Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 6094–6105
Marseille, 11–16 May 2020

RP-DNN: A Tweet level propagation context based deep neural networks for early rumor detection in Social Media

Jie Gao, Sooji Han, Xingyi Song, Fabio Ciravegna
Regent Court, 211 Portobello, Sheffield, UK, S1 4DP
{j.gao, sooji.han, x.song, f.ciravegna}@sheffield.ac.uk

Abstract

Early rumor detection (ERD) on social media platform is very challenging when limited, incomplete and noisy information is available. Most of the existing methods have largely worked on event-level detection that requires the collection of posts relevant to a specific event and relied only on user-generated content. They are not suitable for detecting rumor sources in the very early stages, before an event unfolds and becomes widespread. In this paper, we address the task of ERD at the message level. We present a novel hybrid neural network architecture, which combines a task-specific character-based bidirectional language model and stacked Long Short-Term Memory (LSTM) networks to represent textual contents and social-temporal contexts of input source tweets, for modelling propagation patterns of rumors in the early stages of their development. We apply multi-layered attention models to jointly learn attentive context embeddings over multiple context inputs. Our experiments employ a stringent leave-one-out cross-validation (LOO-CV) evaluation set-up on seven publicly available real-life rumor event data sets. Our models achieve state-of-the-art(SoA) performance for detecting unseen rumors on large augmented data which covers more than 12 events and 2,967 rumors. An ablation study is conducted to understand the relative contribution of each component of our proposed model.

Keywords: Early Rumor Detection, Social Media, Recurrent Neural Network, Attention Mechanism, Context Modeling

1. Introduction

Research on social media rumors has become increasingly popular to understand the emergence and development of rumor events. An automatic and efficient approach for the early identification of rumors is vitally necessary in order to limit their spreading and minimize their effects.

A typical rumor resolution process can include four sub-tasks: rumor detection, tracking, stance classification, and verification (Zubiaga et al., 2018). Rumor detection which aims to identify whether a claim is a rumor or non-rumor is a fundamental task for rumor resolution. Once a rumor is identified, it becomes possible to track its evolution over time, identify its sources, perform stance detection, and finally determine the its veracity (Zubiaga et al., 2018) (Kochkina et al., 2018). Recent research on online rumors has largely focused on the later stages of the process, that is, stance classification and verification. Although these are crucial for rumor resolution, they cannot be performed until rumors are identified. Several studies skip this preliminary task, either leaving the development of approaches for them for future work or assuming that rumors and their associated posts are manual inputs. In this work, we highlight the importance of developing automated ERD systems for the success of the entire rumor resolution process.

We propose a hybrid and context-aware deep neural network framework for *tweet-level* ERD, which is capable of learning not only textual contents of rumors, but more importantly social-temporal contexts of their diffusion. A large body of SoA research on rumor detection (Lukasik et al., 2015; Chen et al., 2018; Zhou et al., 2019) only leverages language modeling at the word level for contents of source tweets and contexts (typically replies). In contrast, we pay more attention to modeling at social context level. Social contextual information typically refers to conversational threads of source tweets such as replies and retweets in the case of Twitter. Conversational threads provide time series information that how rumor-mongering changes people's opinions and how social media allows self-correction. Some research uncovers two competing rules including majority preference and minority avoidance that affect the evolution of public opinion through information exchange (Wang et al., 2017a). Therefore, conversational threads offer valuable insights about rumor propagation at the single tweet level before events become widespread and obtain far-reaching impact.

Twitter metadata provides rich explicit and implicit cues related to replies and retweets (e.g.,author information, decay of interest, and chain of replies) which can provide useful complementary signals for early diffusion and have the potential advantage of platform, domain and language portability. Different from most existing work which is exclusively based on textual contents, we argue that a good model for temporal sequence learning can benefit from multiple inputs. Multi-modal temporal data can offer different representations of the same phenomenon. In the case of content and metadata in conversational threads, they are correlated and share high-level semantics (Kıcıman, 2010). Motivated by this observation, our method aims to extend a model based on rumor source content (SC) with social context information. A SoA context-aware Neural Language Model (NLM) fine-tuned specifically for the task of rumor detection is employed to encode contents. Social contexts are modeled as the joint representation of conversational contents and metadata through a Recurrent Neural Network (RNN) architecture. We leverage two types of complementary contextual information which are strongly correlated with source tweet contents. Specifically, we utilize social context content (CC) to provide insights about how public opinion evolves in early stages and social context metadata (CM) to provide auxiliary information on how rumors spread and how people react to rumors.

The main contributions of this work can be summarized as

follows:
(1) We propose a hybrid deep learning architecture for rumor detection at the *individual tweet level*, while the majority of recent work focuses on *event-level* classification. It advances SoA performance on *tweet-level* ERD.
(2) We exploit a *context-aware* model that learns a unified and noise-resilient rumor representation from multiple correlated context inputs including SC, CC and CM beyond the word-level modeling via a rumor task-specific neural language model and multi-layered temporal attention mechanisms.
(3) A large, augmented rumor data set recently released (Han et al., 2019) is employed to train our proposed model. Extensive experiments based on an ablation study and *LOO-CV* are conducted to examine its effectiveness and generalizability. Our model outperforms SoA models in *tweet-level* rumor detection and achieves comparable performance with SoA *event-level* rumor detection models.

2. Related Work

There are two different objectives in most recent techniques proposed to date, including 1) **event-level rumor detection**: its purpose is to classify the target event into rumor and non-rumor. It involves story or event detection and tracking as well as grouping retweets or similar tweets in clusters during pre-processing (Chen et al., 2018; Kwon et al., 2017; Ma et al., 2016; Guo et al., 2018; Nguyen et al., 2017; Jin et al., 2017b; Wang et al., 2018). 2) **tweet-level detection**: in contrast to the event-level detection, it aims to detect individual rumor-bearing source tweets before events unfold (Zubiaga et al., 2016). This paper focuses on tweet-level detection. This is more challenging work than the event-level detection because individual tweets are short, noisy, and of divergent topics due to intrinsic properties of social media data. Thus, modeling tweet-level ERD with limited context is still considered as open issue (Zubiaga et al., 2018).
Event-level rumor detection (Yu et al., 2017) proposes a CNN-based misinformation detection architecture which allows CNNs to learn representations of contents of input tweets related to an event. (Ma et al., 2016) proposes various models based RNNs which learn tweet content representations based on tf-idf. (Ruchansky et al., 2017) proposes a framework which jointly learns temporal representations and user features of input posts. (Ma et al., 2018a) proposes a GRU-based, multi-task learning architecture which unifies both stances and rumor detection. (Chen et al., 2018) is one of early work that uses RNNs and attention mechanism to model deep representation of aggregated tweet content of rumor event. (Guo et al., 2018) exploits content representations and hand-crafted social contexts features with attention-based bidirectional LSTM networks.
Message-level rumor detection (Zubiaga et al., 2017) proposes a conditional random fields-based model that exploits a combination of context content and metadata features to learn sequential dynamics of rumor diffusion at the tweet level. (Ma et al., 2018b) proposes recursive neural networks models which take a tree structure of each input source tweet as input. Tree structures represent relations between source tweets and their contexts (i.e., replies and retweets). (Liu and Wu, 2018) proposes a hybrid of CNNs and RNNs which is capable of learning rumor propagation based on features of users who have participated in rumor spreading. (Jin et al., 2017a) proposes a multi-modal model comprising CNN and LSTM with attention mechanism. It jointly learns representations of rumour textual contents and social contexts. The joint representations are fused with images embedded in tweets encoded using CNNs. A recent trend is to exploit multi-task learning frameworks for rumor detection and other rumor resolution sub-tasks (Kochkina et al., 2018) (Veyseh et al., 2019; Li et al., 2019). The majority of such work focuses on leveraging tweet content representation and the conversational structure of their context (e.g., replies). (Kochkina et al., 2018) decomposes conversation threads into several branches according to Twitter mentions (i.e., @username) which allows the application of majority voting for per-thread prediction. (Veyseh et al., 2019) examines the effectiveness of recent NLMs in content embedding. (Li et al., 2019) incorporates user-level information as an additional signal of credibility. (Geng et al., 2019) incorporate the sentiment of replies into their GRU model and applies self-attention to source tweet content. (Han et al., 2019) modified the RNN-based multi-task learning model originally proposed by (Kochkina et al., 2018). The authors evaluate the proposed model using their augmented data generated via weak supervision.
In this paper, we identify several limitations of existing work on tweet-level rumor detection. The majority of SoA methods are limited to contents of source posts and/or those of their contexts and rely on hand-crafted features for both content and propagation context. Our work avoids any sophisticated feature engineering on content and only adopts a limited number of generic features commonly used to encode context metadata. In addition, prevalent word-level attention mechanism is not applied in our model. This helps us to focus on examining the effectiveness of our propagation context-based model and task-specific language model. Furthermore, data scarcity is a known limitation in the field of ERD. Most studies have evaluated their methods on small or proprietary data sets with a conventional approach for splitting data into train and test sets. To our best knowledge, it is the first work presenting an extensive experimental comparison with both LOO-CV and k-fold CV procedures to provide an almost unbiased estimate of the generalizability of a model to unseen events and in realistic scenarios.

