3-2023

Improving rumor detection by promoting information campaigns with transformer-based generative adversarial learning

Jing MA
Hong Kong Baptist University

Jun LI
University of Electronic Science and Technology of China

Wei GAO
Singapore Management University, weigao@smu.edu.sg

Yang YANG
University of Electronic Science and Technology of China

Kam-Fai WONG
Chinese University of Hong Kong

Improving Rumor Detection by Promoting Information Campaigns with Transformer-based Generative Adversarial Learning

Article *in* IEEE Transactions on Knowledge and Data Engineering · September 2021

DOI: 10.1109/TKDE.2021.3112497

CITATIONS
0

READS
158

5 authors, including:

Jing Ma
The Chinese University of Hong Kong
20 PUBLICATIONS **1,606** CITATIONS

SEE PROFILE

Wei Gao
Singapore Management University
98 PUBLICATIONS **2,739** CITATIONS

SEE PROFILE

Kam-Fai Wong
The Chinese University of Hong Kong
324 PUBLICATIONS **5,498** CITATIONS

SEE PROFILE

Improving Rumor Detection by Promoting Information Campaigns with Transformer-based Generative Adversarial Learning

Jing Ma, Jun Li, Wei Gao*, Yang Yang, and Kam-Fai Wong

Abstract—Rumors can cause devastating consequences to individuals and our society. Analysis shows that the widespread of rumors typically results from deliberate promotion of information with unknown veracity aiming to shape the collective public opinions on the concerned news event. In this paper, we attempt to combat such chaotic phenomenon with a countermeasure by mirroring against how such chaos is created in order to make automatic rumor detection more robust and effective. Our idea is inspired by adversarial learning method originated from Generative Adversarial Networks (GAN). We propose a GAN-style approach, where a generator is designed to produce uncertain or conflicting voices, further polarizing the original conversation threads with the intention of pressurizing the discriminator to learn stronger rumor indicative features from the augmented, more challenging examples. We reveal that feature learning effectiveness is highly relevant to the quality of generated parody, viz., how hard it is to get distinguished from real posts. Given the strong natural language generation performance of transformer, we propose a transformer-based method to improve the generated posts, so that they appear to be closely responsive to the source post and retain the authentic propagation structure and context of information. Different from traditional data-driven approach to rumor detection, our method can capture low-frequency but more salient non-trivial discriminant patterns via adversarial training. Extensive experiments on THREE benchmark datasets demonstrate that our rumor detection methods and the transformer-based model achieve much better results than state-of-the-art methods.

Index Terms—Information Campaigns, Rumor Detection, Generative Adversarial Networks, Transformer, Self-attention.

1 Introduction

The proliferation of rumors is a rampant phenomenon on social media. Rumor producers can cause devastating influences by manipulating public events. Information campaigns are frequently carried out by rumor makers via social networks to promote controversial memes, fake news, etc. with high volume of misinformation that is produced to compete with genuine information for dragging people's attention on to bogus claims. For example, during the US 2016 presidential election, Russia reportedly had coordinated thousands of "social bots" like covert human agents and automated programs to spread false information about the candidates by corroborating each other[1]. The widespread of rumors has triggered huge debates and has already unprecedentedly influenced US politics since then. In this paper, we attempt to "fight the evil with itself" to make automatic rumor detection more robust and effective.

Social psychology literature defines a rumor as a story or a statement whose truth value is unverified or deliberately false [1]. Fact-checking websites such as snopes.com and politifact.com rely on manual effort to track and debunk rumors, which have obvious limitation on efficiency and coverage. Existing automated methods typically resort to supervised classifiers trained on a wide range of features crafted from message contents, user profiles and diffusion patterns [2], [3], [4], [5], [6]. To avoid feature engineering effort, data-driven models were exploited more recently and demonstrated state-of-the-art detection performances. For example, Ma et al. [7] employed recurrent neural networks (RNNs) to learn hidden features from text content of relevant posts regarding given claims for detecting rumors. Yu et al. [8] used convolutional neural networks (CNNs) for obtaining local-global features from the relevant posts.

Nevertheless, existing data-driven approaches typically rely on capturing indicative responses such as skeptical and disagreeing opinions for detection. Rumor producers can take advantage of promoted campaigns to entangle public opinions or influence collective stances to get it widely disseminated and amplified. This poses a major technical challenge to data-driven methods as discriminant text patterns become distorted in substance. Various conflicting and uncertain voices that co-exist can seriously disturb the learning (or extraction) of useful features. Figure 1 illustrates a case of promoted campaign for the rumor about "Saudi Arabia beheads first female robot citizen"[2], which shows how the popular indicative patterns expressing skepticism and disagreement such as "fake news", "not sure", "no

- *Jing Ma is with Hong Kong Baptist University, Hong Kong. E-mail: majing@comp.hkbu.edu.hk*
- *Jun Li and Yang Yang are with University of Electronic Science and Technology of China, China. E-mail: lj_321@std.uestc.edu.cn, dlyyang@gmail.com*
- *Wei Gao is with Singapore Management University, Singapore. E-mail: weigao@smu.edu.sg*
- *Kam-Fai Wong is with The Chinese University of Hong Kong, HK. E-mail: kfwong@se.cuhk.edu.hk*

Manuscript received April 19, 2005; revised August 26, 2015.

1. http://time.com/4783932/inside-russia-social-media-war-america/

2. While being true that Saudi Arabia indeed has conferred its first "female" robot citizenship, it is false that the country beheaded the robot. It would be a valuable future research direction to investigate further from language understanding perspective for identifying which exact portion(s) of the claim makes it as a rumor or being false.

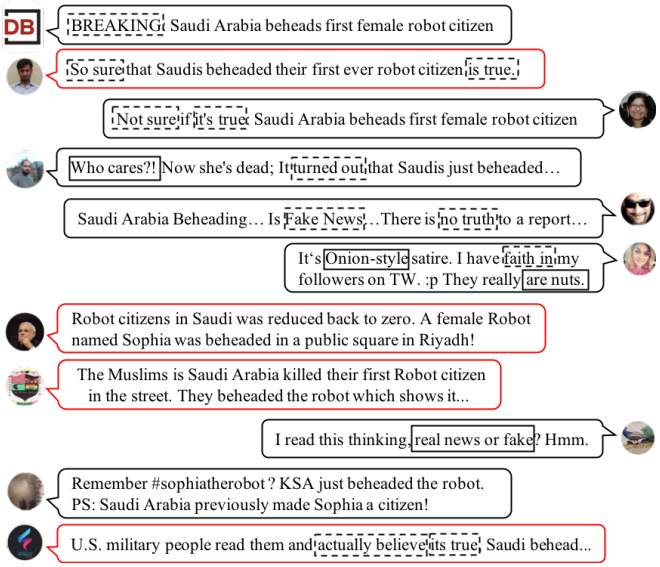


Fig. 1: Sample responses to a rumor claim about "Saudi Arabia beheads its first female robot citizen" in a promoted campaign. Social bots activities are marked as red box. The supportive responses are listed at left side and denial (or uncertain) responses at right side. Patterns captured by existing methods are marked by dashed rectangles, and patterns that can be missing are marked by solid rectangles.

truth" are inundated by the promoted posts. Therefore, developing a more robust feature learner for rumor detection is urgently desirable.

To the best of our knowledge, no existing rumor detection model has been focused on dealing with such promoted information campaign directly except our recent work [9], where we conducted a preliminary study by making use of adversarial learning framework to enhance the representation learning of current data-driven methods for detection and classifying rumors in social media. Our idea is motivated from the common problem that models trained on existing rumor detection datasets are vulnerable to information campaign, as genuine patterns can be pushed towards the long tail by the campaign such that useful features are weakened intentionally. Therefore, we proposed a novel method [9] based on Generative Adversarial Networks (GAN) [10], [11], in which a RNN-based generator is designed to produce campaign-like voices, and a rumor discriminator is learned to be more robust by capturing the low-frequency but non-trivial features. The overview of our proposed method is shown in Figure 2. This paper is a natural extension of our previous work [9] by 1) replacing the original sequence-to-sequence generator with a Transformer-based framework to greatly strengthen the post sequence representation and generation so as to improve the robustness of the detector; and 2) performing more in-depth experiments and analysis based on two balanced datasets (TWITTER and WEIBO) and an unbalanced dataset (PHEME), on which we obtain superior performance with the new rumor detection model.

Overall, we present a radically new rumor detection method by leveraging the mechanism of information campaign and promoting it in a controlled manner in order

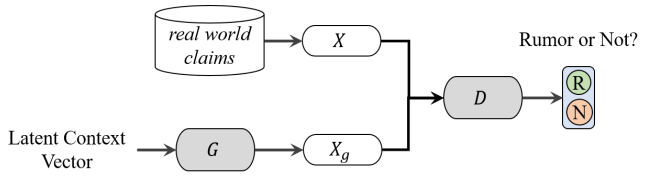


Fig. 2: The overview framework for our GAN-style rumor detection model, where G and D denote the generator and discriminator, the latent context vector is the source context of the claim to produce campaign-like voices.

to achieve more robust and effective detection. Our seemingly counter-intuitive idea is inspired by GAN [10], [11], where a classifier learns to distinguish whether an instance is from real world, and a generative model is trained to confuse the discriminator by generating approximately realistic examples. The harder are the generated examples to be distinguished from real-world ones, the stronger is the discriminator that can be learned. We train our generator to output challenging examples by mimicking a campaign promotion where misleading narratives and misled grassroots conversations are mixed in uncertain and conflicting voices, so as to push the discriminator to strengthen feature learning from such difficult examples by capturing more useful discriminating patterns. Unlike typical GAN-style models such as in computer vision [11] which aim to learn a stronger generator, our goal however is to force our rumor detector, i.e., *the discriminator*, to be more discriminative and robust. It is worth noting that the choice of discriminator is orthogonal to our proposed framework which can be easily replaced with any existing rumor detection model without any other change to our GAN-style architecture.

