Attention-based Graph Neural Networks for Multi-class Fake News Classification

Chen Xihao, Hon Jia Jing, Wang Changqin, Zhang Haolin, Zhang Lei
Group 23
Mentored by Xue Fuzhao
{e0426114, e0324929, e0732558, e0732603, e0732512}@u.nus.edu

Abstract

Fake news is an issue of pressing concern and a potential threat to high-quality journalism and well-informed public discourse. Hence, there is an increasing demand for reliable methods for the automatic detection of fake news. Of the published work trying to distinguish untrusted news from reliable ones, none has yet to exploit attention mechanisms to aid in the extraction of features from news articles. This project explores an attention-based Graph Neural Network for multi-class fake news classification. We first augment Long Short-Term Memory neural networks (LSTMs) to include the attention mechanism and empirically show its effectiveness. Then, we propose a new model that attaches the attention-based LSTM layers as input to Graph Convolutional Networks (GCNs). With experiments ran on currently-available news datasets, we empirically show that our proposed model architecture out-performs current neural network baselines and several recently published models. Lastly, we demonstrate our model's ability to generalize to out-of-domain datasets. Code is available at https://github.com/archiewang0716/CS4248-Fake-News-Detection.

1 Introduction

The Internet is a free domain where anyone can post anything they like. However, users usually turn to "trustable" sources, such as newspaper sites, for their daily share of the on-goings in the world. While these articles may be written by trained and credible professionals, they do not always provide impartial details of the topic. Furthermore, a bulk of viewers also get their news from less trust-worthy sources such as Twitter and Facebook. Collectively, they may provide information that are biased, unreliable, or of political origins. For example, US media Fox News is known to be right-swinging and always complementing the Republican Party.

The problem of automatically detecting the reliability of a piece of news (be it an article or a Tweet) arises. Moreover, unreliable news can be further categorised into three different sub-types: satire, hoax and propaganda. Thus, we model this more comprehensively as a four-way classification problem.

In this project, we survey the use of attention layers and their effect in improving various Long short-term Memory neural network model architectures. We experiment these changes on the "Labelled Unreliable News" and "Satirical and Legitimate News" databases.

2 Related Work

In recent years, much work have been done to address related issues, such as fact checking. For example, Zhong et al. (2019) proposed methods to reason about the semantic-level structure of retrievable evidence for fact checking. Compared to fact checking, fake news detection focuses more on news-related documents. Some works (Jin et al. (2016); Tacchini et al. (2017)) consider the detection of fake news in social media, while other research (Shu et al. (2019); Zhang et al. (2020)) discuss methods to distinguish fake news with the help of social contexts.

The writing style of a news document has also shown to help in the detection of fake news. For example, some works (Oshikawa et al. (2018); Sitaula et al. (2020)) placed emphasis on writing styles to improve classification performance. Rashkin et al. (2017) explored the use of linguistic features such as profanity, number, hedges, etc alongside text as input, and the use of max-entropy classifiers and LSTMs. They showed that with only text input, LSTM outperforms the other models in 4-way classification tasks on the Labelled Unreliable News (LUN) dataset, and that lexical features can help the model understand the differences between the unreliable and reliable news.

Furthermore, recent works, such as Yang et al. (2017), have shown that the sophisticated neural network models can be used for satirical news classification. Vaibhav et al. (2019) proposed the use of GCNs (Zhang et al., 2019) to capture interactions between sentences in the same document. This is the first paper to discuss about deep neural network models in this dataset. It is based on the observation that while two sentences may be far apart, they can be closely-related, e.g. by substantiating another sentence. As GCNs require a graph as an input, they also proposed a novel way to represent a document as a fully-connected graph, by characterising each sentence as a vertex, and an edge by the semantic textural similarity between the two connected sentences. They have also shown empirically the performance of the new Graph Attention Network (GAT) model on the LUN and the Satirical and Legitimate News (SLN) datasets.