3. Methodology

3.1. Problem Statement

Rumors are commonly considered as statements presenting facts that lack substantiation. Therefore, candidate rumor tweets should be factual or informative. In our task, a potential rumor is presented as a tweet which reports an update associated with a newsworthy event, but is deemed unsubstantiated at the time of release. Individual social media posts can be very short in nature, containing very limited context with variable time series lengths. This is a typical characteristic on Twitter. A rumor claim in the very early stages of event evolution is usually from a candidate source tweet x_i at timestamp t_i, which can be considered as a source of a potential rumor event. In this paper, we focus on conversational content and associated metadata which are considered as two separate but correlated sequential sub-events.

A set of candidate source tweets is denoted by $X = \{x_1, ..., x_n\}$ which contains i candidate tweets, where each candidate tweet $x_i = \{[CC_i, CM_i], t_i\}, x_i \in X$ consists of two correlated observations (reactions) CC_i and CM_i over time series t_i. Let j be the length of conversational threads (i.e., the number of replies) of each input source tweet. $CC_i = \{cc_{i,0}, cc_{i,1}, ..., cc_{i,j}\}$ is a set of temporal-ordered observations from **context content**. $CM_i = \{cm_{i,0}, cm_{i,1}, ..., cm_{i,j}\}$ is a set of temporal-ordered observations from **context metadata**. Let $y = \{0, 1\}$ be binary labels. The task is to predict the most probable tag for each candidate source tweet x_i based on source tweet content and all context sub-events CC_i and CM_i, given a time range $t_i \subseteq [0, j]$. $y_i = 1$ if x_i is a rumor, and $y_i = 0$ otherwise.

3.2. Overview of Model Architecture

The overall architecture of the proposed tweet-level *Rumor Propagation based Deep Neural Network* (RP-DNN) is shown in Figure 1. Basically, we learn a neural network model that takes source tweets x_i and corresponding contexts (CC_i and CM_i) as input and outputs predictions $\hat{y_i}$. RPDNN consists of four major parts including 1) data encoding layers, 2) stacked RNN layers, 3) stacked attention models, and 4) classification layer.

Tweet-level EDR using RP-DNN follows the four key stages: **a)** Once candidate source tweets X and associated context inputs (CC_i and CM_i) are loaded and preprocessed (see details in section 4.2.), the two types of raw context inputs will be encoded in ***data encoding layers***. These are important layers that convert source tweets and conversational context into inputs for subsequent RNN layers for contextual modeling. It consists of a content embedding layer (section 3.5.) and a metadata encoding layer (section 3.6.). The objective of the former is to convert tweets into embeddings V^i_{cc}. The latter is to use a Metadata Feature Extractor (MFE) to extract features from the corresponding metadata of the tweets that characterizes public engagement and diffusion patterns. The output of the MFE is represented as feature vectors V^i_{cm} which are normalized by applying a global mean and variance computed from training data. **b)** Subsequently, encoded context inputs will be fed into a social-temporal context representation layer consisting of ***stacked RNN layers*** and ***stacked attention models*** (illustrated in section 3.3. and 3.4. respectively). We stack multiple LSTMs together to form a stacked LSTM that takes input representations (i.e., V^i_{cc} and V^i_{cm}; outputs of the data encoding layers) arranged in chronological order. Let the number of layers be L. L-layer LSTMs ($L = 2$ in our case) are utilized to process the two types of contextual data separately. **c)** The recurrent structure models features of sequential data and then uses soft hierarchical attention models (the 1st attention layer) to produce an optimal representation. The contextual embeddings from the two recurrent layers (hidden states) output (H^i_{cm} and H^i_{cc}) are then temporally combined to form a joint representation (H^i_c). The third attention model (the 2nd attention layer) is performed on the joint hidden sequential embedding H^i_c and eventually produces a compact representation of context sequences V^i_c, followed by (masked) layer normalisation (Ba et al., 2016). **d)** Finally, we combine two embeddings of SC and context via concatenation to form the final rumor source representation in the ***classification layer***. This is the final output layer which provides the result of rumor detection. Cross-entropy loss are computed to optimize the whole network. A 3-layer fully-connected neural network with Leaky ReLu activations and softmax function takes the final representation to yield the output.

3.3. Stacked RNN layer

A natural choice is to use Recurrent Neural Network (RNN) to model rumor context. An RNN processes a sequential input in a way that resembles how humans do it. It performs an operation, $h_t = f_W(x_t, h_{t-1})$, on every candidate tweet context (x_t) of a sequence, where h_t is the hidden state a time step t and W is the weights of the network. The hidden state at each time step depends on the previous hidden state. Therefore, the order of time series-based reaction context input is important. Intuitively, this process enables RNNs to model the evolution of public opinion about each source claim and diffusion patterns of public engagement (e.g., retweets, likes) through corresponding metadata. Meanwhile, it enables to handle inputs of variable lengths.

Regarding utilizing complementary context clues and modeling context with different types of features (considered as two different sub-events), conventional approaches (Xing and Paul, 2017; Zhou et al., 2017; Jin et al., 2017a; Gu et al., 2018) simply concatenate embeddings of different data inputs or process them through a linear combination of different feature embeddings to form a single representation. This practice completely ignores the correlations and differences between different context inputs. We argue that a model should have the ability of learning weights separately from different context inputs in order to find salient parts of each context type. In addition, a model should have the ability to learn important clues across multiple context observations (as illustrated in section 3.4.).

To this end, we propose two (simultaneous) context embeddings to explore two correlated context inputs, and use two layers of forward LSTMs in order to learn more abstract features respectively. Concretely, to model the temporal evolution of public opinions, context content embeddings (V^i_{cc}) are given as input to two layers of forward LSTMs. The context output state H^i_{cc} at time t is abbreviated as $\overrightarrow{h^i_{cc,t}} = \overrightarrow{LSTM_l}(\overrightarrow{h^i_{cc,t-1}}, v^i_{cc,t}), \forall t \in [0, j]$.