Intuitively, why can such a GAN-style method do better in feature learning? As shown in Figure 1, various users engagements can easily break the past data-driven methods that typically resort to repetitive patterns in responding posts. Due to the effect of generated campaign, the high-frequency patterns such as "fake news" or "no truth" commonly occurring in the responses to rumor claims and "be true/sure" in those to non-rumor claims become less discriminative. Therefore, the discriminator is expected to be able to adaptively focus on capturing the relatively less salient but useful patterns such as "onion-style" and "be nuts". Achieving this is non-trivial as their frequencies are relatively low and used to be paid no much attention in existing feature learning methods. Meanwhile, we also need to retain the discriminant power of the high-frequency patterns which are seriously distorted by the manipulation of campaigns.

Furthermore, a problem of our generative model proposed in [9] is that the real post sequences are encoded into one fixed length vector, from which decoder decodes the generated posts at each time step. Each time step in the model corresponds to a continuous time interval defined over the input sequence of posts (see Section 3 for detail). This may make it difficult for the model to cope with long post sequences, rendering the contents of the generated posts at each time step similar to each other. For promoting information campaigns, we expect to diversify word usage

and viewpoints in the generated posts at different time steps while keeping them topically coherent with that of the real posts. Inspired by the success of transformer network, we propose a novel generative model based on transformer. Different from using the fixed length vector to generate the whole sequence, our model devises a specific context vector at each decoding step, which can lead to diversified content being obtained by attending over the real input posts. We believe that generating a sequence of campaign-like post representations which embed more lifelike uncertain or conflicting voices would benefit adversarial learning. Meanwhile, the self-attention mechanisms used both in transformer encoder and decoder are expected to retain the properties of the generated sequence representation being in consistency with that of the real posts. We conduct extensive experiments on THREE microblog datasets of different languages and show that 1) the proposed GAN-style method yields substantial improvements over the state-of-the-art baselines; 2) the transformer-based model is more effective in generating campaign-like post threads representations; and 3) our method performs particularly well on early rumor detection which is crucial for timely intervention.

The main contributions of our paper are four-fold:

- This is a significant extension of the first generative approach for rumor detection using a text-based GAN-style framework in our recent work [9], where we make the text generator and discriminator strengthen each other for enhancing the adversarial training of rumor-indicative representations.
- We model rumor dissemination as generative information campaigns for generating confusing training examples to challenge the discriminator against its detection capacity. Furthermore, we enhance the text generator using transformer to produce more hard-to-distinguish campaign-like examples by leveraging the attention mechanisms to model the correspondence of replying posts across different time intervals during the spread of the source post. As a result, the generated text representations at each interval can be better aligned with the real-world posts semantically.
- Under the GAN-style framework, we boost our discriminator by training it on a set of more challenging examples replenished by the generator and reinforcing it to be focused on learning low-frequency discriminative features.
- We experimentally demonstrate that our model is more robust and effective than state-of-the-art baselines based on three public benchmark datasets for the tasks of rumor classification and early detection in social media.

2 Related Work

In this section, we provide a review of the research work on three different topics that are related to our study.

2.1 Rumor Detection

Automatic rumor detection in social media stems from the pioneering study of information credibility on Twitter [12]. Following that, numerous studies [3], [13], [14], [15] proposed to learn a supervised classifier by utilizing a wide range of features crafted from post contents, user profiles and propagation patterns. Zhao et al. [16] focused on early rumor detection by utilizing pre-defined regular expressions (e.g., "really?", "whaaat?!", etc.) to find questioning and denying tweets as the key for debunking. Subsequent studies worked on capturing temporal traits for understanding rumor diffusion. Friggeri et al. [17] and Hannak et al. [18] studied misinformation cascades by analyzing relevant comments on Facebook and Twitter with web links linking to specific rumor debunking websites. Kwon et al. [5], [6] introduced a time-series-fitting model based on the temporal properties of tweet volume. Ma et al. [4] extended the model using dynamic time series to capture the variation of social context features over time. On Twitter datasets collected during five breaking news stories, Zubiaga et al. [15] experimented Conditional Random Fields that leverages context learnt during an event for rumor detection. These approaches attempted to improve rumor detection performance via sophisticated pre-processing and feature engineering.

Data-driven methods are proposed to automatically capture rumor-indicative patterns from large-scale datasets. Ma et al. [7] employed recurrent neural networks (RNN), specifically LSTM and GRU, to learn hidden features from text content of relevant posts with respective to given claims for detecting rumors. Recently, they studied to mutually reinforce stance detection and rumor classification in a neural multi-task learning framework [19]. Alkhodair et al. [20] used an RNN model to detect breaking news rumors with an effect of mitigating news topic shift. Yu et al. [8] used convolutional neural networks (CNN) to obtain local-global features from the relevant posts sequence. To extract useful clues jointly from content semantics and propagation structures, kernel learning models [21], [22] and recursive neural networks [23] (RvNN) were exploited to differentiate various types of rumors based on the propagation trees of concerned claims. In recent years, attention mechanisms are successfully utilized for rumor detection. For example, the attention-based RvNN model [24] was designed to selectively focus on evidential posts in the propagation tree, and the Post-Level self-Attention Networks (PLAN) [25] was proposed to model latent dependencies between any pair of microblog posts in a cascade. However, these methods are less robust because they are purely data-driven and vulnerable to deliberate data manipulation and adversarial attack. For instance, when the thread corresponding to a rumorous claim is manipulated by adding amount of posts of support mouthpiece, the performance of these models may suffer. Yang et al. [26] proposed a rumor detection model by simulating Camouflages on Graph with Adversarial Training (CGAT), where four pre-defined camouflage strategies are utilized to dynamically add perturbations on the graph-structured social network to fool the rumor discriminator. However, previous studies [2], [4], [8], [19] show the graph structure (e.g., follow relations, reply/retweet relations) for rumor detection is not always available. Here we focus on the more general problem of content manipulation without considering the attack of graph structure. As with most of previous works, we formulate the task as classification at event (or claim) level where an event is comprised of a set

of relevant posts.

2.2 Adversarial Learning

Our work is inspired by the idea of adversarial learning based on Generative Adversarial Networks [10]. GAN are previously used to generate images [27], [28], [29] through a min-max game, in which a generative model is trained to generate real-like examples to fool a discriminative model. More specifically, there are two players under the adversarial framework, i.e., a generator G and a discriminator D. The learning process is that the generator tries to cheat the discriminator, but the discriminator tries to classify the fake and real instances correctly.

In recent years, GAN has demonstrated superior performance in a variety of Natural Language Processing (NLP) tasks, such as textual entailment [30], reading comprehension [31], dialog generation [32] and sentiment analysis [28]. Different from these previous works aiming to obtain an effective generator, we attempt to 1) build a stronger rumor detector via constructing a powerful generator that produces campaign-like posts more difficult for the discriminator to detect; and 2) differentiate rumors and non-rumors rather than the real and generated data, resulting in a different GAN mechanism.

2.3 Transformer Networks

Transformer is originally introduced as a sequence-to-sequence architecture, consisting of an encoder and a decoder. The encoder maps the input sequence into a dense vector, which is then fed into the decoder to generate an output sequence. Different from the conventional sequence-to-sequence model, transformer utilizes a multi-head self-attention mechanism [33] where a given element *attends to* the other elements in the same sequence. This process forces words related to each other to combine together, regardless of their positions in the sequence.

Due to the excellent capability of extracting latent features, transformer demonstrates state-of-the-art performance on various NLP tasks, such as sentence representation [34], neural machine translation [33], generative dialogues [35], machine reading comprehension [36] and semantic labeling [37]. Motivated by the successful applications of transformer, we propose a transformer-based model to generate the campaign-like social media posts representations, and adopt adversarial learning network to make a more robust rumor discriminator. This is the first instance of applying transformer in GAN for rumor detection task.

3 Problem Statement

In general, rumor detection task can be defined as a binary classification problem, which aims to learn a classifier from training claims labeled as **rumor** or **non-rumor** for predicting the label of a test claim. A claim is a factual (rather than opinionated) assertion or statement that something is true, such as the example statement "Saudi Arabia beheads its first female robot citizen" in Figure 1.

Typically, a claim is short which contains very limited context. For reliable feature extraction, in the task of rumor detection on social media (e.g., Twitter, Sina Weibo, etc.), a claim is commonly represented by a set of posts (e.g., tweets, or microblog posts in general) relevant to the claim which can be collected via the search function of social media platforms (e.g., Twitter search). Specifically, we represent a rumor dataset as a set of examples $\{X\}$, where each $X = (y,\ x_1 x_2 \dots x_T)$ is a tuple representing a given claim: the ground-truth label $y \in \{R, N\}$ of the claim (i.e., Rumor or Non-Rumor) and a sequence of relevant posts $x_1 x_2 \dots x_T$, where each x_t can represent a post or more generally a batch of posts in a time interval, and is indexed with a time step t. Thus, a claim can be considered as a time sequence of relevant posts. For clarity, we write a claim X as X_y, that is, X_R denotes a rumor and X_N a non-rumor.