Moreover, Vaswani et al. (2017) proposed the Transformer, an architecture based on the simple neural network, that removes reliance on recurrence, and instead, focuses on the use of attention mechanism to achieve desirable classification results. This research is the basis of our project.

Based on the Transformer, Liu and Guo (2019) introduced bidirectional Encoder Representation for Transformer (BERT), demonstrating the importance of bidirectional representation for language. BERT becomes the first fine-tuning based representation model that achieves state-of-the-art performance on a large suite of sentence-level and token-level tasks, outperforming many task-specific architectures.

To the best of our knowledge, previous work based on the LUN dataset have yet to explore the integration of attention mechanisms with the sequential document encoder models to improve classification performance.

3 Methods

3.1 Dataset

In this project, we use two currently-available datasets for training and testing our models.

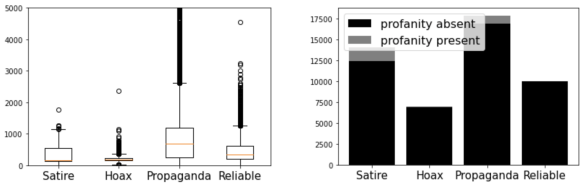
Labeled Unreliable News Database (LUN) (Rashkin et al., 2017) is the main dataset used. It consists of around 50,000 unique news documents of four labels, namely "satire", "hoax", "propaganda" and "trusted/reliable". The LUN dataset has a predefined split, namely LUN-train for the training set and LUN-test for the test set. We will explore their differences in more details later.

We also make use of the Satirical and Legitimate News Database (SLN) (Rubin et al., 2016), a dataset containing two labels, "satire" and "legitimate", to cross-check the models and ensure they are able to generalise to a similar classification problem. Details involving experimenting on this dataset will not be discussed, unless necessary.

3.2 Corpus Analysis

By studying the distribution of features in the dataset, we explore the use feature engineering, and evaluate its usefulness. Features based on the content of the document can be extracted to help in classification. Stylometric, syntactic and semantic features (Bezerra, 2021; Owais et al., 2015; Volkova et al., 2017; Barrón-Cedeño et al., 2019; Rashkin et al., 2017) can be used as input alongside features such as word embedding to improve model performance.

A total of 12 features were examined: length of document, number of quotations, number of first person singular pronouns, number of third person plural pronouns, count of numbers, presence of profanity (Owais et al., 2015), sentiment scores, presence of hedging and booster words (Farrokhi and Emami, 2008). Of 12 features implemented, 6 features stood out to have improved results, namely: length of document, number of quotations, count of numbers, presence of profanity, compound sentiment scores and presence of hedging words. For example, in Figure 1a, we observe that the length of a document may be used to better distinguish between articles of propaganda and hoax classes. Furthermore, from Figure 1b, we observe that presence of profanity may improve the differentiating satire and propaganda from hoax and reliable articles.



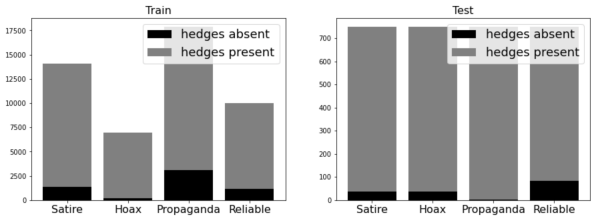
(a) Lengths of documents (b) Frequency of profanity

Figure 1: Distribution in the LUN-train set

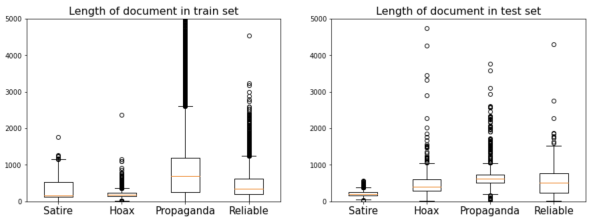
The evaluation of the usefulness of these features are elaborated in A.1.

