Dual-stream Self-Attentive Random Forest for False Information Detection

Manqing Dong[1], Lina Yao[1], Xianzhi Wang[2], Boualem Benatallah[1], Xiang Zhang[1], and Quan Z. Sheng[3]
[1]School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
[2]School of Software, University of Technology Sydney, Sydney, Australia
[3]Department of Computing, Macquarie University, Sydney, Australia
manqing.dong@unsw.edu.au, lina.yao@unsw.edu.au

***Abstract*—The prevalence of online social media facilitates massive knowledge acquisition and sharing throughout the Web. Meanwhile, it inevitably poses the risk of generating and disseminating false information by both benign and malicious users. Despite there has been considerable research on false information detection from both the opinion-based and fact-based perspectives, they mostly focus on tailored solutions for a particular domain and carry out limited work on leveraging multi-faceted clues such as textual cues, behavioral trails, and relational connection. We propose a novel dual-stream attentive random forest that is capable of selecting clues of discriminative information from individuals, collective information (e.g., texts), and correlations of entities (e.g., social interactions) adaptively. In particular, we use an interpretive attention model for learning textual contents. The model treats the important and unimportant content differently when constructing the textual representation and employs a multilayer perceptron to capture the hidden complex relationships among features of side information. We further propose a unified framework for leveraging the above clues, where we use attentive forests to provide probabilistic distribution as predictions over the two learned representations, which are then leveraged to make a better estimation. We conduct extensive experiments on three real-world benchmark datasets for fake news and fake review detection. The results show our approach outperforms multiple baselines in the accuracy of detecting false information.**

***Index Terms*—Attentive Forest, Dual-stream, False Information Detection**

I. Introduction

People are increasingly relying on social media as an alternative to TV channels and traditional publications to obtain daily information. Large volumes of false information are generated on a daily basis for various purposes such as rumor spreading, product promotion, or deliberately misleading people [1]–[3], due to the easiness of online information dissemination.

False information generally belongs to either opinion-based (e.g., fake reviews) or fact-based (e.g., fake news and hoaxes [4], [5]) false information. Opinion-based false information may appear on shopping websites, such as fake ratings and e-commerce reviews. Usually, people create this type of information for financial purposes such as inducing other people to make unwise decisions. Different from opinion-based false information, fact-based false information consists of lies about entities or events that have unique ground truths [1], [6]. Fact-based false information like fake news could spread fast and make a quick impact. In the long run, it adversely affects the way people respond to the real news and undermines the foundation of an honest society.

Many efforts have been devoted to detecting false information in the past decades. For detecting the opinion-based false information, researchers use clues from either textual information such as lexical features [7] or side information such as product-user relationship [8] and user's rating behaviors [2]. For example, many sophisticated fraudsters would not post duplicated or very similar reviews but semantically similar [9] to avoid being caught; typical users who spread fake e-commerce reviews have skewed rating distribution [10]. For detecting the fact-based false information, textual characteristics such as writing styles [11] and linguistic patterns play a vital role. For example, although fake news tends to pack the central claim of an article, a fake news article is usually short, repetitive, and less informative [4]. Other features like user characteristics and network characteristics are also concerned. For example, the creators of hoaxes typically have more recently registered accounts and less editing experience, according to [12].

Despite considerable efforts on understanding the false information, most existing work focus on either a particular task domain or a single feature domain. Although some work considers leveraging various types of features, most hybrid methods treat different types of features equally or are limited in exploiting the diverse relationship information. For example, the work in [13] detects fake news by concatenating and utilizing multiple types of features such as the text of an article, the user response it receives, and the source users who promote it for the classification. Here, we propose a unified framework that considers the common characteristics of both opinion-based and fact-based false information and leverages the clues with an attentive forest. We illustrate the building blocks of the proposed approach in Figure 1. Specifically, we first propose an attentive Bi-GRU model to learn hidden text representations and multilayer perceptrons to learn the side information. Then, we feed the learned representations into attentive forests where each tree provides a prediction. Finally, we design an attentive mechanism to adaptively select the most helpful trees and leverage the predictions from two attentive forests to make the final prediction.

In a nutshell, we make the following contributions:

- We propose a self-attentive ensemble model based on at-

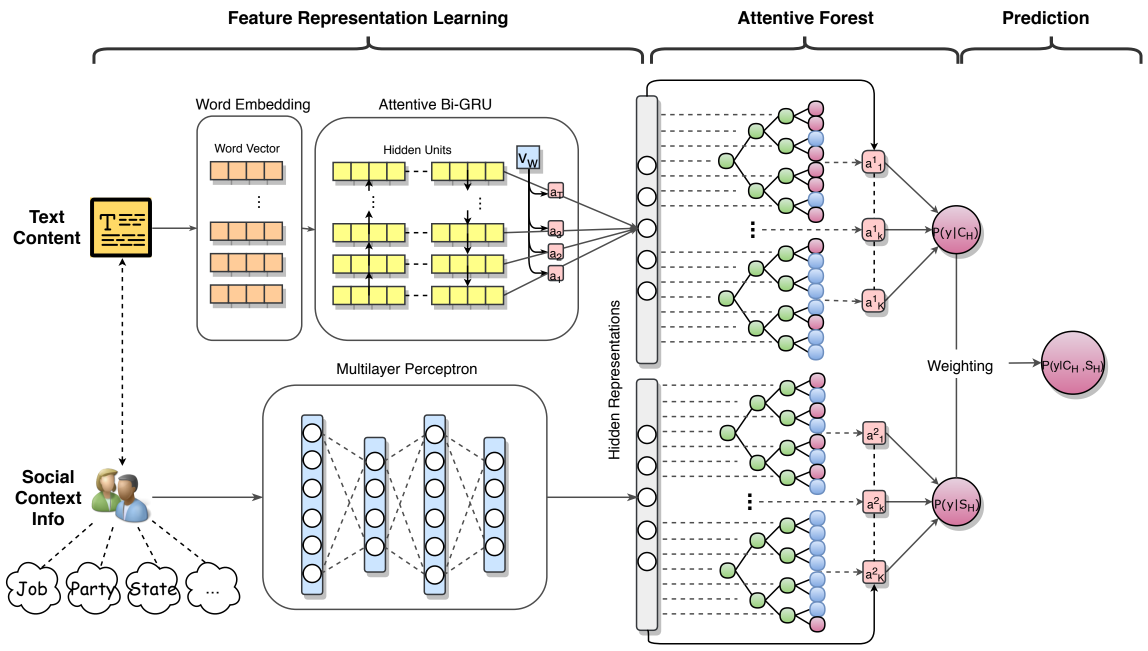

Fig. 1. An example of using our proposed model for the false information detection scenario. The left part is showing how we preprocess the input. The right side gives an example of the attentive forest: the leaf nodes in each tree are randomly initialized, where the red nodes are for false ones and the blue ones are for truths; the attention values for selecting the trees are then calculated (e.g., the bottom tree has a high probability of predicting the input as truths; thus we should try to decrease the attention value for this tree if the input are false ones); and the last, we balance the prediction from the two forest.

tentive forests for false information detection. The model leverages the advantages of both random forest and deep neural networks. The forest module helps alleviate the over-fitting problem, and the attentive module automatically distinguishes the importance of trees and chooses the trees that are helpful for the final prediction.

- We utilize a mechanism for preprocessing the two scope of features. The mechanism uses an attentive bidirectional Gated Recurrent Units (GRU) model to obtain robust representations of textual features and deep neural networks to transform the related side information into discriminating feature representations non-linearly. It is suitable for diverse false information datasets.
- We evaluate our framework on three real-world benchmark datasets for opinion-based and fact-based false information detection. The experimental results demonstrate that our framework can detect false information effectively and achieve superior performance to multiple baselines and competitive approaches.

II. Related Work

A. Opinion-based False Information Detection

A most common type of opinion-based false information is false reviews, an interesting topic that emerges with the development of e-commerce. Until now, most related work has been focusing on mining discriminate features from the linguistic [7], relational [8], and behavioral aspects [7] of review text or ratings. For example, Li et al. [2] modeled the distribution over reviewers' posting rates and utilized a coupled hidden Markov model to capture both reviewers' posting behaviors and co-bursting signals; Bhat et al. [14] focused on the content based information and combined positive unlabeled learning with domain adaptation to train a classifier. Apart from discussing on a single perspective features, some recent work has considered the collective feature domains. For example, Rayana et al. [10] employed Bayesian inference based on multi-perspective features to get the final prediction; Dong et al. [6] used auto-encoders for learning the hidden representation of various types of features and developed a neural forest for the classification.

B. Fact-Based False Information Detection

Fake news is one of the common types of fact-based false information. There has been plenty of work researching with single-domain information. The previous research focuses on either knowledge-based techniques [15] to detect the truthfulness of news or mining the writing style of the news [11]. For example, Shi et al. [16] used a knowledge graph to check whether the claims in the news content can be inferred from existing facts. Ma et al. [17] used a Recurrent Neural Network (RNN) to capture the changes in posts over time and then detected rumors. And [18] mainly considered textual information for detecting false information and uses three steps of attention, namely word level, sentence level, and headline-body attention for solving the problem. Similar to opinion-based false information detection, researchers also have considered various domains of information, such as text, relationship, speaker/writer files. One way to hybrid multi-domain information is extracting different kinds of features and feeding them directly into classification models. For example, Janze et al. [19] extracted different types of features: cognitive cues, visual cues, effective cues, and behavioral cues. They used a support vector machine (SVM) as their classifier. One shortcoming of this kind of method is that one model could not well grasp all different types of features, and they neglect the relationship between the features. Another way is to get hidden representations of different types of features first and then concatenate those representations. Most related works leverage deep learning models [13], [20]. Ruchansky et al. [13] considered three aspects of features: the text of an article, the user response it receives, and the source users promoting it. They proposed a model called CSI (capture, score and integrate), where 'Capture' used a Recurrent Neural Network (RNN) capture the temporal pattern of user activity, 'Score' learned the source characteristic based on the behavior of users, and 'Integrate' unified these two hidden representations to do the classification. However, a straightforward concatenation of the hidden representation vectors may exclude the relationship information.

III. The Proposed Method

In this section, we present our proposed method by first introducing methods for preprocessing the input and then the attentive forest. Our proposed model for detecting false information (shown in Figure 1) considers two types of features: textual features and side information.