4 Generative Adversarial Learning for Rumor Detection

Information campaigns pose tremendous challenges to existing rumor detection models since frequent patterns indicative of veracity become distorted and misleading. Our basic idea is to strengthen representation learning of rumor indicative features, inspired by the mechanism of generative adversarial learning [10]. We propose a GAN-style model where a generator attempts to promote campaigns by generating hard examples and a discriminator aims to identify robust features to overcome the difficulty.

Unlike the recent event adversarial model EANN [38] for multi-modal fake news detection and the neural user response generator TCNN-URG [39] for early detection, our idea and the adopted mechanisms are significantly different. Compared to EANN, our model learning is not only adversarial but also generative, for which we build a transformer-based post generator to fool the discriminator. Compared to TCNN-URG, our method is based on the GAN-style learning framework where the generator is just one of its components and our transformer-based generator demonstrates much better generation quality than the Two-Level Convolutional Neural Network (TCNN) used in the model which is crucial for improving the detection performance.

Unlike the common goal of a GAN discriminator which is to differentiate real and generated data, the goal of our discriminator is to classify rumor and non-rumor claims, which is orthogonal to the purpose of a general GAN discriminator. The key technical challenge is that our generator needs to effectively fool the discriminator by generating controversial posts to be added into each instance X to blur rumor and non-rumor classes from information credibility point of view, instead of generating new instances. This suggests the crucial distinction of our model with the general GAN model learning.

4.1 Controversial Example Generation

The goal of our generative model is to produce uncertain or conflicting voices conditioned on a given claim, making it hard to be differentiated as a rumor or non-rumor simply based on the used repetitive patterns. A straightforward way is to twist or complicate the opinions expressed in the original data via a handful of rule templates. For instance, we can 1) incorporate enquiry or questioning expressions such as "really?", "is it true?" etc. into responding posts;

2) negate the stance of a post by adding a "not" after a "be" verb; and/or 3) apply antonym replacement at certain parts of the keywords, e.g., by replacing "fake" with "true", "right" with "wrong", etc. However, it is difficult to generalize these rules for formally producing controversial voices of all kinds of possibilities. A general approach would be to cast our generator as trainable model that can cover a wide range of variations of expressions.

To this end, we design two generators, one for distorting non-rumor to make it look like a rumor, and the other for "whitewashing" rumor so that it looks like a non-rumor: 1) $G_{N\to R}$ generates skeptical or opposing voices against non-rumor claims; and 2) $G_{R\to N}$ generates supportive utterances towards rumor claims. We formulate a function f_g as:

$$X'_y = f_g(X_y) = \begin{cases} G_{N\to R}(X_y) & \text{if } y = N; \\ G_{R\to N}(X_y) & \text{if } y = R \end{cases} \tag{1}$$

where X_y is an original instance from training set which is either a rumor or a non-rumor, and X'_y is the transformed instance with the generator while the label remains intact.

One might question how to validate the generated posts is "opposing" or "supportive". We would like to provide some explanations on this further:

- Our design choice is that the model does not really generate any textual posts, and instead the generated information of our model is retained as vectorial presentations. This is because, from modeling point of view, 1) word-level generation will suffice for adversarial learning to work; 2) it can prevent malicious actors from abusing the generator to generate rumor campaigns in the real world. Also, the generated representation is not of an individual post, but of a batch of posts falling into each time interval. For this reason, there is no easy way to really output the textual posts. Note that our goal is to improve rumor detection performance via generation rather than the performance of producing campaign texts.
- The generation of adversarial examples is designed to be controlled and validated internally by the discriminator in the adversarial learning process (as described in Section 5.2). For instance, $G_{N\to R}$ is guided to generate examples that can flip the judgement of discriminator from non-rumor to rumor while whether the generated content can flip the class label is validated by the discriminator as part of the adversarial learning. Therefore, we do not need any explicit validation of the generation at text level. Our intuition is that adding controversial narratives tends to flip the prediction from non-rumor to rumor (since rumor is roughly characterized as receiving more uncertain and controversial feedbacks), and this shall hold true the other way round, meaning that if the discriminator flips its judgement from non-rumor to rumor it is very likely that the added information makes the posts threads more controversial. Basically, our generator is designed to generate posts representations that are able to flip the discriminator's prediction.

We will describe the relevant technical detail of the generator in this section.

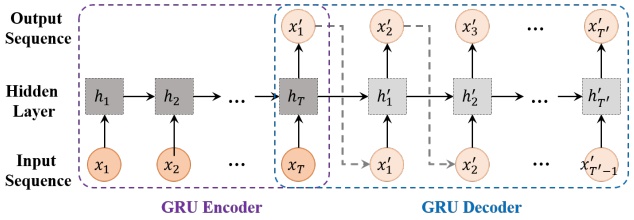


Fig. 3: Framework of our generator with a neural seq-to-seq model. x'_t is the transformation of x_t, assuming that the length of output sequence equals to that of the input.

4.1.1 RNN-based Generator

Considering the time sequence structure of posts in each instance, we use a sequence-to-sequence model [40], [41] for the generative transformation, which is illustrated in Figure 3. We encode an input sequence of relevant posts X_y into a dense vector via a RNN encoder, and then generate the transformed sequence of posts X'_y from it via an RNN decoder.

GRU-RNN Encoder: We batch relevant posts into time intervals and treat each batch of posts as a single unit in the time sequence, following the similar time segmentation described in [7]. Using RNN, we map each input unit $x_t \in X_y$ at time t into a hidden vector h_t, for which we use GRU [42] to store the hidden representation:

$$\tilde{x}_t = x_t E \tag{2}$$

$$h_t = \text{GRU}(\tilde{x}_t, h_{t-1}; \theta_g) \tag{3}$$

where x_t is the input unit represented as a vector of tf*idf values of vocabulary words computed from the posts falling into the t-th interval, and $E \in \mathbb{R}^{K\times d}$ denotes the embedding matrix. Note that we prune the vocabulary by keeping the top-K terms according to their tf*idf values, so the input dimension is K. In this way, x_t is a sparse vector, and E transforms x_t into a dense d-dimensional vector $\tilde{x}_t$ to represent the content at t-th interval. $\text{GRU}(\cdot)$ denote standard GRU transition equations, h_{t-1} refers to the hidden state of x_{t-1}, and θ_g represent all the parameters of GRU. The state of the last time step h_T from the encoder will be the representation of X_y. Note that the sequence length T is not fixed which can vary with different instances.

Noticeably, we batch the posts into time intervals because we have only a single output unit indicating the class at claim level while there could be a large number of relevant posts for a claim which can result in an excessively long post sequence if we treat each x_t as an individual post, making it impractical to use sequential models like RNN and transformer from the efficiency point of view. We choose tf*idf instead of transformer encoder to represent each interval because 1) transformer generally treats word sequence as input, while there could be tens of thousands of posts in a time interval resulting in an extremely long sequence, which is not effective nor efficient for model training; and 2) we empirically formulate a time interval as a bag of words involved, which is also consistent with the previous works [7], [8], [24], but transformer is not designed for handling such unordered textual data.

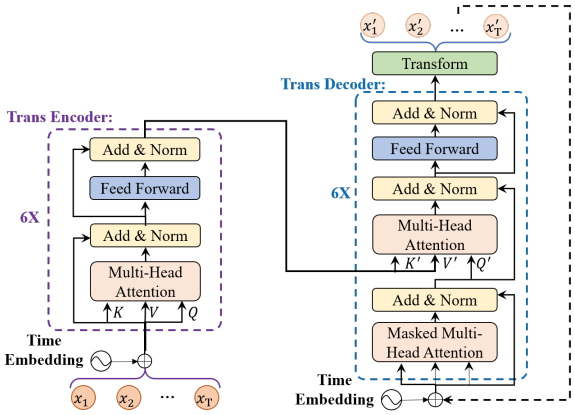


Fig. 4: The framework of Transformer-based generator. $\{x_1, \ldots, x_T\}$ is the tf*idf vector of the original posts sequence, and $\{x'_1, \ldots, x'_T\}$ is the vector containing words of the generated campaign-like posts.

GRU-RNN Decoder: The decoder transforms h_T into the generated sequence $X'_y = x'_1 x'_2 \ldots x'_T$. Each unit is sequentially generated using GRU followed by a pre-defined output function. In each time step t, an output layer maps the hidden state h'_t which is obtained via GRU into the target representation x'_{t+1} of a batch of posts by computing a distribution over all vocabulary words:

$$\begin{aligned} h'_t &= \text{GRU}(x'_t, h'_{t-1}; \theta'_g) \\ x'_{t+1} &= f(V_g h'_t + b_g) \end{aligned} \tag{4}$$

where h'_{t-1} is the hidden state of x'_{t-1} and θ'_g represents all the parameters inside the GRU[3]. In consistent with the form of input vector, the function f is defined to generate a vector of probability distribution over the vocabulary indicating the possible words appeared in each interval. Specifically, V_g and b_g are trainable parameters to transform h'_t into a K-dimensional vector, and $f(\cdot)$ is an activation function to generate a positive vector, e.g., ReLU, Sigmoid, Softmax, etc.

Particularly, the output of the decoder is either the word distribution or possible set of words with weights, which ignores the order of words. Previous studies [7], [8], [16] showed that cue terms itself can be indicative signals to identify rumors. On the other hand, the vector provides the keywords appearing in the campaign-like posts, capable of conveying salient information. From modeling point of view, we do not need to actually generate posts since 1) word-level generation will suffice for adversarial learning to work; 2) it can prevent malicious actors from abusing the generator to generate rumor campaigns in the real world.