To investigate the deterioration of performance when including the engineered features, we com-

pared the distribution of features in LUN-train and LUN-test, shown in Figure 2a and Figure 2b. We observe that the distribution in both sets are largely different. Hence, we believe that including these features may easily result in over-fitting, thus explaining the performance degradation. While these features may help to differentiate the classes in LUN-train, the distribution no longer hold in the LUN-test dataset or other unseen corpus.



(a) Distribution of frequency of hedges



(b) Distribution of lengths of documents

Figure 2: Comparison of distributions between the LUN-train and LUN-test sets

3.3 Word Embeddings

We explore two types of initializations to the embedding layer.

Xavier Uniform The Xavier Uniform distribution (Glorot and Bengio, 2010) is a state-of-the-art technique to initialize the weights in the embedding layer to our model. In the Xavier distribution, each bias is set to 0 and each weight is sampled from the Xavier Uniform distribution:

$$W \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}\right] \quad (1)$$

where n_j and n_{j+1} are the sizes of the input and output layers respectively.

The Xavier Uniform distribution is proven to effectively take advantage of the Sigmoid activation function, which is an activation that's commonly used in neural networks. The derivative of the Sigmoid function is only significant over a small range around the origin, and vanishes quickly when the input is far from 0. The Xavier initialization maintains the variance of activations and back-propagated gradients throughout the layers of the neural network.

Pretrained Embedding We have also explored the use of a pretrained embedding, Global Vectors for Word Representation (GloVe) (Pennington et al., 2014). To ensure the fairness of our comparison to the recently published models, we used 100-dimensional word vectors as our embedding weights and set them to be trainable together with the neural network.

3.4 Baseline Models

We first explore the use of some tried-and-tested models to establish a baseline for comparison.

Logistic Regression Logistic Regression is the foundation idea to more complex neural networks. After preprocessing and initializing the embedding weights, we trained a Logistic Regression model with a maximum of 2000 iterations.

CNN We established a simple Convolutional Neural Network (CNN) model to extract relationships between neighbouring words in the document (Chen, 2015). We used 100 filters, each with a window size of 3 to capture relationships among immediately-adjacent words. The CNN model is used as an encoder and its outputs are then fed as inputs into a fully-connected neural network (FCNN) for classification.

LSTM A Long Short-Term Memory (LSTM) neural network model is used to better capture sequences of words, and to extract only words and relationships which the model deems useful (Hochreiter and Schmidhuber, 1997). We implemented both a uni-directional LSTM and a bi-direction LSTM (BiLSTM) encoder. In comparison, the BiLSTM is able to better extract the context of a word by using a lookahead down the sentence or document. The LSTM is used as an encoderand its outputs are then fed as inputs into an FCNN for classification.

BERT + LSTM BERT is a novel transformer-based pre-training technique to encode a word's relationship with other words preceding and following it (Devlin et al., 2018). As compared to other encoders that use sequential input, the transformer class of encoders takes in an entire sequence at once for context extraction, and are considered bidirectional. This property allows the encoder to learn the context of a word based on all of the

words in the sequence. We use a pretrained BERT to extract sentence vectors for each sentence in each document. The output vectors from BERT are then passed through an LSTM and finally fed as inputs into an FCNN for classification of each document.

3.5 Attention

An important feature of human perception is that we do not process all the stimuli from the outside world at once. Instead, we focus on the important parts first in order to get information that are necessary for understanding. We say that these information are given "attention".

Likewise in language modelling, various words, phrases and sentences in a document are more crucial and provide more information in understanding the document. Hence, the idea of attention can be applied to context extraction as well, through an attention mechanism.

An attention mechanism allows the language model to dynamically "highlight" features that it deems relevant, by adaptively selecting the most relevant input features and give higher weights to the corresponding original feature sequence.