A. Problem Statement

Let $C = \{c_1, c_2, \cdots, c_N\}$ be the text content information of users, and $S = \{s_1, s_2, \cdots, s_N\}$ be the side information of users, where N is the number of instances. We aim to learn hidden representation of two scope of features (C_H and S_H) and then input them into the attentive forest to predict the labels for the statements: $Y = attForest(C_H, S_H) = \{y_1, y_2, \cdots, y_N\}$.

B. Feature Representation Learning

User's side information $S \in \mathbb{R}^{M_s}$ could include both numeric features (e.g., the number of shares) and categorical features (e.g., user's job title), where M_s is the number of features. It is relatively easy to reprocess the numerical features; For categorical features with small and large factor sizes, we transform them with one-hot encoding (a group of bits in which all bits are '0' except one '1') and then transform them into the probability of being false information from the historical records, respectively. For example, if the user's job title is 'writer,' and 20% of the writer users produce false information in the historical records, then this feature for this user is valued at 0.20. The reason for using this method is that the one-hot encoding generally produces sparse and unbalanced feature vectors for categorical features (sometimes over 1000). We reuse S to denote the transformed features but use few fully-connected feed-forward neural networks, which could capture the implicit feature relationships, for learning the hidden representations S_H of those features:

$$S_H = \sigma(W_S S + b_S) \tag{1}$$

where W_S, b_S are the weights and biases, and σ is the activation function such as a sigmoid function. To be concise, we omit the explanation for W_* and b_* unless it is necessary.

For preprocessing the textual features, we first transform the words into word vectors with word embedding techniques [21]. Since each word vector is with the dimension M_c, we have $c_* \in \mathbb{R}^{N \times M_c}$. We use attentive Bidirectional-Gated Recurrent Unit (Bi-GRU) [22] to learn the hidden representation. Gated Recurrent Unit (GRU) [22] is an Recurrent Neural Network based model that uses two gates z_t and r_t to control how information is updated to the state, where t stands for time, or more specifically, the order of words in our context. Bi-GRU considers both directions of a sentence for the better word context. We have two GRU outputs: $\overrightarrow{h_t} = \overrightarrow{GRU}(c_t)$ and $\overleftarrow{h_t} = \overleftarrow{GRU}(c_t)$. Concatenating the two directional hidden units, we get $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$. We next add word attention to the hidden units:

$$v_t = tanh(W_\omega h_t + b_\omega) \tag{2}$$

$$\alpha_t = \frac{\exp(v_t^\top v_\omega)}{\Sigma_t \exp(v_t^\top v_\omega)} \tag{3}$$

$$C_H = \Sigma_t \alpha_t h_t \tag{4}$$

where v_ω is a randomly initialized vector, α_t is the attention value of t^{th} word in the text. Finally, we get the representation $S_H \in \mathbb{R}^h$.

C. Self-Attentive Forest

The neural probabilistic forest is a model that leverages the advantages of both random forest and deep learning methods. Random forest is an ensemble model that can alleviate the possible over-fitting of a single decision tree. Deep learning methods have the power of dealing with large scale datasets and mining the complex hidden relationship of features [23]. In a neural probabilistic forest, each decision node denotes the probability that one parent node delivered to the left or right sub-trees and finally reaches the leaf nodes; each leaf node contains a randomly initialized probability distribution over the prediction. The final prediction is made by averaging the predictions from the forest. The randomly initialized probability distribution over leaf nodes has the risk of being potentially biased for the final prediction. For example, for a neural tree with four leaf nodes, the probability of over 70% nodes of this tree predicting the same label is 25%. To minimize the potential cost of choosing the inappropriate trees, we use attention ideas to select the trees rather than to average over the trees. The selection helps improve the final performance while retaining the advantages of the ensemble model.

Suppose we have input X for one attentive forest that has K trees. Each tree T_k consists of decision nodes D and leaf nodes L. Consider the example shown in Figure 1, where green nodes represent decision nodes, blue or red nodes stand for leaf nodes, and each decision node has two branches. If we define the depth of the trees as q, we have 2^q decision nodes in depth q and 2^{q+1} leaf nodes. Specifically, the depth of the probabilistic tree in Figure 1 is 2.

Each decision node $d \in D$ holds a probabilistic distribution for deciding the probability of delivering the input from the parent node to the left or right branch. We use $\mathcal{D}_d(X) \in [0,1]$ to represent the probability that a sample reaching decision node d and then being sent to the left sub-trees, and $\overline{\mathcal{D}_d}(X) = 1 - \mathcal{D}_d(X)$ for the probability of the same sample being sent to the right sub-trees. The $\mathcal{D}_d(X)$ is calculated as follows:

$$\mathcal{D}_d(X) = \sigma(f_d(X)) \tag{5}$$

where $\sigma(x) = (1 + e^{-x})^{-1}$ is a sigmoid function, Θ stands for the set of parameters similar to what are used for learning the representation, and $f_d(X)$ is the transfer function for X, satisfying

$$f_d(X) = W_T X. \tag{6}$$

Since leaf nodes are the terminal nodes of the tree, each leaf node $l \in L$ holds a probability distribution p_l over Y. For example, considering a binary classification where each label is either 0 or 1, p_l could be $p_l = (0.2, 0.8)$ and thus this leaf node has high probability to predict the sample as label 1. We use P_{l_y} denotes the probability that leaf l predicts the input as label y. Then, the probability that a sample reach tree k to be predicted as class y would be the following:

$$P_{T_k}[y|X, \Theta, \Lambda] = \Sigma_{l \in L} P_{l_y} (\Pi_{d \in \mathcal{D}} \mathcal{D}_d^{\mathbf{1}_{left}} \overline{\mathcal{D}_d}^{\mathbf{1}_{right}}) \tag{7}$$

where $\mathbf{1}_*$ indicates the indicator function for the node goes left or right and Θ stands for the parameters used for learning X and Λ stands for the neural forest parameters. Suppose we have a forest with the depth of 1, meaning that we have 1 decision node as the top decision node d_1, and there are two decision nodes (d_2 and d_3) in depth 1 and 4 leaf nodes. Then, the probability of the sample reaches the first leaf node will be $d_1 d_2$, and the second leaf node $d_1 \overline{d_2}$.

We next construct a forest for those probabilistic trees. Since the probability distribution of the leaf nodes is randomly initialized, some trees might be biased (e.g., over 80% leaf nodes have a high probability of predicting label 1). We propose adding an attentive mechanism to help gain a better prediction. We call those focused trees *good trees* and the others *bad trees*. For learning those attention values, we first construct a tree indicator $U = (u_1, u_2, \cdots, u_K)$ where $u_k \in \mathbb{R}^{M_u}$ that contains the information about the trees. For each tree, we learn one attention head, which operates on an input sequence as $X = (x_1, \cdots, x_n)$ of n elements (where $x_i \in \mathbb{R}^{M_x}$) and then computes an attention value α_k, which is computed using a softmax function:

$$\alpha_k = \frac{\exp e_k}{\Sigma_{k=1}^K \exp e_k} \tag{8}$$

And e_{ij} is computed using a compatibility function:

$$e_k = (W_{T_k} X)^\top u_k \tag{9}$$

We employ dot product for the compatibility function to enable efficient computation, where linear transformations of the input add sufficient expressive power. We use $W_{T_k} \in \mathbb{R}^{M_u \times M_x}$ to denote parameter matrices, which are unique for each layer and attention head.

We get the forest of decision trees, i.e., an ensemble of decision trees $\mathcal{F} = \{T_1, ..., T_K\}$ that delivers a prediction for sample x by averaging the output of each tree as follows:

$$P_{\mathcal{F}}[y|x] = \Sigma_{k=1}^K \alpha_k P_{T_k}[y|x] \tag{10}$$

Given an attentive forest, the prediction for the label y would be as follows:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P_{\mathcal{F}}[y|x] \tag{11}$$

D. Prediction

We regard a user's side information and textual information as two scope of features. Thus, we use two attentive forests for learning each scope of features. We feed the hidden representation S_H to the first tree and C_H to the another and get $P_{\mathcal{F}}[y|S_H]$ and $P_{\mathcal{F}}[y|C_H]$. We then leverage the results by the following:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, \rho \cdot P_{\mathcal{F}}[y|S_H] + (1 - \rho) \cdot P_{\mathcal{F}}[y|C_H] \tag{12}$$

where ρ is a randomly initialized variable and is learned from the data during the training. This will be helpful for focusing on the features that are more helpful for the final predictions.

More generally, if we consider I different scope of features, we construct I attentive forests and add a series of hyper-parameters $\rho = (\rho_1, \cdots, \rho_I)$ for the results, which means $\hat{y} = \Sigma_{i=1}^I \rho_i \hat{y}_i$, and $\Sigma_{m=1}^M \rho_m = 1$.

E. Optimization

The overall goal is to minimize the loss between the predicted and real labels while ensuring each attentive forest to be good for the final prediction along with their input. The loss for each forest is the average of the loss in all of its trees, and the whole loss functions can be defined as follows:

$$\mathcal{L} = \mathcal{L}_A + \beta \cdot E_{F \in \mathcal{F}}(\mathcal{L}_F) \tag{13}$$

where $\mathcal{L}_A$ is the loss function for our prediction, which is taking the cross-entropy form:

$$\mathcal{L}_A = -\Sigma_{i=1}^K y_i \log(\hat{y}_i), \tag{14}$$

where β is the learning ratio and $\mathcal{L}_F$ is the loss function of each forest, (the second equal sign is a true statement when dealing with binary classification).

$$\mathcal{L}_F = E_{T \in F}(\mathcal{L}_T) = E_{T \in F}(-log(\mathbb{P}_T[y|x])). \tag{15}$$

Our final optimal function aim to find $\operatorname{argmin}_{\Theta, P} \mathcal{L}$ to minimize the whole loss function. The weights, biases and hyper-parameters are updated with the Adam optimizer [24]. Algorithm 1 shows the pseudocode for the training process.