4.1.2 Transformer-based Generator

A critical defect of the naive RNN-based generator is the incapability of capturing long-term dependencies of input sequences. For example, the generated campaign-like posts may be agnostic about the posts propagated at earlier time steps. Because the decoder is built on top of a singular vector h_T. Such a model cannot preserve long-distance information of the input [33]. Intuitively, information campaigns could happen throughout diffusion process, especially early-stage promotion could contribute to the widespread propagation of information. Therefore, we propose a transformer-based generator by creating shortcuts between the generated vector and the entire input post sequence, which shares similar intuition with the transformer-based Neural Machine Translation [33]. Figure 4 gives an overview of our transformer-based generator, which consist of two modules: a Transformer Encoder and a Transformer Decoder.

Transformer Encoder: Previous studies [5], [6], [22] reveal that rumor detection can benefit from taking into account the order of the input sequence. Thus, we utilize the positional encoding method in the conventional Transformer. Let t denote the position of x_t, and note that in our setting x_t corresponds to a batch of posts at the t-th time interval. Then the Position Embedding (PE) is defined as:

$$\begin{aligned} PE_{(t,2i)} &= sin\left(\frac{t}{10000^{2i/d}}\right) \\ PE_{(t,2i+1)} &= cos\left(\frac{t}{10000^{2i/d}}\right) \end{aligned} \tag{5}$$

where d is the dimension of x_t, and thus $i \in [0, \frac{d}{2} - 1)$. To incorporate propagation properties, we redefine Eq. 2 to concatenate the textual contents and positional embedding.

$$\tilde{x}_t = x_t E + PE_t \tag{6}$$

where $PE_t \in \mathbb{R}^d$ is a combination of $sine$ and $cosine$ functions, which encode the position of t in a post sequence.

Given a sequence of real posts $x = \{\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_T\}$, in the Multi-Head Attention layer, the input sequence x is firstly transformed into multiple subspaces with different linear projections. Attention functions are then applied to generate the output states:

$$O^h = \text{softmax}\left(\frac{Q^h \cdot {K^h}^\top}{\sqrt{d_h}}\right) \cdot V^h \tag{7}$$

where $\{Q^h, K^h, V^h\}$ are respectively the query, key and value representations, $\sqrt{d_h}$ is the scaling factor and d_h represents the dimensionality of the h-th head. Finally, the output of the representation is regarded as a concatenation of all the heads, which followed by a normalization layer and a feed-forward network.

Transformer Decoder: The decoder aims to generate the controversial voices $\{x'_1 x'_2 \cdots x'_T\}$ based on the original sequence $\{x_1 x_2 \cdots x_T\}$, where each x'_i is the distortion for x_i. To this end, we again utilize two attention mechanisms: (1) a masked multi-head self-attention layer to deduct the unseen time steps for the generated sequence; and (2) an encoder-decoder attention layer to enhance the correlations between the input sequence and the generated sequence.

Specifically, for the the generated sequence, after adding the positional embedding and linear transformation, we obtain $[Q^h_M, K^h_M, V^h_M]$ representing the query, key and value of the h-th head. In the Masked Multi-head Attention Layer, a matrix for masking $M \in \mathbb{R}^{T \times T}$ is defined to let the current step only attend over the previous time steps. Then the output of the masked attention layer is computed by:

$$O^h_M = \text{softmax}\left(\frac{Q^h_M \cdot {K^h_M}^\top}{\sqrt{d_h}} + M\right) \cdot V^h_M \tag{8}$$

3. Note that each generator ($G_{R \to N}$ or $G_{N \to R}$) corresponds to a unique sequence-to-sequence model. The encoding and decoding GRUs have similar yet different set of parameters: θ_g and θ'_g.

The updated representation of each generated time step is fed into a Normalization layer. To enhance the correlations between x_i and x'_i, we obtain the query and key based on the output of the encoder, and the value based on the generated sequence. Then an encoder-decoder attention layer which works like the self-attention mechanism is applied to update representation of the generated voices, which followed by a feed-forward and Normalization layers.

We now translate the obtained dense context vector into a vector indicating the possible words appearing in the t-th interval. The transformation is the same as that in Eq. 4.

Summary: Considering individual posts as input can only learn the semantics of individual posts but may fail to capture useful trend properties. Capturing the semantic trends hidden in post sequence during the lifecycle of information propagation is important for rumor detection. Our transformer-based model could enhance the representation learning of each time interval by attending over the other intervals having the similar trend properties, since the properties of trend have been encoded in the representations of sub-sequence of posts (i.e., the sequence of time intervals). Also, while both the RNN-based generator and the transformer-based generator aim to model sequences of time intervals, we conjecture that the transformer-based generator would be better due to the generally stronger representation power of transformer.

4.2 Rumor Discriminator

With the training data augmented from the generative processing, the discriminator learns to capture more discriminative features, especially low-frequency non-trivial patterns.

Our design choice of the discriminator is to allow the framework to use an existing rumor detection model without any other change to our GAN-style architecture. Our goal is to improving the effectiveness of discriminator via generating challenging training examples, instead of changing the discriminator itself by introducing more complex model structure into it such as creating a transformer-based discriminator. We believe this setting helps better justify the effectiveness of our GAN-based framework as a whole even though it only uses a less a powerful discriminator. Specifically, we consider three widely-used rumor detection models as the discriminator.

4.2.1 Bag-Of-Words (BOW) Rumor Detection Model

BOW model is a fundamental method of many NLP tasks. Based on the frequency of the words which appear in the relevant posts at t-th time step, we obtain the one-hot vector to represent x_t. Then we concatenate the vectors at each time step, which is then fed into a trainable classifier:

$$X = x_1 \oplus x_2 \ldots \oplus x_T \tag{9}$$

where $\oplus$ denotes the concatenation operation of two vectors, and $x_1 \in \mathbb{R}^V$ is a vector with V as the vocabulary size. Thus $X \in \mathbb{R}^{V \times T}$ represents the claim diffusion.

Our BOW model consists of two linear layers to map X to a hidden space, and utilize a 2-class softmax function to get the predicted probability of being a rumor or non-rumor.

$$\hat{y} = \text{softmax}(W_d^2(W_d^1 X + b_d^1) + b_d^2) \tag{10}$$

where W_d^1, b_d^1 and W_d^2, b_d^2 are the parameters corresponding to the linear layers of the discriminator.

4.2.2 RNN-based Rumor Detection Model

Given an instance (either original or generated), the RNN model [7] first maps relevant posts x_t at the t-th time step into a hidden vector s_t using GRU:

$$s_t = \text{GRU}(x_t, s_{t-1}; \theta_d) \tag{11}$$

where s_{t-1} is the previous hidden vector and θ_d denotes all the GRU parameters in discriminator.

Following [7], we feed the hidden vector s_T at the last time step as the representation into a 2-class softmax function for classifying the instance:

$$\hat{y} = \text{softmax}(W_d \cdot s_T + b_d) \tag{12}$$

where $\hat{y}$ is the vector of predicted probabilities over the two classes, W_d is the weight matrix of output layer and b_d is the trainable bias.

4.2.3 CNN-based Rumor Detection Model

Similar to the RNN-based model, CNN is successfully utilized for rumor detection [8]. Different with RNN-based model, the core idea of the CNN-based model is to capture the local rumor-indicative patterns from the post sequence via fixed-sized windows. Specifically, the post sequence is firstly represented as a matrix $S = [x_1; x_2; \ldots; x_T]$. The CNN-based model consists of two-level feature extractors. In the first level, a convolutional layer with a kernel with d as the window size is used to capture the local feature of S:

$$F[i] = \text{Conv}(S[:, i : i + d]; \theta_W) \tag{13}$$

where $\text{Conv}(\cdot)$ denotes convolution with the Frobenius inner product, $S[:, i : i+d]$ is the i-th to the $(i+d)$-th columns of the event matrix S, and θ_W denotes the kernel weights.

Based on the local features, a k-max-pooling layer is utilized to pick the top k conspicuous features to filter salient information:

$$F^k = \text{max-pooling}_k(\{F[1], \ldots, F[T-d]\}) \tag{14}$$

Repeating the above operation to the second feature extractor, i.e., a convolutional layer and a max-pooling layer, we again obtain a high-level local feature matrix F^c to extract c largest features. The prediction layer uses a fully-connected network to transform the feature matrix into a low-dimensional vector, which is followed by a softmax function for prediction the same as that in Eq. 12.

5 Model Training

5.1 Discriminator Training

The discriminator loss is defined as the square error between distributions of the predicted and the ground-truth class:

$$L_D(\bar{y}, \hat{y}) = ||\bar{y} - \hat{y}||_2^2 + \lambda ||\Theta_D||_2^2 \tag{15}$$

where $\bar{y}$ and $\hat{y}$ are respectively the ground-truth and predicted class probability distributions, Θ_D are discriminator parameters, $||.||_2$ is the L_2 regularization term over Θ_D, and λ is the trade-off coefficient.

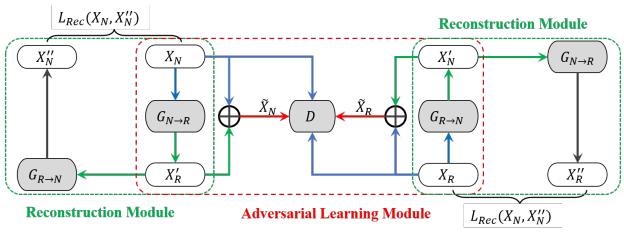


Fig. 5: Overview of our GAN-style rumor detection model. $G_{N\to R}$ ($G_{R\to N}$) is a generator. D denotes the discriminator.