Formally, an attention mechanism can be described as mapping a query and a set of key-value pairs to an output (Kim et al., 2017). The output is computed as a weighted sum of values, where the weight assigned to each value is computed by the query's compatibility function with the corresponding key as shown:

$$z^t = \sum_{j=1}^{w_n} \beta^j h^j \tag{2}$$

where z^t is the the t^{th} context vector, β^j is the attention weight and h^j is the j^{th} hidden state. The attention weight β^j is obtained by normalizing the output score of a feed-forward neural network:

$$\beta^j = \frac{\exp\left(\alpha^j\right)}{\sum_{k=1}^{w_n} \exp\left(\alpha^k\right)} \tag{3}$$

and α^j is the score obtained based on the LSTM hidden state h^{i-1} using an alignment model a:

$$\alpha^i = a\left(h^{i-1}, z^{t-1}\right). \tag{4}$$

where a is parametrized using feed-forward neural network and is jointly trained with all the other components of the proposed system.

3.6 Graph Convolutional Network

To learn the correlation and the interaction among sentences in the same document, we used a Graph Convolutional Network (GCN) to obtain inputs to the attention-based LSTM (Kipf and Welling, 2016). This is motivated by the fact that GCNs are better at capturing sentence interactions globally than LSTMs.

The GCN architecture incorporates the convolution technique from CNN. Unlike CNN, however, GCNs learn from complex non-Euclidean graphs:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{5}$$

where $\tilde{A}$ is the self-looped adjacency matrix, $\tilde{D}$ is the degree matrix, $H^{(l)}$ is the matrix representation of the l^{th} hidden layer and W represents the weight matrix of the network.

By applying the renormalization trick to GCN layers, GCN is capable of learning both of the sentence (vertex) features and cross-sentence correlations (adjacency and degree matrices) without having the issues of vanishing and exploding gradients.

3.7 Proposed Model and Training

A summarized diagram of our proposed model architecture is shown in Figure 3.

In the model, we first initialize each weight in the embedding layer by sampling from the Xavier Uniform distribution section 3.3. Then, an LSTM or BiLSTM layer is applied to encode the sentences, followed by an Attention layer to extract the feature sequences of sentence vectors effectively. The output from the attention layer is fed into a Graph Neural Network. Finally, we applied a fully connected (projection) layer which acts as the decoder, followed by a Softmax layer to perform classification.

In models that include a GCN, we define the graph $G_d = (V, E)$ where V is the feature matrix obtained from the set of vectors for each sentence in the document d, and E is the adjacency matrix that captures pairwise similarity between sentences, similar to that proposed by Vaibhav et al. (2019).

Namely, we propose the following models:

- *AT-LSTM* for attention-LSTM (unidirectional),
- *AT-BiLSTM* for attention-bidirectional LSTM,