Regarding diffusion patterns of public engagement, we employ shallow features extracted from explicit information in social reactions to induce a hierarchical RNN model. In contrast to previous work (Ma et al., 2015; Zubiaga et al., 2017), our RNN-based method avoids painstakingly complicated feature engineering, and instead allows RNN to learn deep, hidden behavioural, and social dynamics of underlying complex hierarchical social-temporal structure. The context output state H^i_{cm} at time t is abbreviated as

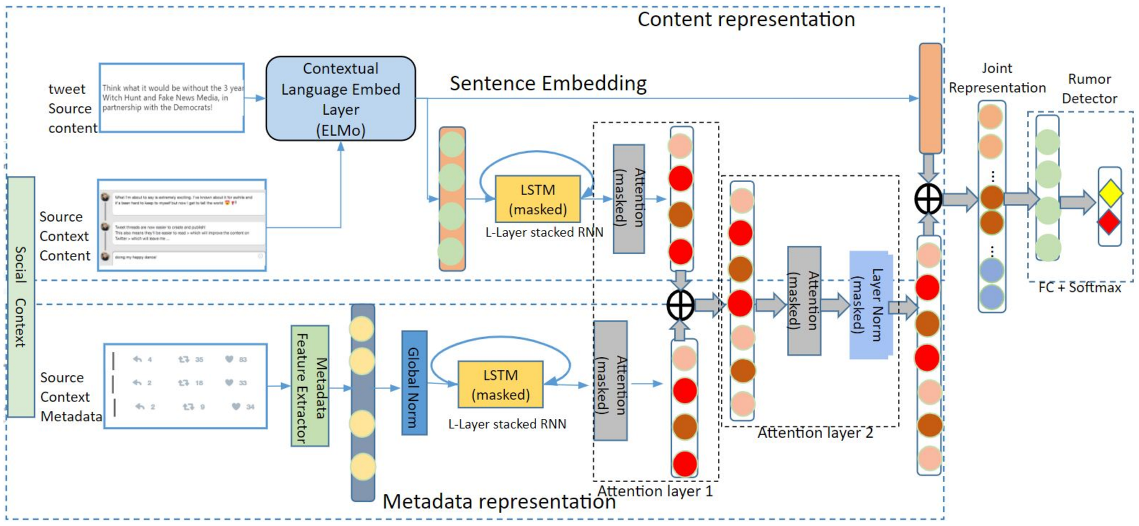

Figure 1: Overview of model architecture

$\overrightarrow{h^i_{cm,t}} = \overrightarrow{LSTM_l}(h^i_{cm,t-1}, \overrightarrow{v^i_{cm,t}}), \forall t \in [0, j].$

3.4. Stacked Soft Attentions

In order to amplify the contribution of important context elements and filter noise or unnecessary information in final representation, we introduce multiple-layer stack attention mechanism in our network. This is inspired by the performance of stacked attentions in recent advances (Dyer et al., 2015; Yang et al., 2016a). By applying attention over multiple steps, the model can focus on more salient features and this has been proved in many visual recognition challenges (Yang et al., 2016a). We explore ways to leverage attention mechanisms for context embeddings at different levels to eliminate invalid information and get more accurate contextual interaction information, thereby improving classification performance.

Specifically, we propose to calculate attention weights by providing information about all time steps for context embedding layers. It takes a context sequence of a predefined length j as input and learns a mapping from this sequence to an output sequence using attention mechanisms. We employ the idea of hierarchical attention networks (Yang et al., 2016b) and adapt the context-aware model in our networks. We here represent attention as a probabilistic distribution over temporally ordered conversational context inputs, and implement its estimation via our end-to-end rumor classification framework. The standard softmax function (Martins and Astudillo, 2016) is used to approximate a normalized probability distribution of importance on entire context. Let H_c be the recurrent hidden states of tweet context (see section 3.3.). Formally,

$$\alpha^t_c = softmax(tanh(W_h h^t_c + b_h)), \forall t \in [0, j]. \quad (1)$$

$$h^t_{c\_new} = \alpha^t_c h^t_c \quad (2)$$

W_h and b_h are the attention layer's weights, which are initialized using He initialization and optimized during training. Zero padding is used to handle variable lengths. Following the same practice adopted in the stacked RNN layer, we mask out padded values with negative infinity float following the practice of (Vaswani et al., 2017). h_{c_new} is the re-weighted context embeddings.

Rather than only computing attention weights once, attention mechanism is applied to two layers in our architecture: 1) stacked RNN layers and 2) joint representation layer. Specifically, the first attention layer contains two sub-layers of attentions on the top of CC context encoder (see section 3.5.) output H^t_{cc} and CM context encoder (see section 3.6.) output H^t_{cm} respectively (as defined in eq. 3 and 4). Two independent attention models are trained and then modify the hidden states of two separate recurrent layers. The output of two attention models are denoted as $H^t_{cc\_new}$ and $H^t_{cm\_new}$. The weighted hidden state vectors for all timesteps from two context encoders are then concatenated and provided as joint representation input for the second attention layer.

$$H^t_{cc\_new} = attention_1(H^t_{cc}) \quad (3)$$

$$H^t_{cm\_new} = attention_1(H^t_{cm}) \quad (4)$$

To determine the inference relationship between two correlated context embeddings and to verify our hypothesis, we use the attention model as a composition layer to mix the two types of sub-event inference information. Different from the first attention layer, the second attention layer aggregates all the hidden states using their relative importance via weighted sum, which is trained in the hope to capture shared semantics between content and metadata. Eventually, the proposed algorithm helps to incorporate additional auxiliary information into a unified representation of reaction and diffusion patterns to achieve outstanding performance in our context-based EDR problem. Formally,

$$h^t_c = attention_2(h^t_{cc\_new} \oplus h^t_{cm\_new}) \quad (5)$$

$$v_c = \sum_t h^t_c \quad (6)$$

where h_c^t is the joint hidden states of context and v_c is the final context vector, i.e., a sum of h_c^t for all time steps.

3.5. Tweet content encoder

A large body of work (Zubiaga et al., 2018) has previously proposed and demonstrated the effectiveness and advantage of textual contents in rumor detection. The user-generated content has been proved to be useful for providing effective signals for identifying emerging rumors. For instance, credibility-related terms (e.g., "reportedly", "I hear that", etc.) can effectively indicate the uncertainty of a candidate tweet (Zubiaga et al., 2017). For rumor tweets that do not have sufficient signals, social context content can provide information about how people react to them, which has been exploited extensively to identify rumors and their veracity. (Maddock et al., 2015; Zubiaga et al., 2018).
In our framework, tweet content embeddings are obtained via ELMo (Peters et al., 2018), a SoA context-aware neural language model (NLM). We allow a NLM to learn signals for linguistic and semantic characteristics from rumor tweet content such as ambiguity and uncertainty in order to avoid using hand-crafted features. ELMo represents each individual word while taking the context of an entire corpus (e.g., a sentence and paragraph) into account. The weight of each hidden state is task-specific and can be learned from domain-specific corpora. In our architecture, tweet sentence embeddings are learned from both domain-specific and general corpora. We employ a SoA ELMo model fine-tuned specifically for the task of rumor detection on social media (Han et al., 2019). This domain-specific language model is pre-trained on an 1 billion word benchmark corpus with vocabulary of 793,471 tokens and then fine-tuned on a large credibility-focus Twitter corpus with 6,157,180 tweets with 146,340,647 tokens and 2,235,075 vocabularies. The fine-tuned model achieves low perplexity in in-domain data sets and SoA performance in the rumor detection task. Following the practice in (Perone et al., 2018; Han et al., 2019), averaging ELMo word vectors is employed to produce the final short-text embeddings, using features from all three layers of the ELMo model.