Algorithm 1 Training Process

```
Require: Textual information $C$, side information $S$, according labels $Y$, the depth of trees $q$, and the number of trees in a forest $K$
1: Preprocess the side information and transform the textual information into word vectors.
2: Construct two forests with $K$ trees and $q$ depth
3: Initialize the parameter $\Theta$ and $\Lambda$ with random variables.
4: Randomly shuffle the dataset
5: for i in 1:$n\_epoch$ do
6:     for j in 1:$n\_batch$ do
7:         Learn $S_H$ by eq(1)
8:         Learn $C_H$ with attentive bi-directional GRU
9:         Feed $S_H$ and $C_H$ to separate random forests
10:        Learn $\alpha_S$ for side information by eq(8) to (9)
11:        Learn $\alpha_C$ for textual information by eq(8) to (9)
12:        Calculate the probability for each forest:
13:        $P_{\mathcal{F}}[y|S_H] = \Sigma_{k=1}^{K} \alpha_{Sk} P_{T_k}[y|S_H]$
14:        $P_{\mathcal{F}}[y|C_H] = \Sigma_{k=1}^{K} \alpha_{Ck} P_{T_k}[y|C_H]$
15:        Produce the prediction by eq(12)
16:        Calculate the prediction loss by eq(13)
17:        Update $\Theta$ and $\Lambda$ with Adam optimizer
18:    end for
19: end for
```

IV. Experiments

In this section, we report our extensive experiments conducted to evaluate the effectiveness of our proposed method on three benchmark datasets for false information detection. We have tested the model with different parameter settings and ablation studies to better extracting its characteristics.

A. Dataset Description

The first two datasets are for fake news detection, and the last is for fake review detection task.

- **PolitiFact dataset** contains 12,836 short statements from 3,341 speakers covering 141 topics in POLITIFACT[1]. Each statement includes text content, topic, and speaker profile.
- **Facebook Fact-Check Dataset**[2] contains 2282 posts published on Facebook from 9 news agencies over a week close to the 2016 U.S. election from September 19 to 23 and September 26 and 27. Every post and linked article was fact-checked claim-by-claim by 5 BuzzFeed journalists. Each post contains textual information as well as side information.
- **Amazon Review dataset** is a benchmark dataset for fake review detection. The raw dataset contains over 100 million reviews covering 24 product categories. Each review includes the textual information, user's ratings, received helpful votes, and review time.

[1]http://www.politifact.com/

[2]https://github.com/BuzzFeedNews/2016-10-facebook-fact-check

For each dataset, we use a binary classifier to distinguish between the truthful and false information. For the PolitiFact dataset, we regard the raw data with the labels 'true', 'mostly-true', or 'half-true' as truthful news and data with the 'mostly-false', 'false', or 'pants-fire' labels as fake news. For the FactCheck dataset, we regard data with the label 'mostly-true' as accurate news and others as fake news. For the Amazon review dataset, we obtained the labels from [25] and used a subset with 8000 samples. We filtered the dataset for keeping a balanced proportion for each class.

B. Comparison Method

We compare with several baselines and state-of-the-art methods for both fake news and fake review detection.

For the first task, we choose five methods for comparison: DT [26], SVM [19], Dual-LSTM [20], CSI [13], and EANN- [27]. The first two both considers various types of features for detecting the fake news. They choose the decision tree (DT) and SVM as their best classifiers, respectively. The last four are the most recent works and use deep learning based methods. Most of them use RNN for extracting textual information. For Dual-LSTM, the authors propose to utilize attentive-based Long Short-Term Memory (LSTM) network to learn context hidden representation and LSTM for learning the speaker's file. CSI uses a recurrent neural network (RNN) for capturing the article representation and a scoring method for learning the vector for users. It concatenates the learned hidden vectors and uses several neural network layers for the prediction. Event Adversarial Neural Networks (EANN) is a model that concatenate hidden visual features and textual features for detecting fake news and events. EANN- is the variant of their proposed model for fake news detection, which uses text-CNN for learning the hidden textual representation.

We either use the source code of the above methods or reproduce the structure written in the original paper together with parameter tuning. We ran each method ten times (with same initialized parameters) and gave the interval of its best prediction results over the test, and used the widely used metrics for comparing the classification results: accuracy, precision, recall, and F1 score [3]. Table I gives the comparison results, where the last two methods are ours: AttForest-C is the attentive forest fed with the concatenation of two scope of features and AttForest-2 is two attentive forests that follow the structure described in Figure 1. In each cell, the two numbers represent the interval for the prediction results.

Regarding fake review detection, we compare our methods with the following techniques, where the Amazon review dataset is also used in those works, and present the results printed in the original papers:

- Mukherjee et al. [28]: proposed unsupervised Bayesian approach that considers user's behavior features and bi-grams.
- Zhang et al. [29]: considered classical supervised classification methods (SVM, Naive Bayes, decision tree