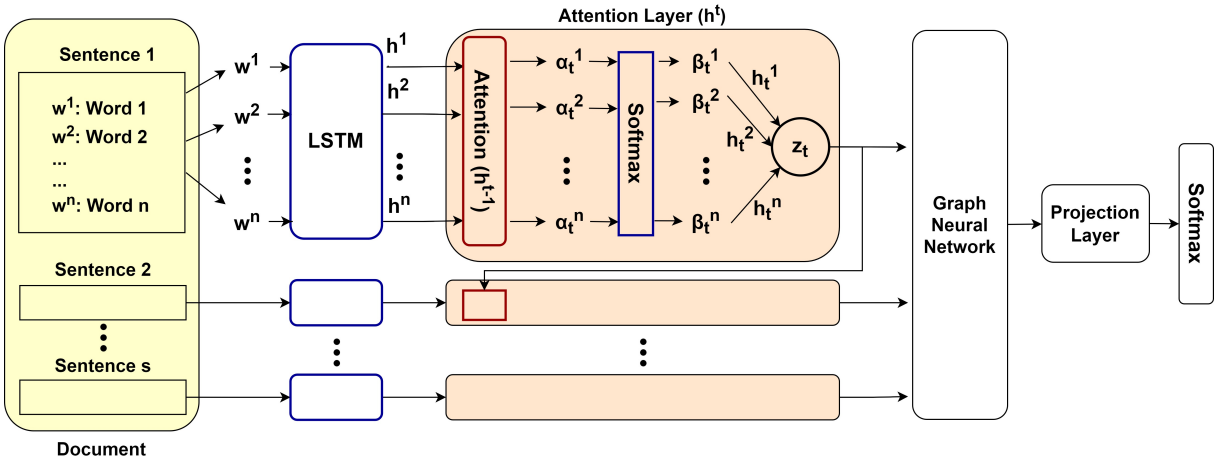


Figure 3: Proposed Attention-based Graph Neural Network, where w^i represents the i^{th} word from the sentence, h^j is the j^{th} hidden state of the LSTM encoder, which is the input to the attention layer. The output z_t of one attention layer is used as the input of the next attention layer. The document is conveted to a document-graph G which serves as input to the Graph Neural Network. The projection layer is an FCNN that acts as the decoder. This is elaborated in subsection 3.7.

- *AT-LSTM + GCN* for the variant of AT-LSTM with GCN embedding,
- *AT-BiLSTM + GCN* for the variant of AT-BiLSTM with GCN embedding.

In training the model, we exploit the cross-entropy loss over the training data with L2-norm:

$$\mathcal{L} = -\sum_{i \in D_{train}} \sum_{j=1} Y_{ij} \cdot \log Z_{ij} + \eta \|\theta\|_2, \quad (6)$$

where D_{train} is the set of news documents for training, Y is the corresponding label indicator matrix, θ is the model parameters and η is the regularization factor.

We then perform stochastic gradient descent (SGD) using this loss function.

4 Experiments

We conducted experiments for different models on both LUN and SLN datasets. Based on existing papers, we conducted both a 2-way and a 4-way classification to analyze and compare the performance of the selected models.

4.1 Hyperparameters

Firstly, we set the dimensions of the embedding layer to be 100 to be consistent with Vaibhav et al. (2019).

We used the Adam optimizer to improve SGD (Kingma and Ba, 2014), with its initial learning rate set to 1e-3.

In the GCN, we used Leaky ReLU with a slope of 0.2 for activation in the hidden layers.

Furthermore, we configured the learning rate to be decaying in order to decrease the variance of SGD The learning rate is halved for every 2 consecutive epochs that did not see any improvements to the accuracy.

We set our regularization factor η to be $1e-6$.

4.2 Performance Metrics

We used accuracy, precision, recall and F1-score to evaluate the performance of our models. As our predefined train-test split had an imbalance in class labels, we value the use of F1-score as the main metric, which is a more robust indicator as compared to accuracy. To remain consistent with current publications, we used the macro F1-score in evaluation in subsection 4.3.

4.3 Overall Results

As mentioned in section 4, we performed experiments for both a 2-way and a 4-way classification task. We will discuss them separately here.[1]

2-way classification In the 2-way classification, we trained our models on the LUN-train split and used LUN-test for validation (we only included the satire and reliable news in LUN dataset for the 2-way task). To evaluate the model's performance

[1]We only report macro F1-score here. The other metrics like accuracy, precision, recall and micro F1-score are detailed on our GitHub repository.

Model	LUN-test	SLN
Logistic Regression	91.8	80.1
CNN	96.6	66.3
BERT	91.3	75.7
LSTM	95.5	82.5
BiLSTM	95.7	76.7
TF-IDF + SVM (Rubin et al., 2016)	97.2	87.0
GCN (Vaibhav et al., 2019)	98.0	85.8
Our Models		
AT-LSTM	96.2	84.2
AT-BiLSTM	97.5	66.3
AT-LSTM + GCN	98.5	87.7
AT-BiLSTM + GCN	**98.7**	**91.1**

Table 1: 2-way classification results.

Model	LUN-dev	LUN-test
Logistic Regression	91.8	52.1
CNN	96.1	44.4
BERT	95.0	55.1
LSTM	87.7	48.8
BiLSTM	93.7	56.0
MaxEnt (Rashkin et al., 2017)	91.0	65.0
GCN (Vaibhav et al., 2019)	96.76	65.0
Our Models		
AT-LSTM	92.5	62.1
AT-BiLSTM	95.6	61.5
AT-LSTM + GCN	**98.2**	**66.1**
AT-BiLSTM + GCN	98.1	56.4

Table 2: 4-way classification results.

on out-of-domain datasets, we performed classification on the entire SLN dataset. The evaluation results are reported in Table 1.