3.6. Conversational Context Metadata

The proposed architecture leverages 27 hand-crafted and generic features (described in Table 1) that can be categorized into tweet-level and user-level. Early work on rumor detection employs supervised learning techniques, and thus has extensively studied manually curated features related to contents, users, and networks to seek distinguishing features of online rumors (Qazvinian et al., 2011b; Kwon et al., 2017; Yang et al., 2012; Sun et al., 2013; Zhao et al., 2015; Zhang et al., 2015; Wu et al., 2015; Ma et al., 2015; Liu and Xu, 2016; Zubiaga et al., 2017; Hamidian and Diab, 2016). These studies have shown that those features have the potential for distinguishing rumors from non-rumors. In recent advances of deep learning architectures, few event-level detection techniques (Ruchansky et al., 2017; Kwon et al., 2017; Liu and Wu, 2018; Guo et al., 2018) have shown the merits of combining both hand-crafted metadata features and deep-learned features.

Table 1: Description of hand-crafted features.

Tweet-level features
Number of retweets
Number of favorites
Whether tweet has a question mark
Whether tweet is a duplicate of its source
Whether tweet contains URLs
Number of URLs embedded in tweet
Whether tweet has native media*
Number of words in tweet except source author's screen name
User-level features
Number of posts user has posted
Number of public lists user belongs to
Number of followers
Number of followings
Whether user has a background profile image
User reputation (i.e., followers/(followings+1))
User reputation (i.e., followers/(followings+followers+1))
Number of tweets user has liked so far (aka "user favorites")
Account age in days
Whether user is verified
User engagement (i.e., # posts / (account age+1))
Following rate (i.e., followings / (account age+1))
Favorite rate (i.e., user favorites / (account age+1))
Whether geolocation is enabled
Whether user has a description
Number of words in user description
Number of characters in user's name including white space
Whether user is source tweet's author
Response time decay (time difference between context and source tweet in mins)

* multimedia shared with the Tweet user-interface not via an external link

• **Tweet-level features** We let unsupervised NLM automatically learn syntactic and semantic representations of input tweets. Therefore, our hand-crafted features related to content mainly include features related to URLs and multimedia embedded in tweets. Twitter users often use URLs as additional references due to a length limit (Qazvinian et al., 2011a). Including them in tweets tends to encourage more people to share rumors (Tanaka et al., 2014) and increase the trustworthiness of tweets (Gupta and Kumaraguru, 2012; Castillo et al., 2011). In particular, (Friggeri et al., 2014) reports that unverified information with links to websites for validating and debunking rumors often goes viral on social media.
• **User-level features** Rumor spreaders are individuals who seek attention and reputation (Sunstein, 2010). Features related to user profiles and reactions contribute to the characterization of rumors (Liu et al., 2015). Previous studies found that rumors tend to spread from low-impact users to influencers, whereas non-rumors have the opposite tendency (Ma et al., 2017; Kwon et al., 2017). Another study reports that trustworthy sources such as mainstream media and verified users participate in rumor spreading by simply sharing rumors and maintaining neutrality (Li et al., 2016).

4. Experiments

In this section, we report the evaluation data set and methods for our proposed model and data processing methods.

4.1. Data sets

Table 2 presents the statistics of all the pre-filtered event data sets used in our experiment. They are obtained from three public data sets. "Avg. tdiff" stands for the average time length of context (conversational threads) in each event data set in minutes.
1. Aug-rnr (Han et al., 2019): This is an augmented version of the *PHEME (6392078)*. It contains rumor and non-

Table 2: Statistics of 12 events data sets.

Event			**Replies**					
	# of rumors	**# of non-rumors**	**Total**	**Avg.**	**Min**	**Max**	**Mdn**	**Avg. tdiff**
charlie	382	1,356	42,081	24	6	341	19	8.6
ferguson	266	746	26,565	26	6	288	18	47.3
german	132	122	4,163	16	6	109	14	12.8
sydney	480	784	26,435	21	6	341	17	7.1
ottawa	361	539	16,034	18	6	208	13	440.6
boston	75	584	23,210	35	6	207	20	8.1
ebola	13	0	208	16	6	26	15	42.6
gurlitt	1	1	23	12	7	16	12	174.1
prince	43	0	452	11	6	21	10	4.7
putin	22	9	379	12	6	25	10	21.1
twitter15	782	323	47,324	43	6	458	28	2.2
twitter16	410	191	27,732	46	6	458	29	16.6
Total	2,967	4,655	214,606					

rumor source tweets and their contexts associated with six real-world breaking news events. Source tweets are labeled with weak supervision. The augmented data set expands original one by 200% of source tweets and 100% of social context data. The temporal filtered version 2.0 data[1] is adopted in our experiments to examine our models' performance in the context-based ERD task. We only use replies (i.e., context data) posted within 7 days after corresponding source tweets were posted. Retweets are excluded [2].

2. Twitter15/16 (Ma et al., 2017): These two data sets consist of rumor and non-rumor source tweets and their context. The context of each source tweet is provided in the form of propagation trees. Source tweets are manually annotated with one of the following four categories: non-rumor, false rumor, true rumor and unverified rumor. As we restrict the experiment set-up to binary classification, all but "non-rumor" class are aggregated into "rumor" class. We collect context data by following the practice introduced in (Han et al., 2019).

3. PHEME (6392078; (Kochkina et al., 2018)): This consists of manually labeled rumor and non-rumor source tweets and their replies for 9 breaking news events. It is used to generate test sets during evaluation.

4.2. Data Preprocessing

In this task, a candidate source tweet has to satisfy the following constraints: (1) *informativeness*: the length of its content (i.e., the number of tokens) should be greater than a minimum value. Tweets that lack enough textual information are generally unremarkable and add noise to data (Ifrim et al., 2014). (2) *popularity*: its context size (i.e., the number of replies to it) should be greater than a minimum value. This pre-filtering allows us to examine the focus of this paper regarding conversational context. Therefore, each input x_i (i.e., a candidate tweet) is set to satisfy both minimum content length ($= 4$) and minimum context length($= 5$). All tweets are lowercased, tokenized and deaccented.

[1]`https://zenodo.org/record/3269768`

[2]Our preliminary results shows that retweets metadata is very noisy. Simply adding retweets into context causes underfitting and poor performance.

Table 3: Statistics of the balanced data sets for LOO-CV.

LOO Event	**Training**	**Holdout**	**Test**
charlie	4,674	496	680
ferguson	4,818	584	466
german	5,144	526	212
sydney	4,474	500	836
ottawa	4,676	536	578
twitter15	3,924	446	646
twitter16	4,600	514	382

4.3. Model Implementations

Models were implemented[3] using Python 3.6, Allennlp (0.8.2) framework(Gardner et al., 2018), and Pytorch 1.2.0. All models were trained on one Tesla P100 SXM2 GPU node with maximum 16GiB RAM. More details of model settings are given in appendix 8..

4.4. Settings and Baselines

Two following evaluation procedures are employed to evaluate our models. Four performance metrics are adopted in our experiments including Accuracy (Acc.), precision (P), recall (R), and F1-measure (F_1). P, R and F_1 are computed with respect to positive class (i.e., rumor). Overall performance is an average over all CV folds.

LOO-CV The mainstream rumor detection methods (Ma et al., 2016; Liu and Wu, 2018; Chen et al., 2018; Ma et al., 2018b; Zhou et al., 2019; Tarnpradab and Hua, 2019) adopt conventional K-fold Cross Validation (CV) procedures with various different split ratios to estimate their models' performance. This practice allows similar distributions between train and test sets, and usually leads to good performance. However, the simple train/test split seems weak when a model is required to generalize beyond the distribution sampled from the same rumor event data. To this end, we adopt *Leave one (event) out cross validation (LOO-CV)* as an approximate evaluation of our proposed models in realistic scenarios.