[3]https://en.wikipedia.org/wiki/Precision_and_recall

TABLE I
COMPARISON RESULTS

Dataset	Metric	Method						
Fact-based								
PolitiFact		DT	SVM	2-LSTM	CSI	EANN-	AttForest-C	AttForest-2
	Acc	0.768/0.790	0.711/0.755	0.796/0.816	0.775/0.793	0.762/0.786	0.774/0.804	0.814/0.828
	Pre	0.767/0.778	0.640/0.656	0.725/0.747	0.758/0.770	0.707/0.764	0.774/0.796	0.762/0.783
	Rec	0.781/0.801	0.682/0.724	0.774/0.814	0.795/0.819	0.786/0.802	0.726/0.752	0.805/0.817
	F1	0.773/0.793	0.652/0.699	0.754/0.774	0.803/0.819	0.755/0.788	0.740/0.750	0.780/0.792
FactCheck		DT	SVM	2-LSTM	CSI	EANN-	AttForest-C	AttForest-2
	Acc	0.711/0.787	0.775/0.805	0.793/0.823	0.775/0.829	0.709/0.722	0.793/0.833	0.822/0.844
	Pre	0.715/0.789	0.801/0.811	0.791/0.835	0.729/0.787	0.744/0.767	0.756/0.829	0.800/0.869
	Rec	0.743/0.791	0.754/0.784	0.777/0.815	0.737/0.791	0.721/0.747	0.784/0.851	0.816/0.848
	F1	0.712/0.758	0.777/0.807	0.794/0.814	0.770/0.824	0.726/0.751	0.809/0.833	0.837/0.851
Opinion-based								
Amazon		Mukherjee et al.	Zhang et al.	Dong et al.	Heydari et al.	Bhat et al.	AttForest-C	AttForest-2
	Acc	0.8610	0.8781	0.9585	-	0.9700	0.913/0.948	0.960/0.967
	Pre	0.7960	0.8712	0.9608	0.8200	0.8580	0.903/0.913	0.942/0.951
	Rec	0.7510	0.8963	0.9415	0.8800	0.8280	0.963/0.977	0.955/0.968
	F1	0.7730	0.8831	0.9511	0.8600	0.8430	0.942/0.949	0.959/0.967

and random forest) that consider nonverbal features. We present the best result reproduced from their work.

- Dong et al. [6]: utilized neural forest that uses auto-encoders for learning robust hidden representations of features.
- Heydari et al. [30]: used time series methods that consider rating deviation, content-based factors and activeness of reviewers.
- Bhat et al. [14]: proposed positive unlabeled learning with utilizes textual features.

From the comparison results, we can observe that deep learning based methods normally perform better than traditional classifiers, which may due to the capability of the deep neural nets for learning the complex relationships. For example, the hyper-plane for SVM method will be fixed once be trained, which will be misfit for the test data that is highly distinctive with the training data; so do the tree-based methods, the decision rule will be fixed after training. Compare our methods with other deep learning based techniques, the most difference is that our method could adaptively extract the useful information for the final prediction and balance the weight of different types of features. Generally, models concerned with only side information perform better than models incorporating only content features; where in the experiments, we also found the learned ratio ρ for side information is higher than 0.5 (e.g. around 0.7 for the FactCheck dataset). And normally the hybrid models perform better than single-aspect models, which indicates there are some useful hints from the correlations of the two types of information. And our proposed model outperforms a series of methods over the different tasks, which proves the efficacy of the proposed method for detecting different types of false information.

C. Sensitivity Analysis

In this part, we perform the model's sensitivity to different parameter settings on FactCheck dataset. Similar results can also be found using other datasets. As mentioned above, we mainly have parameters in the hidden representation learning part and the attentive forest, as well as hyper-parameters for learning the forest. As a default setting, we randomly separate the training dataset with ratio 80%; the word vector is of dimension 100 , and each sentence is padded to the same length for the input of Bi-GRU model ; two fully connected neural networks are used for learning the side information, where each neural layer use Relu function as the activation function and a dropout layer to avoid over-fitting, where the settings for the number of nodes in the neural nets are based on the scale of the input; the hidden units in Bi-GRU are with size 100; there are 30 trees in each attentive forest, and each tree is with depth 3; and the default value of hyper-parameter β is 0.1. In particular, we compare the following different settings of parameters: the number of fully connected layers for learning the hidden representation of the side information; the activation function used in the neural networks; the number of trees in each attentive forest; and the depth of each tree. Figure 2 gives the comparison results on the test dataset, where the horizontal axis shows the learning epochs and the vertical axis stands for accuracy.

We could see that using one or two fully-connected layers for learning the hidden representation of side information give more stable results and show better performance in extracting the hidden representations. Two fully-connected layers might lead to over-fitting in the latter epochs, but it is hard for three fully-connected layers to capture the hidden complex representations. Activation functions in neural networks do not affect the results much. However, the Sigmoid function is somewhat unsuitable for our case. We could observe that small number of trees (e.g. 5 and 10) can already produce good performance for this case and is more stable. Typically, a deeper and larger attentive forest needs more time for learning, and the prediction results should be good with a considerable number of depths and trees. And the performance becomes better with higher training ratio. Generally, those results indicate the model could extract essential information

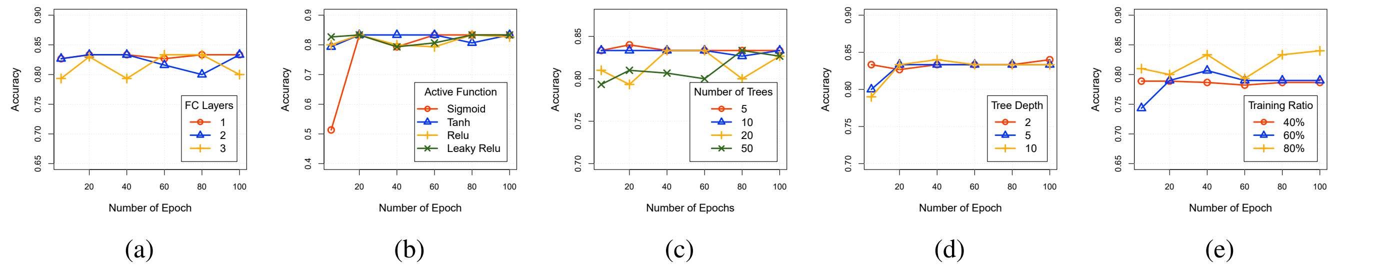

Fig. 2. Sensitivity towards different parameter settings: (a) number of fully connected layers for learning the hidden representation of side information; (b) the activation function used in the neural networks; (c) the number of trees K in each attentive forest; (d) the depth q of the tree; and (e) the ratio of the training dataset.