We observe that *Attention-BiLSTM + GCN* is the best performing model, with a macro F1-score of 91.1. By attaching a GCN, we increased the F1-score of *Attention-BiLSTM* by approximately 25 on the SLN dataset, which is a significant improvement. This shows that the extracted sentence-level relationships can be extremely useful.

When comparing *Attention-LSTM* to *Attention-LSTM + GCN* and comparing *Attention-BiLSTM* to *Attention-BiLSTM + GCN*, we can clearly see the performance improvements resultant from the additional GCN.

However, comparing *Attention-LSTM* to *Attention-BiLSTM*, we see an increase in performance in the LUN-test but the *Attention-BiLSTM* has worse performance when testing on the out-of-domain SLN set. We suspect that the bidirectional encoder, while useful in capturing more sequential contextual relationships, had overfitted on the additional relationships which are only present in the LUN dataset.

4-way classification In the 4-way classification, we created a train-validation split using the LUN dataset with an 80:20 split (80% for training and the 20% to create the LUN-dev validation set). We retained the predefined LUN-test dataset to simulate unseen data of a distinctively different distribution (as elaborated in subsection 3.1). The 4-way classification results are shown in Table 2.

In this 4-way classification task, we see that alll models struggled to score when testing on the LUN-test dataset, as compared to their performance in the LUN-dev validation set. As explained previously in subsection 3.1, we suspect that this is mainly due to the drastically different distribution of structure and features in the two subsets.

The *Attention-LSTM + GCN* model had attained the highest F1-score when testing on the predefined LUN-test dataset. Comparing *Attention-LSTM* to *Attention-LSTM + GCN*, we can see a significant increase in performance (around 6.0 in LUN-dev and 4.0 in LUN-test). However, such a trend is not observed in the BiLSTM variants. We suspect that this is more likely to be due to overfitting caused by the bidirectional encoding than by the GCN.

5 Discussion

5.1 Uni- vs Bi-directional Encoder

To encode sentences in a text into embedding vectors that can be used as input in GCN, we experimented with both of the uni-directional LSTM and the bi-directional LSTM architectures (Chiu and Nichols, 2016). As shown in Table 1, in a 2-way classification task, GCN with BiLSTM encoder outperforms GCN with LSTM in F1-score under both test. However, in the 4-way classification task in Table 2, the BiLSTM-encoded GCN performs worse than the LSTM-encoded GCN in both tests.

Usually, a BiLSTM model is more powerful than

LSTM in encoding tasks, as the BiLSTM learns a word's context in both the forward and backward direction. In the simpler 2-way classification task, the BiLSTM encoder is able to better differentiate the two classes. Since the number of classes small, the BiLSTM outperforms the unidirecional LSTM model by better capturing the contextual meaning of sentences.

However, the BiLSTM is unable to show its advantage in a more complex 4-way classification task. We believe that in a multi-class classification task, the BiLSTM model tries to encode context with more precision and details (as compared to the 2-way) to better distinguish between the four classes. Furthermore, as discussed in subsection 3.1, the distribution in LUN-dev (derived from LUN-train) and LUN-test are significantly different, by capturing the context more precisely may have been the main reason why the BiLSTM is more prone to overfitting. Hence, it struggles to generalize to the LUN-test dataset, producing poorer results in the 4-way classification as discussed in section 4.3.

5.2 Importance of Attention Mechanisms

Table 2 shows the improvement in performance in LSTM models with attention mechanisms. Our *AT-LSTM* and *AT-BiLSTM* outperforms the baseline models LSTM and BiLSTM respectively, achieving a significant improvement in F1-scores on the LUN-test. This emperically indicates the importance of the attention mechanism in the multi-class fake news detection tasks.