5.2 Generator Training

In this encoder-decoder framework, $f_g(X_y)$ defines the policy that generates the representation of controversial posts which are used to confuse the discriminator based on the input X_y. A crucial issue is how to control the generator to generate the needed controversies. To this end, we use the performance of discriminator as a reward to guide the generator. The architecture and control mechanism of our GAN-style model is shown in Figure 5, which consists of an adversarial learning module and two reconstruction modules (one for rumor and the other for non-rumor).

Adversarial learning module: In our model, the generators are encouraged to produce campaign-like instances to fool the discriminator so that discriminator can be focused on learning more discriminative features. Such a goal suggests a special training objective resembling adversarial learning [10]. We formulate adversarial loss as the *negative* discriminator loss based on the generator-augmented data:

$$L_{Adv} = -L_D(\bar{y}, \hat{y}) \tag{16}$$

where $L_D(\cdot)$ is the loss between the ground-truth class probability distribution $\bar{y}$ and the predicted class distribution $\hat{y}$.

We combine the generated examples and original ones to augment the training set by taking the union of them, i.e., $\{\{X_y\} \cup \{\tilde{X}_y\}\}$, where $\tilde{X}_y = X_y \oplus X_y'$ is the elementwise addition of the representations of original and generated examples. Note that the elementwise addition has the effect to cancel out influential high-frequency patterns and promote the chance of important yet less frequent patterns for being selected. Meanwhile, we do not want to seriously weaken those useful features in the original example. Thus, the original example X_y is combined with $\tilde{X}_y$ for training.

Reconstruction module: It is likely that the generator could distort the original example towards unexpected direction by changing some essential aspects of a story. For example, the theme of "Saudi beheads robot citizen" might get distorted as "Saudi planned to invade Qatar" which is irrelevant and not helpful. To avoid that, we introduce a reconstruction mechanism to make the generative process reversible. The idea is that the opinionated voices will be reversible through two generators of opposite direction so as to minimize the loss of fidelity of information. We define the reconstruction function as follows:

$$X_y'' = f_r(X_y) = \begin{cases} G_{R\to N}(G_{N\to R}(X_y)) & \text{if } y = N; \\ G_{N\to R}(G_{R\to N}(X_y)) & \text{if } y = R \end{cases} \tag{17}$$

Algorithm 1: Generative adversarial training.

Input : A set of training claims $\{X\}$, learning rate ϵ

Initialize Θ_G, and Θ_D with random weight values;
for *epoch from 1 to maxIter* **do**
  for *each mini-batch* $\{\{X_N\},\{X_R\}\}$ **do**
    Generate $\{\tilde{X}_N\}$:
    $\{X_N \oplus G_{N\to R}(X_N)\} \to \{\tilde{X}_N\}$;
    Generate $\{\tilde{X}_R\}$:
    $\{X_R \oplus G_{R\to N}(X_R)\} \to \{\tilde{X}_R\}$;
    Augment training set:
    $\Big\{\{X_N\} \cup \{\tilde{X}_N\}, \{X_R\} \cup \{\tilde{X}_R\}\Big\}$;
    /* Minimize loss w.r.t. Θ_G */
    Compute gradient $\nabla(\Theta_G)$;
    Update generators: $\Theta_G \leftarrow \Theta_G - \epsilon\nabla(\Theta_G)$;
    /* Maximize loss w.r.t. Θ_D */
    Compute gradient $\nabla'(\Theta_D)$;
    Update discriminator: $\Theta_D \leftarrow \Theta_D - \epsilon\nabla'(\Theta_D)$;
  end for
end for

where X_y'' is the reconstructed instance from an original instance X_y via two opposite generators. We formulate the difference between X_y'' and X_y as a reconstruction loss:

$$L_{Rec} = \frac{1}{T}\sum_{t=1}^{T} ||x_t - x_t''||_2 \tag{18}$$

where x_t and x_t'' are the t-th unit representing in the original and reconstructed batch of posts at time step t, T is the posts sequence length, and $||\cdot||_2$ is the L2-norm of a vector.

Objective of optimization: The overall loss function of our GAN-style adversarial learning is defined as linear interpolation of L_{Adv} and L_{Rec}, and the objective of adversarial learning takes the following min-max form:

$$\max_{\Theta_D}\left(\min_{\Theta_G}(\alpha L_{Adv} + (1-\alpha)L_{Rec})\right) \tag{19}$$

where α is the trade-off coefficient between adversarial and reconstruction losses, Θ_G are generator parameters, and Θ_D are discriminator parameters. In the min-max process, we first optimize Θ_G by minimizing adversarial loss L_{Adv} (i.e., maximizing discriminator loss L_D) and reconstruction loss L_{Rec} to generate confusing but reversible examples; we then optimize discriminator parameters Θ_D for classification by maximizing adversarial loss L_{Adv} (i.e., minimizing discriminator loss L_D), and note that L_{Rec} is independent of Θ_D.

5.3 Overall Training

Algorithm 1 presents the iterative training process of the generators and discriminator in our GAN-style framework. Unlike original GAN [10] for obtaining better generators, our goal is to reinforce the discriminator to be more discriminative and generalizable. The discriminator is trained to differentiate the veracity, i.e., rumor and non-rumor, instead of real and generated examples, which is different from standard GAN, while the generator is trained to generate examples that are hard to identify rumors from non-rumors.

The generator and discriminator are alternately trained using stochastic gradient decent with mini-batches [43].

TABLE 1: Statistics of the datasets

Statistic	TWITTER	PHEME	WEIBO
Total # of Posts	822,924	105,354	2,796,938
Total # of Users	407,288	50,593	2,856,741
# of Claims	992	6,425	4,664
# of Non-Rumors	498	4,023	2,351
# of Rumors	494	2,402	2,313
Avg. time length	2,226 hours	15 hours	1,507 hours
Avg. # of posts	829	16	599
Max # of posts	41982	346	51271
Min # of posts	10	1	1

In each epoch, controversial examples are generated and augmented into the original training data. We optimize the generator and discriminator against the augmented training examples with Eq. 19 that enforces a min-max game through steps 8-11 in Algorithm 1. In the training, we initialize model parameters with uniform distribution and update them by the derivative of the loss through back-propagation [44]; we set the maximum epoch number as 200; the vocabulary size as 5,000, hidden vector dimension as 100, and tune the hyper-parameters α, λ and ϵ using a held-out dataset; we chop relevant posts into a time sequence following [7] which takes variable length dependent of specific instance. For the transformer-base model, the dimension of input post embedding, feed-forward layer, the head number and the dropout rate are respectively set as 300, 600, 5 and 0.5. We use ADAM optimizer [45] to speed up convergence. We warm up the training process with 1,500 steps for the generator and 4,000 steps for the discriminator. We initialize the learning rate as 0.0005 and the batch size is 50. The training process ends when the model converges or the maximum epoch number is met.

6 Experiments and Results

6.1 Datasets

We resort to three public datasets TWITTER [7], PHEME [46] and WEIBO [7] for experimental evaluation[4]. The three datasets were used for binary classification of rumor and non-rumor with respect to a claim via its relevant tweets. In particular, PHEME were collected based on 5 breaking news, thus its claims overlap more than TWITTER which was collected based on the claims reported on snopes.com. Moreover, the number of instances of the two classes in TWITTER and WEIBO is more balanced than that in PHEME, with which we will analyze the performance of our model on the balanced and unbalanced datasets. The reason we choose these datasets is that they are collected and widely used for the problem of rumor detection on social media, i.e., differentiating rumors from non-rumors. Statistics of the resulting datasets are given in Table 1.

6.2 Settings and Protocols

We made comparisons among the following models: 1) **DT-Rank**: A Decision-Tree-based Ranking method that identifies trending rumors [16] through searching for disputed claims; 2) **DTC**: A Decision Tree Classifier for modeling Twitter information credibility [2] using various hand-crafted features; 3) **SVM-TS**: A linear SVM classifier using time series to model the chronological variation of social context features [4]; 4) **BOW**: A naive baseline for rumor detection representing the texts using bag-of-words; 5) **RNN**: A RNN-based rumor detection model [7] with GRU for representation learning of relevant posts over time; 6) **CNN**: A CNN-based model for misinformation identification [8] for learning rumor representations by framing the relevant posts as fixed-length sequence. 7) **TCNN-URG**: The Two-Level Convoluational Neural Network with User Response Generator [39]; 8) **GAN (Generator+Discriminator)**: Our GAN-style learning model where the Generator can be RNN or Transformer and the Discriminator can be BOW, RNN or CNN.

We implement **DT-Rank**, **DTC** using Weka[5], **SVM-TS** using LibSVM[6] and all neural-network-based models with Pytorch[7]. We use micro-averaged and macro-averaged F1 score, and class-specific precision, recall and F-measure as evaluation metrics. We hold out 10% of each dataset for tuning the hyper-parameters, and conduct 5-fold cross-validation on the rest of data. We implemented all these models under comparison and release our source codes[8].

6.3 Results and Analysis

Table 2 demonstrates the performance of all the compared models based on the three datasets. The results indicate that our GAN-based models basically yield much better performance than all the baselines, which confirms the advantage of generative adversarial learning and Transformer-based generator for rumor detection task.

The three baselines in the first group using hand-crafted features perform worse than the three purely data-driven baselines in the second group, indicating that they are limited with respect to learning rumor-indicative features. Among them, SVM-TS is relatively better because it incorporate additional temporal information into the conventional models. The results of DT-Rank are poor due to the low coverage of the patterns in the three datasets it defined.

In the second group, BOW performs surprisingly well which is comparable to or even outperforms using hand-crafted features which confirms the advantage of using the simplest data-driven approach. RNN performs the best among all the baselines, since it takes advantage of deep neural networks to capture complex hidden features indicative of rumors beyond explicit and shallow patterns.