Our LOO-CV data is presented in Table 3. 12 real-world rumor event data sets in total are used to generate balanced training, hold-out and test data. Two types of samples (i.e., rumor and non-rumor) are randomly shuffled in each data set. Training and hold-out sets contain augmented data sets from *Aug-rnr*, generated from 11 (out of 12) events with a split ratio 9:1. 7 manually labeled event data sets from *PHEME (6392078)* and *Twitter15/16* are selected as test sets, thus it is 7-fold LOO-CV.

K-fold CV We also evaluate our models via 5-fold cross validation following the common practice in this field in order to provide a comparative evaluation with more SoA methods. Stratified k-fold CV is employed to ensure that the percentage of samples for each class is preserved in each returned stratified fold. The split ratio for three data sets is 18:1:1, which results in 4,382 source tweets in the training set and 246 in hold/test set per fold.

Baselines Our models (see Section 4.5.) are evaluated with the following SoA models that are comparable and utilize conversational threads.

[3]The source code is available at `https://github.com/jerrygaoLondon/RPDNN`

• **(Zubiaga et al., 2017):** LOO-CV results for tweet-level classification on positive class (i.e., rumor) are given on 5 PHEME event sets.

• **(Zhou et al., 2019):** Overall results of event-level ERD for two classes with a 3:1 train/test split ratio are provided for the 5 PHEME event sets .

• **(Han et al., 2019):** LOO-CV results for tweet-level ERD on the 5 PHEME event sets are provided based on a train/test split ratio of 3:1.

• **(Ma et al., 2018b):** 5-fold CV results for tweet-level ERD for four classes on the Twitter 15/16 are available with a 3:1 train/test split ratio.

• **(Liu and Wu, 2018):** 3-fold CV results for event-level ERD for two classes are reported on the Twitter 15/16 with a 3:1 train/test split ratio.

4.5. Ablation study

A set of exploratory experiments is conducted to study the relative contribution of each component in our message-level ERD model.

• **RPDNN**: This is our full model setting that we will compare with baseline methods.

• **RPDNN-cxt**: Only source contents are used.

• **RPDNN-SC**: Only social contexts are used.

• **RPDNN-CC**: This is the full model excluding context contents.

• **RPDNN-CM**: This is the full model excluding context metadata.

• **RPDNN-Att**: This is the full model excluding attention mechanisms. The last hidden state of LSTM output is used for classification with this setting.

• **RPDNN-SC-CC**: Only context metadata are used.

• **RPDNN-SC-CM**: Only context contents are used.

Table 4: Comparison of overall CV results.

Methods	P	R	F1	Acc.
RPDNN	0.768	**0.876**	**0.817**	0.803
RPDNN-cxt	**0.785**	0.844	0.811	**0.804**
RPDNN-SC	0.730	0.839	0.780	0.762
RPDNN-CC	0.762	0.846	0.801	0.788
RPDNN-CM	0.754	0.868	0.805	0.789
RPDNN-Att	0.766	0.847	0.803	0.792
RPDNN-SC-CM	0.779	0.733	0.754	0.762
RPDNN-SC-CC	0.624	0.597	0.609	0.617
(Zhou et al., 2019)	**0.843***	0.735*	0.785*	**0.858***
(Liu and Wu, 2018)	–	–	**0.843**	0.853
(Ma et al., 2018a)	–	–	0.753	0.730

*evaluation metrics are computed over all classes.

Table 5: Comparison of overall LOO-CV results.

Methods	P	R	F1	Acc.
RPDNN	**0.648**	0.834	**0.727**	**0.684**
RPDNN-cxt	0.626	0.838	0.715	0.667
RPDNN-SC	0.621	0.796	0.694	0.648
RPDNN-CC	0.631	0.800	0.705	0.654
RPDNN-CM	0.625	**0.862**	0.723	0.669
RPDNN-Att	0.643	0.814	0.717	0.679
RPDNN-SC-CM	0.59	0.862	0.697	0.625
RPDNN-SC-CC	0.568	0.519	0.514	0.544
(Han et al., 2019)	**0.716**	**0.614**	**0.656**	**0.685**
(Zubiaga et al., 2017)	0.692	0.559	0.601	–

5. Results and Discussion

5.1. Classification Performance

As shown in Table 4 and 5, our proposed model yields SoA performance with larger test data comparable to all the baseline models under two different evaluation techniques while our architecture provides a more abstract context representation and does not specially model many aspects of factuality (e.g., stance, word-level context, sentiment, follower/following relationship, etc.). The full model (RPDNN) achieved an average F_1 score of 0.817 in 5-fold CV and that of 0.727 in 7-fold LOO-CV. The result of more stricter LOO-CV shows 7% improvement over the best comparable SoA method. Details of LOO-CV results are presented in appendix 8. In brief, we observe that performance varies slightly for different LOO events. The variance of cross-event performance is 0.0033 in F_1 and 0.0055 in $Acc.$, which could be attributed to structural issues of different LOO event context rather than actual model capabilities.

Ablation study observation The ablation study of the internal baseline models of shows that **1) source content:** the content of candidate source tweets can be considered as the most important and influential factor in ERD. This observation is consistent with a large body of previous work that exploits source contents alone to measure the credibility of rumors. The source content only model ("RPDNN-cxt") achieved performance comparable to the full model (only 1% difference with two metrics). The experiment results show that the adoption of the rumor task-specific ELMo model proves to be effective for short-text content embeddings with limited context by capturing significant contextualized representations of rumor-bearing tweets' content. The ELMo embeddings make the most contribution and improve the overall results, which is further supported by "RPDNN-SC-CM" setting. **2) conversational context:** the context of source tweets can provide additional and effective information to detect rumors. The context-only model "RPDNN-SC" achieved comparable performance to the full model (0.780 F_1 in CV and 0.694 F_1 in LOO-CV respectively). It is worth noting that our context content only model ("RPDNN-SC-CM") also achieved SoA performance based on two metrics (0.754 F_1 in CV and 0.697 F_1 in LOO-CV). The results indicate that modeling the evolution of public opinion and self-correction mechanism in tweet context is an important and effective approach to ERD. In addition, the metadata only model ("RPDNN-SC-CC") achieved reasonable performance (0.609F_1 in CV and 0.514F_1 in LOO-CV respectively) and incorporating metadata helps to improve precision of full model by 2.3% with LOO-CV (as observed in "RPDNN-CM"). This verifies our assumption that the context metadata of rumor source tweets is useful in capturing relevant characteristics of rumor diffusion in early stages. Our observation from the comparative results suggests that although context metadata is more noisy than context content, it can provide effective complementary evidence in the early stages of rumor diffusion with respect to the identification of weak signals. Further experiments can be conducted to investigate its usefulness in cross-platform (i.e., other social media platforms) and cross-language prediction in terms of exploiting a pre-trained metadata model with transfer learning techniques. By comparing "RPDNN-CC" and "RPDNN-CM" to the full model, the final unified model improves F_1 performance by around 1-2%, which can be attributed to its modeling of higher-order feature interactions of two correlated contexts. **3) context-aware attention mechanisms:** the benefits of incorporating stacked attention mechanisms into a context model are further justified in our experiments by comparison of performance between the full model and attention excluded model ("RPDNN-Att"). Our context-aware attention mechanism can slightly improve both recall and precision, and overall performance with attention achieves a slight improvement in F-measure under the two evaluation settings by 1.4% and 1% respectively. Empirical observation in our data indicates that the stacked attention models can reweigh contexts according to their relevance and significance layer by layer. Due to the recurrent structure, the hidden vector close to the end is more informative than its beginning. Thus, for small context, the performance of the attention-based full model is similar to that of the standard LSTM model (i.e., "RPDNN-Att"). Few representative context samples from the test set with 3 layers of attention weights can be found in Figure 10 in Appendix.

