been utilized for fake news detection where features are extracted automatically (Wang, 2017; Ren and Zhang, 2016; Volkova et al., 2017; Karimi et al., 2018). In this study, we also utilized an automated feature extraction instead of relying on hand-craft features.

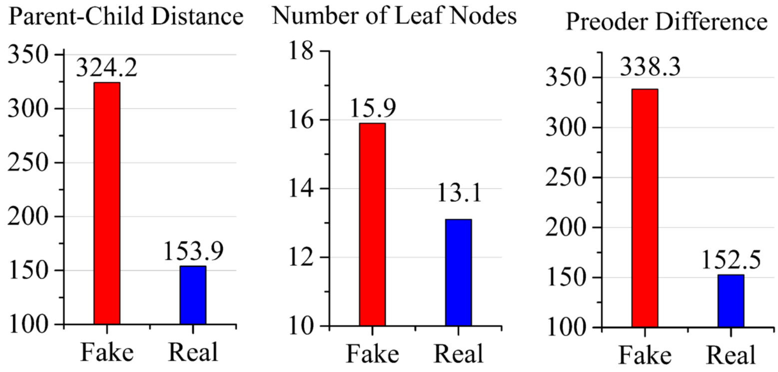


Figure 5: Average values of the proposed structure-related properties for fake and real news documents

Syntax-based approaches have been employed to take into account the hierarchical structure of textual documents for fake news detection (Feng et al., 2012; Pérez-Rosas and Mihalcea, 2015). One caveat of syntax-based approaches is their reliance on auxiliary parsing tools which might propagate error in later part of a developed model. Moreover, generating syntactic production rules in an automated manner is a complicated process.

Another way of incorporating structure is discourse-level parsing (Mann and Thompson, 1988; Li et al., 2014a; Ren and Zhang, 2016) which has seldom been explored for fake news detection. The noticeable exception is the approach developed by (Rubin and Lukoianova, 2015). They extracted a set of RST relations in fake and real documents and vectorized them using a Vector Space Model (VSM) method (Strehl and Ghosh, 2000). Unlike (Rubin and Lukoianova, 2015), in this work, we proposed an automated and data-driven discourse-level parsing approach which used neither any annotated corpus nor any external tool.

7 Conclusion and Future Work

In this work, we looked into fake news detection from a new perspective. We hypothesized that hierarchical discourse-level structure of news documents offers a discriminatory power for fake news detection. To investigate this hypothesis, we proposed a new framework HDSF, which can automatically extract discourse-level structures of real/fake news documents represented by dependency trees while does not rely on an annotated corpus. Moreover, we defined a set of insightful properties describing tree structures. Conducted experiments confirmed the power of our approach where it outperformed representative baselines. More importantly, we highlighted noticeable differences between structures of fake and real news documents. These differences also indicated less coherency in the fake news documents.

The new perspective pursued in this paper can be continued in several directions. First, we intend to define more advanced properties from the discourse dependency trees. Second, investigating the hierarchical structure at the word-level will be an exciting research inquiry. Finally, unsupervised discourse-level structure extraction of fake/real news documents is a worthwhile research topic.

Acknowledgments

This work is supported by the National Science Foundation (NSF) under grant numbers IIS-1714741, IIS-1715940, IIS-1845081, CNS-1815636, and a grant from Criteo Faculty Research Award. We would like to thank Mr. Tyler Derr and other members of the Data Science and Engineering (DSE) lab at Michigan State University[1] for their constructive comments.

References

Hadeer Ahmed, Issa Traore, and Sherif Saad. 2017. Detection of online fake news using n-gram analysis and machine learning techniques. In *International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments*, pages 127–138. Springer.

Hunt Allcott and Matthew Gentzkow. 2017. Social media and fake news in the 2016 election. *Journal of Economic Perspectives*, 31(2):211–36.

[1] http://dse.cse.msu.edu/

Joan Bachenko, Eileen Fitzpatrick, and Michael Schonwetter. 2008. Verification and implementation of language-based deception indicators in civil and criminal narratives. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 41–48. Association for Computational Linguistics.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Parminder Bhatia, Yangfeng Ji, and Jacob Eisenstein. 2015. Better document-level sentiment analysis from rst discourse parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2212–2218.

Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

Bella M DePaulo, James J Lindsay, Brian E Malone, Laura Muhlenbruck, Kelly Charlton, and Harris Cooper. 2003. Cues to deception. *Psychological bulletin*, 129(1):74.

Song Feng, Ritwik Banerjee, and Yejin Choi. 2012. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 171–175. Association for Computational Linguistics.

Camille Guinaudeau and Michael Strube. 2013. Graph-based local coherence modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 93–103.

Robert Hecht-Nielsen. 1992. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 13–24.

Hamid Karimi, Proteek Roy, Sari Saba-Sadiya, and Jiliang Tang. 2018. Multi-source multi-class fake news detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1546–1557.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. 2017. Structured attention networks. *arXiv preprint arXiv:1702.00887*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2039–2048.

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014a. Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2061–2069.

Sujian Li, Liang Wang, Ziqiang Cao, and Wenjie Li. 2014b. Text-level discourse dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 25–35.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 997–1006. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association of Computational Linguistics*, 6:63–75.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mathieu Morey, Philippe Muller, and Nicholas Asher. 2018. A dependency perspective on rst discourse parsing and evaluation. *Computational Linguistics*, (Just Accepted):1–54.

Myle Ott, Claire Cardie, and Jeffrey T Hancock. 2013. Negative deceptive opinion spam. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 497–501.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 309–319. Association for Computational Linguistics.

James W Pennebaker, Ryan L Boyd, Kayla Jordan, and Kate Blackburn. 2015. The development and psychometric properties of liwc2015. Technical report.

Verónica Pérez-Rosas and Rada Mihalcea. 2015. Experiments in open domain deception detection. In *Proceedings of the 2015 Conference on Empirical*

Methods in Natural Language Processing, pages 1120–1125.

Rashmi Prasad, Eleni Miltsakaki, Nikhil Dinesh, Alan Lee, Aravind Joshi, Livio Robaldo, and Bonnie L Webber. 2007. The penn discourse treebank 2.0 annotation manual.

Yafeng Ren and Yue Zhang. 2016. Deceptive opinion spam detection using neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 140–150.

Victoria L Rubin and Tatiana Lukoianova. 2015. Truth and deception at the rhetorical structure level. *Journal of the Association for Information Science and Technology*, 66(5):905–917.

Bernhard Scholkopf and Alexander J Smola. 2001. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Kai Shu, Suhang Wang, and Huan Liu. 2017. Exploiting tri-relationship for fake news detection. *arXiv preprint arXiv:1712.07709*.

Angelika Storrer. 2002. Coherence in text and hypertext. *Document Design*, 3(2):156–168.

Alexander Strehl and Joydeep Ghosh. 2000. Impact of similarity measures on web-page clustering.

Svitlana Volkova, Kyle Shaffer, Jin Yea Jang, and Nathan Hodas. 2017. 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)*, volume 2, pages 647–653.

William Yang Wang. 2017. " liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. 2015. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.