TCNN-URG is a CNN-based early detection approach that utilize user responses generated based on the claim text and historical user responses. So, TCNN-URG performs comparable with the three data-driven baselines in the second group on PHEME and WEIBO. But the claim text is not contained in TWITTER, that is the reason why the result of TCNN-URG is not satisfactory. Although TCNN-URG and our GAN-based method focus on generating additional user responses, our GAN-based method perform better because

4. This PHEME dataset is not the one widely used for stance detection [47]. It is defined specifically for rumor detection: https://figshare.com/articles/PHEME_dataset_of_rumours_and_non-rumours/4010619.

5. www.cs.waikato.ac.nz/ml/weka
6. www.csie.ntu.edu.tw/~cjlin/libsvm
7. pytorch.org
8. https://github.com/majingCUHK/Rumor_GAN

TABLE 2: Results of comparison among different models. (Prec.: precision; Rec.: recall; F_1: F-score.)

(a) TWITTER dataset

Method	Micro-F_1	Macro-F_1	Non-Rumor			Rumor		
			Prec.	Rec.	F_1	Prec.	Rec.	F_1
DT-Rank	0.666	0.663	0.716	0.519	0.602	0.652	0.814	0.724
DTC	0.734	0.732	0.778	0.675	0.723	0.694	0.794	0.741
SVM-TS	0.798	0.798	0.823	0.760	0.790	0.778	0.837	0.807
BOW	0.774	0.774	0.811	0.761	0.761	0.746	0.833	0.787
CNN	0.820	0.821	0.829	0.810	0.819	0.815	0.831	0.823
RNN	0.835	0.835	0.852	0.812	0.832	0.821	0.858	0.839
TCNN-URG	0.706	0.695	0.722	0.682	0.689	0.714	0.719	0.702
GAN (RNN+BOW)	0.791	0.791	0.830	0.733	0.779	0.761	0.850	0.803
GAN (RNN+CNN)	0.851	0.851	0.853	**0.852**	0.852	0.853	0.850	0.851
GAN (RNN+RNN)	0.862	0.862	0.885	0.833	0.858	0.843	0.892	0.866
GAN (TRANS+BOW)	0.802	0.801	0.794	0.815	0.804	0.813	0.788	0.799
GAN (TRANS+CNN)	0.863	0.862	**0.892**	0.831	0.859	0.839	0.896	0.866
GAN (TRANS+RNN)	**0.871**	**0.871**	0.890	0.851	**0.868**	**0.855**	**0.893**	**0.873**

(b) PHEME dataset

Method	Micro-F_1	Macro-F_1	Non-Rumor			Rumor		
			Prec.	Rec.	F_1	Prec.	Rec.	F_1
DT-Rank	0.657	0.545	0.695	0.867	0.772	0.472	0.239	0.317
DTC	0.670	0.625	0.687	0.837	0.755	0.572	0.435	0.494
SVM-TS	0.717	0.610	0.832	0.786	0.814	0.318	0.541	0.405
BOW	0.756	0.718	0.822	0.824	0.823	0.636	0.592	0.613
CNN	0.754	0.733	0.803	0.816	0.804	0.702	0.653	0.661
RNN	0.775	0.745	0.825	0.840	0.832	0.667	0.643	0.658
TCNN-URG	0.774	0.756	0.809	0.837	0.823	0.711	0.668	0.688
GAN (RNN+BOW)	0.775	0.748	0.786	0.881	0.831	0.750	0.599	0.666
GAN (RNN+CNN)	0.798	0.779	0.820	0.869	0.844	0.756	0.680	0.716
GAN (RNN+RNN)	0.819	0.807	0.754	0.858	0.856	0.761	0.754	0.757
GAN (TRANS+BOW)	0.781	0.753	0.787	**0.892**	0.836	**0.767**	0.595	0.670
GAN (TRANS+CNN)	0.807	0.790	0.829	0.871	0.850	0.764	0.699	0.730
GAN (TRANS+RNN)	**0.821**	**0.809**	**0.858**	0.858	**0.857**	0.765	**0.760**	**0.760**

(c) WEIBO dataset

Method	Micro-F_1	Macro-F_1	Non-Rumor			Rumor		
			Prec.	Rec.	F_1	Prec.	Rec.	F_1
DT-Rank	0.732	0.731	0.726	0.749	0.737	0.738	0.715	0.726
DTC	0.831	0.830	0.815	0.847	0.830	0.847	0.815	0.831
SVM-TS	0.857	0.859	0.878	0.830	0.857	0.839	0.885	0.861
BOW	0.850	0.849	0.954	0.742	0.835	0.782	0.963	0.863
CNN	0.914	0.913	0.942	0.883	0.907	0.900	0.944	0.919
RNN	0.899	0.899	0.940	0.852	0.894	0.865	0.946	0.904
TCNN-URG	0.854	0.853	0.841	0.877	0.858	0.870	0.830	0.849
GAN (RNN+BOW)	0.931	0.931	0.948	0.912	0.930	0.915	0.949	0.932
GAN (RNN+CNN)	0.929	0.929	0.944	0.913	0.928	0.915	0.945	0.930
GAN (RNN+RNN)	0.940	0.939	0.953	0.926	0.939	0.928	0.952	0.940
GAN (TRANS+BOW)	0.935	0.935	0.950	0.919	0.935	0.921	0.951	0.936
GAN (TRANS+CNN)	0.933	0.933	0.949	0.917	0.932	0.919	0.949	0.934
GAN (TRANS+RNN)	**0.952**	**0.952**	**0.960**	**0.944**	**0.952**	**0.944**	**0.960**	**0.952**

1) TCNN-URG produce user responses similar to the historical posts thus no new pattens could be captured; and 2) TCNN-URG requires long claim text, which may not be generalize well on our three datasets with shorter claim text.

On balanced dataset TWITTER and WEIBO, our adversarial models outperform their counterparts in baselines that are not generative adversarial. The improvements of our models over these baselines range from 3.6% (1.0%) to 5.2% (5.9%) on TWITTER (WEIBO) in terms of accuracy, indicating that the adversarial learning with the generative-discriminative process is generally helpful and effective. Among our three GAN-based models in the third group, GAN (RNN+RNN) performs the best, since both generator and discriminator explore the sequential nature of the relevant posts. Our extension of GAN-based model using transformer consistently outperforms their counterparts without transformer, suggesting the effectiveness of our transformer-based adversarial learning for enhancing the conflicting voices generation. We also observe that GAN (TRANS+RNN) achieve the best in accuracy, since it take advantages of the sequential nature and the representation power of transformer. Furthermore, rumor detection task emphasizes the identification performance on the rumor category. GAN (TRANS+RNN) achieves the highest recall on rumor category, indicating that more rumors can be found.

6.3.1 Results on unbalanced dataset PHEME

In Table 2(b), we observe that all the feature engineering-based systems perform worse than the data-driven method. We conjectured that CNN should be comparable to RNN because both can learn deep latent features from data. This turned out to be incorrect on PHEME where CNN performs

much worse. The reason is that RNN can inherently deal with variable-length sequence while CNN is essentially not a sequential model. All the post sequences are zero-padded into the same length when using CNN. We observe that the relevant posts per claim in PHEME is significantly fewer than TWITTER and WEIBO (see Table 1), rendering lots of zero-valued input units to CNN that can worsen convolution operations, but RNN can easily get rid of zero input units by shortening sequence length.

Our proposed GAN-based approaches outperform their counterparts without adversarial learning. Specifically, the improvement made by GAN-style framework ranges from 4.9% to 8.6% in terms of macro-F1. We observe the improvement obtained on TWITTER/WEIBO datasets is relatively lower than that on PHEME. Two reasons may prevent further improvements on TWITTER/WEIBO: 1) The post content associated with PHEME claims overlap heavily since they come from only 5 breaking news, rendering high-frequency patterns more sensitive to topical categories than veracity categories. The claims in TWITTER/WEIBO dataset are however easier to classify as per veracity since each of them is an independent news topic, making high-frequency patterns well correlated to veracity rather than topic[9]. The potential of GAN-style learning by promoting the chance of low-frequency patterns is thus somewhat limited on TWITTER/WEIBO. 2) In addition to overlapping topics, PHEME is harder to classify also because there are only 16 posts per claim in average, thus useful information available is relatively limited based on the conventional data-driven method. In contrast, generating more high-quality posts for each claim is helpful for model generalization.

Among all the GAN-based methods, we observe the superiority of using the transformer-based generator is relatively less obvious on PHEME dataset than that of using the RNN-based generator. This might be due to the generally short-length post sequence in PHEME dataset, which verifies the hypothesis we made in Section 4.1.2 that transformer can effectively cope with long post sequence.

6.4 Early Rumor Detection

Early alerts of rumors can prevent further spreading of rumorous contents. By setting a detection checkpoints of "delays" that can be either the time elapsed or the count of corresponding posts since the first posting, only tweets posted no later than the checkpoints is available for model evaluation. The performance is evaluated by the detection accuracy (i.e., micF) obtained at each checkpoint. To satisfy each checkpoint, we incrementally scan test data in order of time until the target time delay or post volume is reached.

Figure 6 shows the performance of GAN (TRANS+RNN) and GAN (TRANS+CNN) versus GAN (RNN+RNN) (the best performed GAN-based model without transformer), RNN (the best performed data-driven system), SVM-TS (the best system using feature engineering) and TCNN-URG (an early-detection-specific algorithm) at various checkpoints.