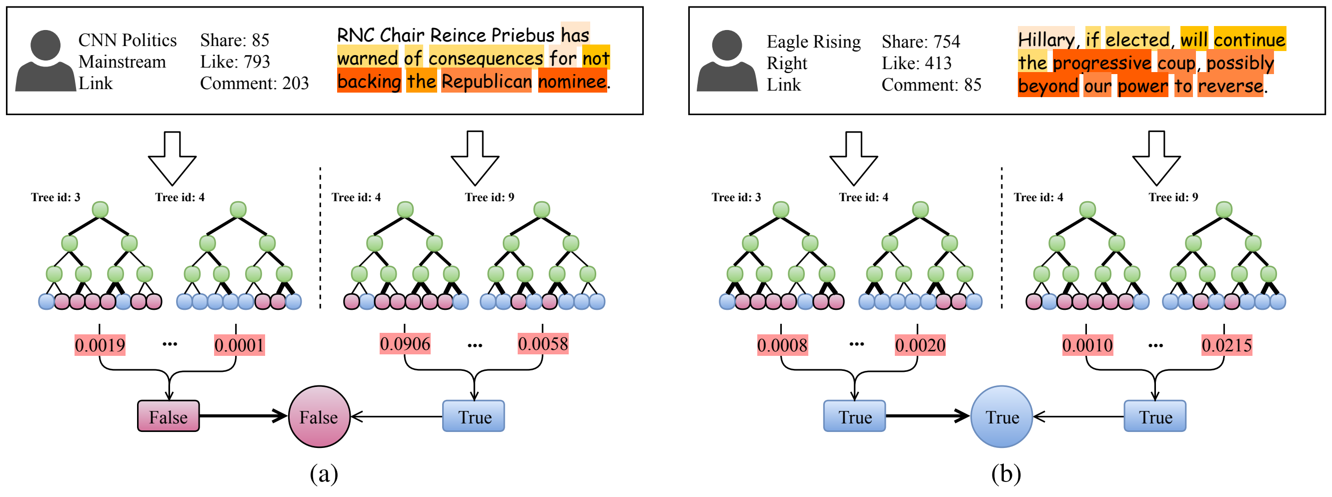

Fig. 3. Visualization of attentive forest on two examples of FactCheck, where (a) is for false information and (b) is a truthful one. The top box shows the raw information, where the colors on sentences represent the different attention values (darker color for higher attention value). The thicker lines in the neural tree show the flows with higher probabilities. Each neural tree follows with an attention value as the weight for selecting the trees, and each attentive forest will balance the trees to calculate a prediction. For leveraging the two scope of features, the automatically learned weight tends to impose more focus on the side information, as showed with thicker lines.

in different settings, and further demonstrate the superiority of the attentive forest in encoding the features.

D. Ablation Study

In this part, we study on the impact of different settings for the model, to find which part helps more with the final prediction. In particular, we consider the followings.

- Compare one attentive forest that concatenates the hidden representations from two type of information with our proposed method (shown in Figure 1).
- Discuss the efficacy of a single view of features.
- Compare the probabilistic forest (without attention on the trees) with the attentive forest.

We did ablation studies on FactCheck dataset, and table II shows the results. The first line is for the proposed method where we use two attentive forests for learning textual and side information, respectively. The Second method concatenates the hidden representation of two scope of features and uses an attentive forest as the classifier. The next two methods test the prediction performance of a single view of features using the attentive forest. The last three methods are based on the neural probabilistic forest.

TABLE II
ABLATION STUDIES ON FACTCHECK DATASET

Method	**Accuracy**
Text+Social+2 AttForest	0.844
Text+Social+Concat+1 AttForest	0.833
Text+1 AttForest	0.667
Social+1 AttForest	0.820
Text+1 ProForest	0.623
Social+1 ProForest	0.811
Text+Social+2 ProForest	0.822

We could observe that concatenation features could not efficiently leverage the multiple features, which indicate that, at least for the false information detection problem, different information contributes differently towards the final prediction. Similar conclusions can be found with the results for single review of features: for the FactCheck dataset, the textual information does not contain significant clues for identifying the false information; while models with hybrid features usually perform better than those based on single-domain of features. And compared with the neural probabilistic forest, the attentive forest does help with the prediction by the organic way of selecting the trees.

Here, we give two examples of correctly predicted false and truthful samples (shown in figure 3) to illustrate the attentive forest. For each sample, two biased trees are chosen for each type of features, and each tree follows with an attention value. The attention values for the trees vary with the samples and are helpful for the final predictions: e.g., in the false example, the attention value for tree 4 in the left side is quite low; while this value is turned to high in the second truthful case. When giving the final prediction, in this case, the learned weight focuses more on the side information, thus avoiding the mistakes made by the textual attentive forest.