5.2. Training Loss and Performance

Based on the experiments, we set the number of epochs to 10 in order to avoid overfitting. Figure 9 presents training loss and accuracy curve with 10 epochs over time during the training of "RPDNN" models in 7-fold LOO-CV.

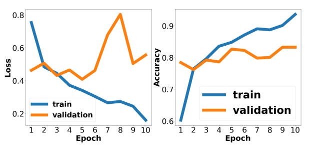

Figure 2: charliehebdo

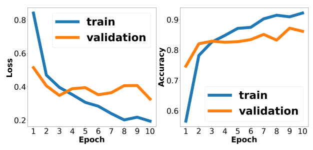

Figure 3: fergusonunrest

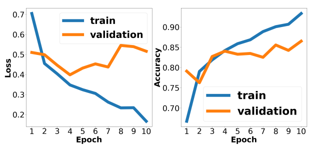

Figure 4: germanwings

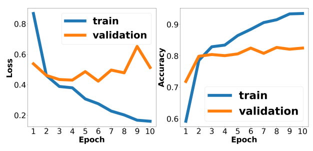

Figure 5: ottawashooting

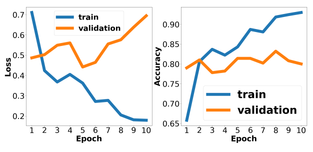

Figure 6: sydneysiege

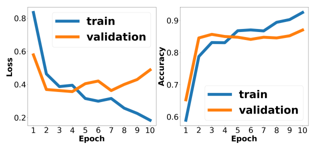

Figure 7: twitter15

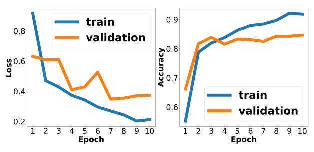

Figure 8: twitter16

Figure 9: Loss and accuracy curves for 7 folds in LOOCV.

The figures on 7 LOO models show steady decreases in training loss within the first 5 epochs and the tendency of overfitting after the 10th epoch. In comparison, we see a constant increase of accuracy in both training and validation sets for all the LOO models. The results show that the "sydneysiege" LOO set is the most difficult one to fit. Its divergence in loss can be observed in the very early stage since the 5th epoch and validation accuracy starts to drop after the 10th epoch. The average training time of full models on LOO-CV data is around 28 hours with GPU.

6. Conclusion

In this paper, we addressed the task of message-level ERD in early development stages of social media rumors where limited information is available. A novel hybrid, context-aware neural network architecture was proposed to learn a unified representation of tweet contents and propagation contexts, enabling the modeling of the evolution of public opinion and the early stages of rumor diffusion. We performed comparative evaluations with two CV techniques and larger test sets from real-life events. The results showed that the proposed model achieves SoA performance. Experimental results showed the advantage of utilizing two types of correlated temporal context inputs from conversational contents and the metadata of tweets in learning an optimal sequential model by improving its effectiveness and generalizability in unseen rumor events. An ablation study proved the positive effect of incorporating a task-specific neural language model and a multi-layered attention model in representation learning in terms of improving resistance to overfitting and noise.
There are several directions for future research. One is to consider the incorporation of social network structure. A potential benefit of modeling retweet chains via follower-following relationship can be studied. In our current work, we find no way to obtain this context data for our public retrospective data using public Twitter API. In addition, the impact of many recent neural language models (typically transformer-based models) and variants of context-aware self-attention models (e.g., multi-head self-attention mechanism in recent work) with larger context size can be examined. Furthermore, generating larger training data with weak supervision technique is promising and can be exploited to allow a deeper NN architecture. It is also interesting to investigate the transferability of a unified model across multiple social media platforms, particularly for the language-independent metadata model. The efficiency and scalability in online social networks are unknown and not examined in this paper.

7. Bibliographical References

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Castillo, C., Mendoza, M., and Poblete, B. (2011). Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM.

Chen, T., Li, X., Yin, H., and Zhang, J. (2018). Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 40–52. Springer.

Dyer, C., Ballesteros, M., Ling, W., Matthews, A., and Smith, N. A. (2015). Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

Friggeri, A., Adamic, L., Eckles, D., and Cheng, J. (2014). Rumor cascades. In *Eighth International AAAI Conference on Weblogs and Social Media*.

Gardner, M., Grus, J., Neumann, M., Tafjord, O., Dasigi, P., Liu, N., Peters, M., Schmitz, M., and Zettlemoyer, L. (2018). Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*.

Geng, Y., Lin, Z., Fu, P., and Wang, W. (2019). Rumor detection on social media: A multi-view model using self-attention mechanism. In *ICCS*.

Gu, Y., Chen, S., and Marsic, I. (2018). Deep mul timodal learning for emotion recognition in spoken language. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5079–5083. IEEE.

Guo, H., Cao, J., Zhang, Y., Guo, J., and Li, J. (2018). Rumor detection with hierarchical social attention network. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 943–951. ACM.

Gupta, A. and Kumaraguru, P. (2012). Credibility ranking of tweets during high impact events. In *Proceedings of the 1st workshop on privacy and security in online social media*, page 2. Acm.

Hamidian, S. and Diab, M. (2016). Rumor identification and belief investigation on twitter. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 3–8.

Han, S., Gao, J., and Ciravegna, F. (2019). Data augmentation for rumor detection using context-sensitive neural language model with large-scale credibility corpus. In *2019 Seventh International Conference on Learning Representations (ICLR) Learning with Limited Labeled Data (LLD)*.

Ifrim, G., Shi, B., and Brigadir, I. (2014). Event detection in twitter using aggressive filtering and hierarchical tweet clustering. In *Second Workshop on Social News on the Web (SNOW), Seoul, Korea, 8 April 2014*. ACM.

Jin, Z., Cao, J., Guo, H., Zhang, Y., and Luo, J. (2017a). Multimodal fusion with recurrent neural networks for rumor detection on microblogs. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, pages 795–816. ACM.

Jin, Z., Cao, J., Guo, H., Zhang, Y., Wang, Y., and Luo, J. (2017b). Rumor detection on twitter pertaining to the 2016 us presidential election. *arXiv preprint arXiv:1701.06250*.

Kıcıman, E. (2010). Language differences and metadata features on twitter. In *Web N-gram Workshop*, page 47.

Kwon, S., Cha, M., and Jung, K. (2017). Rumor detection over varying time windows. *PLOS ONE*, 12(1):1–19.

Li, Q., Liu, X., Fang, R., Nourbakhsh, A., and Shah, S. (2016). User behaviors in newsworthy rumors: A case study of twitter. In *Proceedings of the Tenth International Conference on Web and Social Media, Cologne, Germany, May 17-20, 2016.*, pages 627–630.

Li, Q., Zhang, Q., and Si, L. (2019). Rumor detection by exploiting user credibility information, attention and multi-task learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1173–1179, Florence, Italy, July. Association for Computational Linguistics.

Liu, Y. and Wu, Y.-F. B. (2018). Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Liu, Y. and Xu, S. (2016). Detecting rumors through modeling information propagation networks in a social media environment. *IEEE Transactions on Computational Social Systems*, 3(2):46–62.

Liu, X., Nourbakhsh, A., Li, Q., Fang, R., and Shah, S. (2015). Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM.

Lukasik, M., Cohn, T., and Bontcheva, K. (2015). Classifying tweet level judgements of rumours in social media. *arXiv preprint arXiv:1506.00468*.

Ma, J., Gao, W., Wei, Z., Lu, Y., and Wong, K.-F. (2015). Detect rumors using time series of social context information on microblogging websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1751–1754. ACM.