Clearly, the accuracies of all systems increase with elapsed time or post counts and our GAN-based models grow more quickly and start to supersede the other baselines at the early propagation stage. Particularly, GAN (TRANS+RNN) only need around 10 (7) hours or about 30 (20) posts on TWITTER/WEIBO (PHEME) to achieve the comparable performance of the best baseline model, i.e., RNN, which needs about 36 (24) hours or around 100 (50) posts on TWITTER/WEIBO (PHEME), indicating superior early detection performance of our method. GAN (TRANS+RNN) achieves 83.3% (80.5%) accuracy on TWITTER (PHEME) within 12 hours and 91.4% accuracy on WEIBO within 8 hours, that is much faster than other models. This is because generation component can enrich training data at early stage when the volume of real posts is generally low. Moreover, on PHEME, all the systems keeps comparable performance at the first 6 hours or 10 tweets because the number of non-rumor is much larger than the rumors, and the methods are better off on the majority class.

We also examine the impact of transformer mechanism on early detection by comparing GAN (RNN+RNN) and GAN (TRANS+RNN). We observe that they perform comparably using less than 8-hour or about 10 tweets on the three datasets, indicating that the RNN-based generator perform well compared with the Transformer-based generator for the short sequence posted at early stage. With more posts available, the length of the posts increases and GAN (TRANS+RNN) exceeds GAN (RNN+RNN), suggesting that generating conflicting voices consistent with the properties of original post sequence is helpful to early detection task.

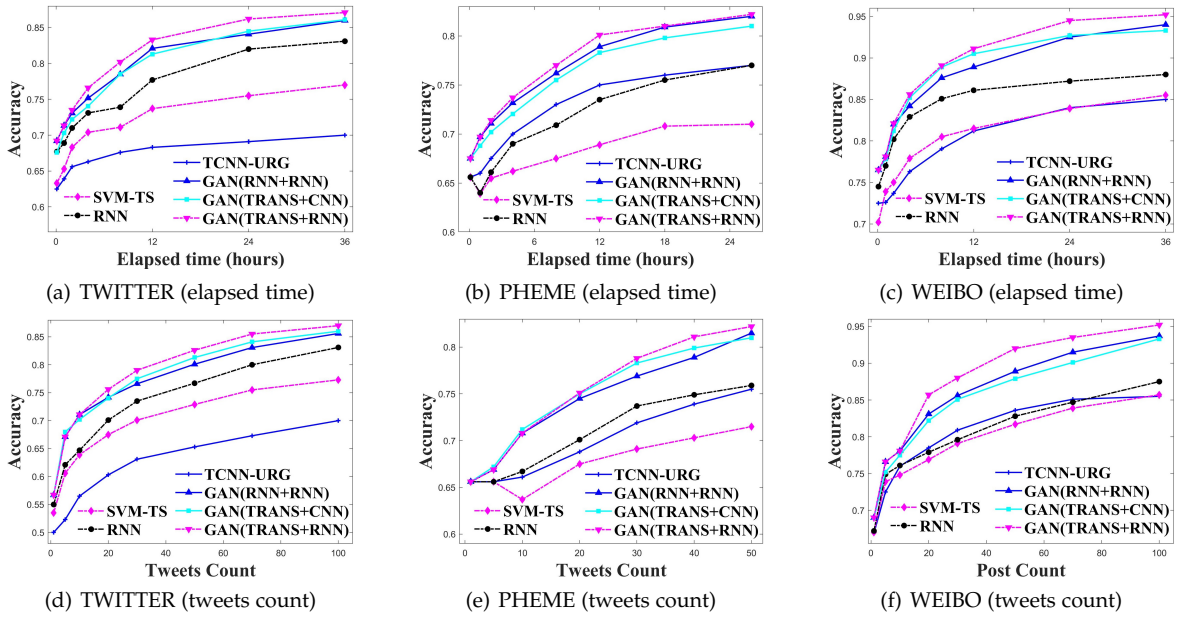
6.5 Case Study

We conduct experiments to explain why the generator can boost the discriminator in an adversarial manner. We sample a rumor and a non-rumor claim from TWITTER. With the real post sequence of each claim as input, we output the decoded vector at each time step via the generative model and list the words with higher value. Specifically, we analyze the behavior of RNN-based generator and Transformer-based generator respectively used in GAN (RNN+RNN) and GAN (TRANS+RNN), due to their superior performances.

In Table 3, we observe that 1) the generations although seemed non-grammatical are relevant to the input claim; 2) the generations can distort the real posts by using conflicting or uncertain expressions, e.g., the supportive terms (like "believing", "confirms") in the rumor claim, and the questioning patten (like "suspect", "right?") in the non-rumor claim; 3) incorporating the generations counterbalances the discriminative power of high-frequency patterns (in yellow), implying a higher chance of lower-frequency features (in blue) being captured by the discriminator; 4) the generated voices from our GAN (TRANS+RNN) show obvious temporal pattens consistent with the properties of the input, e.g., the generated keywords (underlined) at each time step closely correspond to that of real posts.

Given the limited number of claims in the datasets, our models perform reasonably well. This is because there are a good number of relevant tweets per claim, where many indicative patterns exist, such as those highlighted keywords and phrases in Table 3. Therefore, the feature space is not sparse, which is generally advantageous for generators to generate diverse campaign-style texts and for the discriminator to capture discriminative features.

9. This can be confirmed by the relatively higher performance of all models on TWITTER/WEIBO.



(a) TWITTER (elapsed time) (b) PHEME (elapsed time) (c) WEIBO (elapsed time)

(d) TWITTER (tweets count) (e) PHEME (tweets count) (f) WEIBO (tweets count)

Fig. 6: Results of rumor early detection

TABLE 3: Examples of original and generated posts for a Non-rumor claim (left) and a rumor claim (right). (ts: time step)

Claim	**[Non-rumor]** Anti-U.S. beef protest draws 100,000 S.Koreans		**[Rumor]** American soldier was refused service at a gas station	
	ts	*Post Contents*	ts	*Post Contents*
Real Posts	1	An article in the WSJ says 9,000 people in South Korea marched in protest against US beef	2	American soldier in uniform refused service in BOGALUSA, LOUISIANA! ENOUGH IS ENOUGH
	2	Dont think I'll ever understand South Korea. Current protests stemming ostensibly from US beef - really?	3	Anyone near Bogalusa LA. join this protest at gas station who refused service to a US Soldier
	8	South Koreans Fill Streets of Seoul to Continue Protest Against U.S. Beef Imports: For the past two ..	11	Bogalusa, LA - National Guardsmen service rejection story most likely not true...
	48	U.S. beef protester fined for embezzlement (Yonhap) A Seoul court fined a South Korean protester W1 million..	43	Outrageous ! Soldier denied service Bogalusa gas station at center of protest
$G_{N\to R}$ or $G_{R\to N}$	1	horrible report that society chaos, public right	2	Oh, soldiers who nationwide defend hero refused.
	2	sign + rt ! video are that horrible, suspect	3	Many citizen protest, joke ?
	8	protests are stopped, believe? breaking video show protesters scare of...	11	Reported Should wait truth in protest, does agrees ?
	48	correct ? protesters guard health. jokes right ?	43	Allegedly videotaped not clearly. Citizen looks happy, heartwarming, not disgusting
$G_{N\to R}$ or $G_{R\to N}$ ($Trans$)	1	protesters against about some america foods. what jokes ? What happen?	2	Protest soldier refused in gas service.trust ?
	2	officials should say some things about the chaos of american foods right ? people suspect it.	3	Any guy near LA maybe protest this soldier refused.
	8	Happening shows people may mad about american beef. what is officials statement ?	11	Reports maybe wrong, official statement is right, waiting
	48	may be another controversy ban US foods, right ?	43	Hahahaha, fact of gas following officials, everyone believing

6.6 Discussion

How to apply the proposed framework in real-world. The task of online rumor detection is generally more challenging due to the heavily unbalanced distribution between rumors and non-rumors in real-world systems. As per our statistics based on Snopes.com since January 2015, the proportions of claims under the non-rumor category is around 76% [19]. Particularly, the ratio of rumors versus non-rumors are dynamically changed over time, and it is very difficult (if not impossible) to know what the actual ratio is in the real world. However, although our model is not trained on extremely unbalanced data, we think that our proposed method could be practically applied on the online rumor detection task. Specifically, the GAN-based model could be re-trained periodically with the emerging data, and predict the class for unseen claims. To enhance the learning for the non-trivial rumor indicative patterns, the training data could also be artificially balanced by using various sampling techniques, such as down-sampling the major classes and/or over-sampling the minor class.

GAN-based models can learn more robust features particularly for rumor detection. The Existing text-based rumor detection systems are vulnerable to adversarial campaign. Our proposed GAN-style detection method is targeted to enhance the representation learning by training on the enriched campaign-like posts representations to make the detector stronger. Intuitively, the usually strong discriminating patterns like high-frequency words are counter-balanced via the artificially promoted campaigns by the algorithm, which allows those hidden but useful patterns to become

more salient so that the discriminator could thus strive to capture them more easily. Note that our model also contains a special treatment to avoid the useful features in the original data from being seriously weakened by the generated content (see Section 5.2). Thus, the model can learn more robust feature representations for rumor detection. Basically, our method models rumor detection problem as a process of better countering information campaign by promoting the campaign at the first place, and such mechanism is not typically viable in most other NLP applications.

7 Conclusion and Future Work

We propose a novel GAN-style model that can generate and exploit the effect of information campaigns for better rumor detection. Our neural-network-based generators create training examples to confuse rumor discriminator so that the discriminator is forced to learn more powerful features from the augmented training data. Furthermore, we propose to strengthen the generators to make more real-like voices based on transformer networks. Results on three real-world datasets show that: 1) our method is more effective and robust compared with state-of-the-art baselines; and 2) our extended models with transformer-based generators make further improvements over the original GAN-style models.