V. Conclusion and Future Work

In this paper, we have proposed attentive forests for detecting opinion-based and fact-based false information. We first extract hidden representations from two scope of features: textural information and side information. Then, we feed the learned hidden representations into the attentive forests which will select the *good trees* self-attentively for the final prediction. We have conducted extensive experiments to evaluate the performance of the proposed model in detecting both opinion-based and fact-based false information. Experimental results on three benchmark datasets demonstrated the efficiency of our model. In this work, We mainly consider textual and side information due to the limitations of the datasets. In future work, we will consider more clues like images and videos. Besides, we aim to improve the capability of visualizing the relationships between those features and their contribution to the final prediction.

References

[1] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[2] H. Li, G. Fei, S. Wang, B. Liu, W. Shao, A. Mukherjee, and J. Shao, "Bimodal distribution and co-bursting in review spam detection," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 1063–1072.

[3] H. Li, Z. Chen, A. Mukherjee, B. Liu, and J. Shao, "Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns." 2015.

[4] S. Kumar and N. Shah, "False information on web and social media: A survey," *arXiv preprint arXiv:1804.08559*, 2018.

[5] M. Dong, L. Yao, X. Wang, B. Benatallah, Q. Z. Sheng, and H. Huang, "Dual: A deep unified attention model with latent relation representations for fake news detection," in *International Conference on Web Information Systems Engineering*. Springer, 2018, pp. 199–209.

[6] M. Dong, L. Yao, X. Wang, B. Benatallah, C. Huang, and X. Ning, "Opinion fraud detection via neural autoencoder decision forest," *arXiv preprint arXiv:1805.03379*, 2018.

[7] S. Feng, R. Banerjee, and Y. Choi, "Syntactic stylometry for deception detection," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, 2012, pp. 171–175.

[8] L. Akoglu, R. Chandy, and C. Faloutsos, "Opinion fraud detection in online reviews by network effects." *ICWSM*, vol. 13, pp. 2–11, 2013.

[9] V. Sandulescu and M. Ester, "Detecting singleton review spammers using semantic similarity," in *Proceedings of the 24th international conference on World Wide Web*. ACM, 2015, pp. 971–976.

[10] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*. ACM, 2015, pp. 985–994.

[11] V. L. Rubin and T. Lukoianova, "Truth and deception at the rhetorical structure level," *Journal of the Association for Information Science and Technology*, pp. 905–917, 2015.

[12] S. Kumar, R. West, and J. Leskovec, "Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes," in *Proceedings of the 25th international conference on World Wide Web*, 2016, pp. 591–602.

[13] N. Ruchansky, S. Seo, and Y. Liu, "Csi: A hybrid deep model for fake news detection," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 797–806.

[14] S. K. Bhat and A. Culotta, "Identifying leading indicators of product recalls from online reviews using positive unlabeled learning and domain adaptation," *arXiv preprint arXiv:1703.00518*, 2017.

[15] Y. Wu, P. K. Agarwal, C. Li, J. Yang, and C. Yu, "Toward computational fact-checking," *Proceedings of the VLDB Endowment*, vol. 7, no. 7, pp. 589–600, 2014.

[16] B. Shi and T. Weninger, "Fact checking in heterogeneous information networks," in *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 101–102.

[17] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha, "Detecting rumors from microblogs with recurrent neural networks." in *IJCAI*, 2016, pp. 3818–3824.

[18] S. Singhania, N. Fernandez, and S. Rao, "3han: A deep neural network for fake news detection," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 572–581.

[19] C. Janze and M. Risius, "Automatic detection of fake news on social media platforms," *Pacific Asia Conference on Information Systems*, 2017.

[20] Y. Long, Q. Lu, R. Xiang, M. Li, and C.-R. Huang, "Fake news detection through multi-perspective speaker profiles," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, vol. 2, 2017, pp. 252–256.

[21] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50, http://is.muni.cz/publication/884893/en.

[22] J. Song, J. Xiao, F. Wu, H. Wu, T. Zhang, Z. M. Zhang, and W. Zhu, "Hierarchical contextual attention recurrent neural network for map query suggestion," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 9, pp. 1888–1901, 2017.

[23] P. Kontschieder, M. Fiterau, A. Criminisi, and S. Rota Bulo, "Deep neural decision forests," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1467–1475.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] A. Mukherjee, B. Liu, and N. Glance, "Spotting fake reviewer groups in consumer reviews," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 191–200.

[26] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on twitter," in *Proceedings of the 20th international conference on World wide web*. ACM, 2011, pp. 675–684.

[27] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, and J. Gao, "Eann: Event adversarial neural networks for multi-modal fake news detection," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 849–857.

[28] A. Mukherjee, A. Kumar, B. Liu, J. Wang, M. Hsu, M. Castellanos, and R. Ghosh, "Spotting opinion spammers using behavioral footprints," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 632–640.

[29] D. Zhang, L. Zhou, J. L. Kehoe, and I. Y. Kilic, "What online reviewer behaviors really matter? effects of verbal and nonverbal behaviors on detection of fake online reviews," *Journal of Management Information Systems*, vol. 33, no. 2, pp. 456–481, 2016.

[30] A. Heydari, M. Tavakoli, and N. Salim, "Detection of fake opinions using time series," *Expert Systems with Applications*, vol. 58, pp. 83–92, 2016.