Ma, J., Gao, W., Mitra, P., Kwon, S., Jansen, B. J., Wong, K.-F., and Cha, M. (2016). Detecting rumors from microblogs with recurrent neural networks. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 3818–3824. AAAI Press.

Ma, J., Gao, W., and Wong, K.-F. (2017). Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 708–717, Vancouver, Canada, July. Association for Computational Linguistics.

Ma, J., Gao, W., and Wong, K.-F. (2018a). Detect rumor and stance jointly by neural multi-task learning. In *Companion of the The Web Conference 2018 on The Web Conference 2018*, pages 585–593. International World Wide Web Conferences Steering Committee.

Ma, J., Gao, W., and Wong, K.-F. (2018b). Rumor detection on twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1980–1989, Melbourne, Australia, July. Association for Computational Linguistics.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3.

Maddock, J., Starbird, K., Al-Hassani, H. J., Sandoval, D. E., Orand, M., and Mason, R. M. (2015). Characterizing online rumoring behavior using multi-dimensional signatures. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 228–241. ACM.

Martins, A. and Astudillo, R. (2016). From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, pages 1614–1623.

Nguyen, T. N., Li, C., and Niederée, C. (2017). On early-stage debunking rumors on twitter: Leveraging the wisdom of weak learners. In *Social Informatics - 9th International Conference, SocInfo 2017, Oxford, UK, September 13-15, 2017, Proceedings, Part II*, pages 141–158.

Perone, C. S., Silveira, R., and Paula, T. S. (2018). Evaluation of sentence embeddings in downstream and linguistic probing tasks. *arXiv preprint arXiv:1806.06259*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Qazvinian, V., Rosengren, E., Radev, D. R., and Mei, Q. (2011a). Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1589–1599. Association for Computational Linguistics.

Qazvinian, V., Rosengren, E., Radev, D. R., and Mei, Q. (2011b). Rumour has it: Identifying misinformation in microblogs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1589–1599. Association for Computational Linguistics.

Ruchansky, N., Seo, S., and Liu, Y. (2017). Csi: A hybrid deep model for fake news detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, pages 797–806. ACM.

Sun, S., Liu, H., He, J., and Du, X. (2013). Detecting event rumors on sina weibo automatically. In Yoshiharu Ishikawa, et al., editors, *Web Technologies and Applications*, pages 120–131, Berlin, Heidelberg. Springer Berlin Heidelberg.

Sunstein, C. (2010). *On Rumours: How Falsehoods Spread, Why We Believe Them, What Can Be Done*. Penguin Books Limited.

Tanaka, Y., Sakamoto, Y., and Honda, H. (2014). The impact of posting urls in disaster-related tweets on rumor spreading behavior. In *2014 47th Hawaii International Conference on System Sciences*, pages 520–529. IEEE.

Tarnpradab, S. and Hua, K. A. (2019). Attention based neural architecture for rumor detection with author context awareness. *CoRR*, abs/1910.01458.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Veyseh, A. P. B., Thai, M. T., Nguyen, T. H., and Dou, D. (2019). Rumor detection in social networks via deep contextual modeling. In *Proceedings of 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE.

Wang, C., Tan, Z. X., Ye, Y., Wang, L., Cheong, K. H., and Xie, N.-g. (2017a). A rumor spreading model based on information entropy. *Scientific reports*, 7(1):9615.

Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X., and Tang, X. (2017b). Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164.

Wang, Y., Ma, F., Jin, Z., Yuan, Y., Xun, G., Jha, K., Su, L., and Gao, J. (2018). 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*, pages 849–857. ACM.

Wu, K., Yang, S., and Zhu, K. Q. (2015). False rumors detection on sina weibo by propagation structures. In *2015 IEEE 31st International Conference on Data Engineering*, pages 651–662.

Xing, L. and Paul, M. J. (2017). Incorporating metadata into content-based user embeddings. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 45–49.

Yang, F., Liu, Y., Yu, X., and Yang, M. (2012). Automatic detection of rumor on sina weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, MDS '12, pages 13:1–13:7. ACM.

Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. (2016a). Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016b). Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Yu, F., Liu, Q., Wu, S., Wang, L., and Tan, T. (2017). A convolutional approach for misinformation identification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3901–3907.

Zhang, Z., Zhang, Z., and Li, H. (2015). Predictors of the authenticity of internet health rumours. *Health Information & Libraries Journal*, 32(3):195–205.

Zhao, Z., Resnick, P., and Mei, Q. (2015). Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 1395–1405. International World Wide Web Conferences Steering Committee.

Zhou, L.-k., Tang, S.-l., Xiao, J., Wu, F., and Zhuang, Y.-t. (2017). Disambiguating named entities with deep supervised learning via crowd labels. *Frontiers of Information Technology & Electronic Engineering*, 18(1):97–106.

Zhou, K., Shu, C., Li, B., and Lau, J. H. (2019). Early rumour detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1614–1623, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Zubiaga, A., Liakata, M., Procter, R., Hoi, G. W. S., and Tolmie, P. (2016). Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3):e0150989.

Zubiaga, A., Liakata, M., and Procter, R. (2017). Exploiting context for rumour detection in social media. In *SocInfo*.

Zubiaga, A., Aker, A., Bontcheva, K., Liakata, M., and Procter, R. (2018). Detection and resolution of rumours in social media: A survey. *ACM Comput. Surv.*, 51(2):32:1–32:36, February.

8. Language Resource References

Han, S., Gao, J., and Ciravegna, F. (2019). Neural language model based training data augmentation for weakly supervised early rumor detection. In *Proceedings of 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE.

Kochkina, E., Liakata, M., and Zubiaga, A. (2018). All-in-one: Multi-task learning for rumour verification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3402–3413. Association for Computational Linguistics, August.

Appendix: Model Settings

All the parameters of stacked LSTM and attention weights are trained by employing the derivative of the cross-entropy loss function through back-propagation. We use the AdaGrad algorithm for parameter optimisation. As described in Section 3.1., source tweets are filtered out based on two constraints: content length and context size. Context sequence size is set to 200 (i.e. $j = 200$). The length of each ELMo content embedding is 1024, and that of each metadata feature vector is 27. The number of forward LSTM layers in each stacked LSTM is set to 2, and that of hidden units is set to twice input size. The learning rate and weight decay are set to 1e-4 and 1e-5, respectively. All training instances with corresponding context inputs are iterated over in each epoch where batch size is 128. The number of epochs is set to 10 to avoid overfitting. Leaky ReLU is employed in 3 dense layers. Drop out rates 0.2, 0.3, and 0.3 are respectively applied after each of the three layers. Preliminary results show that the RPDNN suffers from "dying ReLU" problem (Maas et al., 2013), which means weights in NNs always drive all inputs to ReLU neurons to negative. This is problematic because ReLu neurons will no longer useful in discriminating the input. Replacing with LReLU fix the problem which gives nonzero gradient for negative value.

Appendix: LOOCV results.

Details of LOO-CV results are presented in Table 6.

Appendix: Analysis of attention degrees

In Figure 10, we present weights of first layers of attention (in "CC" and "CM" columns) and second layer of attention (in "CC+CM" column). The context-level attention weights of example threads are highlighted in different colors according to the rank of their weights in different layers.