For future work, we will explore two issues: 1) we only enhance the sequential rumor detection models in this paper. But more complex propagation structures have shown usefulness for rumor detection [21], [22], [23]. We will use GAN to generate structured data to boost rumor detection. 2) Besides the textual information of the relevant posts, we will incorporate more information types (e.g., user profiles, post time, etc.) for improving our GAN-style models.

Acknowledgment

Jing Ma was supported by HKBU direct grant (Ref. AIS 21-22/02). Wei Gao was supported by the Singapore Ministry of Education (MOE) Academic Reserach Fund (AcRF) Tier 1 grant. Yang Yang was supported by the National Natural Science Foundation of China (U20B2063) and the Sichuan Science and Technology Program (2020YFS0057).

References

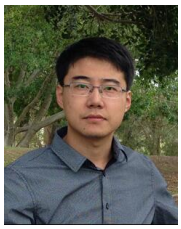
[1] G. Allport and L. Postman, *The psychology of rumor*. Russell & Russell, 1965.
[2] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on twitter," in *Proceedings of WWW*, 2011, pp. 675–684.
[3] X. Liu, A. Nourbakhsh, Q. Li, R. Fang, and S. Shah, "Real-time rumor debunking on twitter," in *Proceedings of the 24th ACM International on CIKM*, 2015, pp. 1867–1870.
[4] J. Ma, W. Gao, Z. Wei, Y. Lu, and K.-F. Wong, "Detect rumors using time series of social context information on microblogging websites," in *Proceedings of CIKM*, 2015, pp. 1751–1754.
[5] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in *IEEE 13th International Conference on Data Mining*, 2013, pp. 1103–1108.
[6] S. Kwon, M. Cha, and K. Jung, "Rumor detection over varying time windows," *PLOS ONE*, vol. 12, no. 1, p. e0168344, 2017.
[7] 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 *Proceedings of IJCAI*, 2016, pp. 3818–3824.
[8] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "A convolutional approach for misinformation identification," in *Proceedings of IJCAI*, 2017, pp. 3901–3907.
[9] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors on twitter by promoting information campaigns with generative adversarial learning," in *The World Wide Web Conference*, 2019, pp. 3049–3055.
[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, 2014, pp. 2672–2680.
[11] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems*, 2015, pp. 1486–1494.
[12] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on twitter," in *Proceedings of WWW*, 2011, pp. 675–684.
[13] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei, "Rumor has it: Identifying misinformation in microblogs," in *Proceedings of EMNLP*, 2011, pp. 1589–1599.
[14] F. Yang, Y. Liu, X. Yu, and M. Yang, "Automatic detection of rumor on sina weibo," in *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, 2012.
[15] A. Zubiaga, M. Liakata, and R. Procter, "Learning reporting dynamics during breaking news for rumour detection in social media," *arXiv preprint arXiv:1610.07363*, 2016.
[16] Z. Zhao, P. Resnick, and Q. Mei, "Enquiring minds: Early detection of rumors in social media from enquiry posts," in *Proceedings of the 24th International Conference on WWW*, 2015, pp. 1395–1405.
[17] A. Friggeri, L. A. Adamic, D. Eckles, and J. Cheng, "Rumor cascades," in *Proceedings of ICWSM*, 2014.
[18] A. Hannak, D. Margolin, B. Keegan, and I. Weber, "Get back! you don't know me like that: The social mediation of fact checking interventions in twitter conversations." in *ICWSM*, 2014.
[19] J. Ma, W. Gao, and K.-F. Wong, "Detect rumor and stance jointly by neural multi-task learning," in *Proceedings of WWW Companion*, 2018, pp. 585–593.
[20] S. A. Alkhodair, S. H. Ding, B. C. Fung, and J. Liu, "Detecting breaking news rumors of emerging topics in social media," *Information Processing & Management*, vol. 57, no. 2, 2020.
[21] K. Wu, S. Yang, and K. Q. Zhu, "False rumors detection on sina weibo by propagation structures," in *Proceedings of 31st International Conference on Data Engineering*, 2015, pp. 651–662.
[22] J. Ma, W. Gao, and K.-F. Wong, "Detect rumors in microblog posts using propagation structure via kernel learning," in *Proceedings of ACL*, 2017, pp. 708–717.
[23] J. Ma, W. Gao, and K. F. Wong, "Rumor detection on twitter with tree-structured recursive neural networks," in *Proceedings of ACL*, 2018, pp. 1980–1989.
[24] J. Ma, W. Gao, S. Joty, and K.-F. Wong, "An attention-based rumor detection model with tree-structured recursive neural networks," *TIST*, vol. 11, no. 4, pp. 1–28, 2020.
[25] L. M. S. Khoo, H. L. Chieu, Z. Qian, and J. Jiang, "Interpretable rumor detection in microblogs by attending to user interactions," in *Proceedings of AAAI*, 2020, pp. 8783–8790.
[26] X. Yang, Y. Lyu, T. Tian, Y. Liu, Y. Liu, and X. Zhang, "Rumor detection on social media with graph structured adversarial learning," in *Proceedings of IJCAI*, 2020, pp. 1417–1423.
[27] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
[28] X. Chen, Y. Sun, B. Athiwaratkun, C. Cardie, and K. Weinberger, "Adversarial deep averaging networks for cross-lingual sentiment classification," *arXiv preprint arXiv:1606.01614*, 2016.
[29] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proceedings of NIPS*, 2016, pp. 2234–2242.
[30] D. Kang, T. Khot, A. Sabharwal, and E. Hovy, "Adventure: Adversarial training for textual entailment with knowledge-guided examples," in *Proceedings of ACL*, 2018.
[31] R. Jia and P. Liang, "Adversarial examples for evaluating reading comprehension systems," in *Proceedings of EMNLP*, 2017, pp. 2021–2031.
[32] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," in *Proceedings of EMNLP*, 2017, pp. 2157–2169.
[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of NIPS*, 2017, pp. 5998–6008.

[34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT*, 2019, pp. 4171–4186.
[35] C. Tao, S. Gao, M. Shang, W. Wu, D. Zhao, and R. Yan, "Get the point of my utterance! learning towards effective responses with multi-head attention mechanism," in *Proceedings of IJCAI*, 2018, pp. 4418–4424.
[36] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," in *Proceedings of EMNLP*, 2016, pp. 551–561.
[37] E. Strubell, P. Verga, D. Andor, D. Weiss, and A. McCallum, "Linguistically-informed self-attention for semantic role labeling," in *Proceedings of EMNLP*, 2018, pp. 5027–5038.
[38] 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 KDD*, 2018, pp. 849–857.
[39] F. Qian, C. Gong, K. Sharma, and Y. Liu, "Neural user response generator: fake news detection with collective user intelligence," in *Proceedings of IJCAI*, 2018, pp. 3834–3840.
[40] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, 2014, pp. 3104–3112.
[41] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of EMNLP*, 2015, pp. 1412–1421.
[42] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.
[43] A. Bordes, L. Bottou, and P. Gallinari, "Sgd-qn: Careful quasi-newton stochastic gradient descent," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1737–1754, 2009.
[44] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.
[45] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.
[46] A. Zubiaga, M. Liakata, and R. Procter, "Exploiting context for rumour detection in social media," in *International Conference on Social Informatics*. Springer, 2017, pp. 109–123.
[47] A. Zubiaga, M. Liakata, R. Procter, G. W. S. Hoi, and P. Tolmie, "Analysing how people orient to and spread rumours in social media by looking at conversational threads," *PloS one*, vol. 11, no. 3, p. e0150989, 2016.

Jing Ma received the PhD degree from The Chinese University of Hong Kong in 2020. She is currently an Assistant Professor at the department of Computer Science, Hong Kong Baptist University. Her current research interests include Natural Language Processing, Information Verification, and Social Media Analytics. She has been serving on the program committee of several international conferences, including: IJCAI, AAAI, WWW, CIKM, ACL and EMNLP.

Jun Li received the bachelor's degree and master's degree from Chongqing University in 2013 and 2016. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China.His current research interests include Natural Language Processing and Social Media Analytics.

Wei Gao is currently an Assistant Professor at Singapore Management University. Previously, he was a senior lecturer at Victoria University of Wellington, a Scientist at Qatar Computing Research Institute, and a Research Assistant Professor in the Chinese University of Hong Kong. His research interests intersect natural language processing, information retrieval, artificial intelligence and social computing. He broadly serves in the program committees of top conferences and reviews for the leading journals in his relevant field. He is currently an Associate Editor of ACM Transactions on ALLIP and a member of standing review committees of TACL and CL.



Yang Yang is currently Professor from University of Electronic Science and Technology of China. He was a Research Fellow in National University of Singapore during 2012-2014. He was conferred his Ph.D. Degree (2012) from The University of Queensland. He obtained Master Degree (2009) and Bachelor Degree (2006) from Peking University and Jilin University. His research interests include multimedia content analysis, computer vision and social media analytics.

Kam-Fai Wong obtained his PhD from Edinburgh University in 1987. He is Professor in the Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong. His research interest include Chinese computing, database and information retrieval. He is a member of the ACM, ACL, and Senior Member of IEEE. He is the founding Editor-In-Chief of TALIP, and serves as associate editor of International Journal on CL. He is the Chair of Conference: Co-Chair of NDBC2016, BigComp2016, NLPCC2015 and IJCNLP2011; the Finance Chair SIGMOD2007; and the PC Co-chair of IJCNLP2006.