Source tweet content

Reports claim Putin disappeared due to impending political coup http://t.co/8IpndT2bsI

	Attention weights					
Context content	**CC**		**CM**		**CC+CM**	
@MailOnline @CathyYoung63	1	0.2755	1	0.1203	10	0.0932
@MailOnline Ah yes to be closer to his billions of rubles	2	0.1386	3	0.1015	8	0.0966
@MailOnline Sure?	3	0.0775	8	0.0946	3	0.1023
@MailOnline Nothing to do with his wife giving birth then?	4	0.0731	10	0.0926	7	0.0998
@MailOnline That's stupid	5	0.0726	9	0.0928	6	0.1004
@MailOnline He should disappear 6 feet under.	5	0.0726	7	0.0963	8	0.0996
@MailOnline he has prolly been having a facelift	5	0.0726	5	0.0981	4	0.1012
Something big is happening right now in Moscow "@MailOnline: Putin disappeared due to impending political coup http://t.co/MKClBsKfvK"	5	0.0726	2	0.1055	2	0.1030
@MailOnline would be nice if it's true but I doubt it. Just one more of Putin's games.	5	0.0726	4	0.1010	1	0.1031
@MailOnline are we ready for war?	5	0.0726	6	0.0973	5	0.1007
Weight sum		1.0003		1		0.9999

Source tweet content

Authorities collecting passports at #MH17 crash site. Australian coat of arms clearly visible. http://t.co/ai16vY46FV http://t.co/JA0gjQt3P5

	Attention weights					
Context content	**CC**		**CM**		**CC+CM**	
@newscomauHQ still unverified footage	2	0.2703	1	0.2015	6	0.1226
@newscomauHQ collecting... They were taking them and showing the cameras the faces of passengers and then throwing them back down. :(	1	0.3427	2	0.1493	8	0.1092
@newscomauHQ @Harriett_Bur it's not authorities...	3	0.1355	3	0.1154	7	0.1115
@newscomauHQ such heart breaking news!	4	0.0614	8	0.1043	5	0.1271
@newscomauHQ Is it just mean who finds these images disturbing. To what length would you have to go to have these passports in your hands?	5	0.0476	5	0.1074	4	0.1303
@newscomauHQ How do you identify the lost souls. They are people with families, probably going on holiday or business not war!	6	0.0475	4	0.1097	3	0.1310
@newscomauHQ Strange that passports look in very good condition when rest of plane demolished.	6	0.0475	7	0.1060	2	0.1329
@newscomauHQ why are they in such good condition reminiscent of the ones found on 9/11	6	0.0475	6	0.1063	1	0.1353
Weight sum		1		0.9999		0.9999

Figure 10: Visualisation of attention weights for example tweets.

The results obtained by the second attention layer (i.e. CC+CM) show that replies expressing doubts and/or questions tend to have higher attention weights. Interestingly, for some replies, the first and second attention layers produce contradictory results, but the latter tends to output more logical results. For instance, the reply "@MailOnline @CathyYoung63" in first example of source tweet is in the first rank according to the first layer's results. However, it does not contain any useful information, and its author is

Table 6: LOOCV results.

Event	**Models**	**P**	**R**	**F1**	**Acc.**
charliehebdo	RPDNN	0.743	0.882	0.807	0.788
	RPDNN-cxt	0.654	**0.956**	0.777	0.725
	RPDNN-SC	**0.754**	0.759	0.757	0.756
	RPDNN-CC	0.712	0.924	0.804	0.698
	RPDNN-CM	0.735	0.944	**0.826**	**0.802**
	RPDNN-Att	0.751	0.868	0.805	0.79
	RPDNN-SC-CM	0.697	0.868	0.773	0.746
	RPDNN-SC-CC	0.559	0.597	0.578	0.563
	(Han et al., 2019)	0.723	0.817	0.767	0.752
	CRFs (Zubiaga et al., 2017)	0.545	0.762	0.636	–
ferguson	RPDNN	0.59	0.884	0.708	0.635
	RPDNN-cxt	0.564	0.781	0.655	0.588
	RPDNN-SC	**0.641**	0.888	**0.745**	**0.695**
	RPDNN-CC	0.567	0.798	0.663	0.594
	RPDNN-CM	0.565	**0.957**	0.710	0.609
	RPDNN-Att	0.627	0.67	0.647	0.635
	RPDNN-SC-CM	0.527	0.996	0.69	0.552
	RPDNN-SC-CC	0.581	0.292	0.389	0.541
	(Han et al., 2019)	0.707	0.535	0.609	0.657
	CRFs (Zubiaga et al., 2017)	0.566	0.394	0.465	–
germanwings	RPDNN	0.594	0.745	0.661	0.618
	RPDNN-cxt	0.577	**0.887**	0.699	0.618
	RPDNN-SC	0.482	0.745	0.585	0.472
	RPDNN-CC	0.555	0.623	0.587	0.561
	RPDNN-CM	0.556	0.708	0.622	0.571
	RPDNN-Att	0.602	0.755	0.67	0.627
	RPDNN-SC-CM	0.511	0.849	0.638	0.519
	RPDNN-SC-CC	0.653	0.65	0.651	0.652
	(Han et al., 2019)	0.601	0.652	0.558	**0.630**
	CRFs (Zubiaga et al., 2017)	**0.743**	0.668	**0.704**	–
ottawashooting	RPDNN	0.647	**0.945**	0.768	0.715
	RPDNN-cxt	0.686	0.924	0.788	0.751
	RPDNN-SC	0.605	0.917	0.729	0.659
	RPDNN-CC	0.743	0.879	**0.805**	0.787
	RPDNN-CM	0.650	**0.945**	0.77	0.718
	RPDNN-Att	0.652	0.914	0.761	0.713
	RPDNN-SC-CM	0.615	0.886	0.726	0.666
	RPDNN-SC-CC	0.63	0.318	0.423	0.566
	(Han et al., 2019)	**0.85**	0.71	0.77	**0.80**
	CRFs (Zubiaga et al., 2017)	0.841	0.585	0.690	–
sydneysiege	RPDNN	**0.784**	0.809	**0.796**	**0.793**
	RPDNN-cxt	0.687	0.861	0.764	0.734
	RPDNN-SC	0.675	0.823	0.741	0.713
	RPDNN-CC	0.673	0.871	0.759	0.724
	RPDNN-CM	0.683	0.847	0.756	0.727
	RPDNN-Att	0.684	**0.902**	0.778	0.743
	RPDNN-SC-CM	0.634	0.90	0.744	0.69
	RPDNN-SC-CC	0.68	0.366	0.476	0.597
	(Han et al., 2019)	0.755	0.644	0.695	0.717
	CRFs (Zubiaga et al., 2017)	0.764	0.385	0.512	–
Twitter 15	RPDNN	0.59	0.79	0.676	0.621
	RPDNN-cxt	0.563	0.734	0.637	0.582
	RPDNN-SC	0.571	0.613	0.591	0.576
	RPDNN-CC	0.581	0.731	0.648	0.602
	RPDNN-CM	0.580	**0.839**	**0.686**	0.616
	RPDNN-Att	**0.595**	0.786	0.677	**0.625**
	RPDNN-SC-CM	0.565	0.69	0.621	0.579
	RPDNN-SC-CC	0.472	0.746	0.578	0.455
Twitter 16	RPDNN	0.588	0.785	0.673	0.618
	RPDNN-cxt	**0.654**	0.723	0.687	0.67
	RPDNN-SC	0.622	**0.827**	**0.71**	**0.662**
	RPDNN-CC	0.585	0.775	0.667	0.613
	RPDNN-CM	0.608	0.7958	0.689	0.641
	RPDNN-Att	0.589	0.801	0.679	0.62
	RPDNN-SC-CM	0.583	0.843	0.69	0.62
	RPDNN-SC-CC	0.573	0.843	0.682	0.607

not a high-impact user. It is ranked last by the second layer. This observation supports the motivation behind adopting multiple attention layers (Yang et al., 2016a; Wang et al., 2017b), that is, they can progressively refine feature maps and focus on more salient features.