We leverage the self-attention mechanism on the top of the encoder LSTMs as an inner-attention for the hidden states h^i, to extract sequential features of sentence vectors in each document (Vaswani et al., 2017). This is to address the shortfall in LSTMs, which are adept to capturing relationships between a sentence and its close neighbours in the document. With the self-attention mechanism built into the LSTM, the encoder is able to better extract relevant information that are further from sentence and potentially prevent the encoder from forgetting important features that should be kept "in memory".

In order to prevent LSTM encoders from information loss (as all the information are compressed into an output vector in LSTM encoder), we designed an attention-based architecture which gives attention to every hidden state in each layer of the LSTM encoder. This is in comparison to only giving attention to the final output of the encoder. Afterwards, the integrated encoder can decide which hidden state is more informative, extracting the key features according to the attention weights. This helps in producing a more detailed sentence feature vector with strong context embedded.

5.3 Use of GCN

In subsection 4.3, we observed that our attention-based LSTM models with GCN have achieved better performance than without using GCNs, thus showing empirically the effectiveness of GCNs in capturing pairwise relationships between sentences in the entire document.

In our proposed model architecture, we represent the usual document classification task as graph-based classification task, as described in subsection 3.7. An important observation to be made in documents is that sentences are related to not only just sentences that are in its field of neighbours. This characteristic is amplified with increasing lengths of documents. For example, if we have two documents, one with 10 sentences and the other with 100, both explaining the same 3 ideas, we would expect the short document to use on average 3 sentences on each idea, but the longer one to use about 33 sentences on each.

Models such as CNN are able to capture complex relationships between a word and its close neighbours, or between a sentence and its close neighbours. However, the "near neighbours" are determined by the size of the convolutional kernels. We note that the size of the kernels in a model must be initialized to a fixed size prior to training. This introduces a large amount inflexlibility, which is a potential shortcoming of traditional convolutional neural networks and may reduce the model's ability to generalize, when given documents of drastically different lengths.

In contrast, G_d is a dynamically sized fully connected graph based on the number of sentences in the document d. Such a graph is capable of fully capturing the relationship between every pair of sentences. The graph representation acts as an input to the GCN embedding model, in which the convolution and pooling layers are able to extract graph-wide information to produce a feature map for the entire document. Such a method substantially increases the amount of information that can be captured compared to a CNN with a fixed filter size. We attribute the majority of improvement in

performance to this property.

6 Conclusion

In this project, we analyzed the use of attention mechanisms in Long Short-Term Memory networks and their empirical performance. With this as a basis, we proposed a new model architecture that uses Attention-based LSTMs as encoders to the Graph Convolutional Network and empirically showed its performance improvement in fake news classification tasks. We also demonstrated its ability to generalize to similar tasks that are out-of-domain.

We believe that future research projects can explore the use of stacked sentence encoders (i.e. LSTMs and GRUs), which have also shown potential in extracting contextual information from the documents with the help of attention mechanisms. Moreover, a combination of neural self-attention and max-pooling layers through a concatenate operation can experimented as an addition to our encoders for context vector extraction.

Finally, we note that such deep neural networks lack human-interpretability and our proposed model, methods and discussions are based on currently-known properties and effects of the individual components as observed from repeated experimentation, which may not be entirely accurate. While attention mechanisms in our model mimics that of human attention, they may behave in ways we are unable to interpret. Hence, future projects may head in the direction of explainable attention mechanism to better understand the effects of these components.

References

Alberto Barrón-Cedeño, Giovanni Martino, Israa Jaradat, and Preslav Nakov. 2019. Proppy: A system to unmask propaganda in online news.

Jose Fabio Ribeiro Bezerra. 2021. Content-based fake news classification through modified voting ensemble. *Journal of Information and Telecommunication*, 5(4):499–513.

Yahui Chen. 2015. Convolutional neural network for sentence classification. Master's thesis, University of Waterloo.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the association for computational linguistics*, 4:357–370.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Farahman Farrokhi and S N Emami. 2008. Hedges and boosters in academic writing: Native vs. non-native research articles in applied linguistics and engineering. *Journal of Applied Linguistics*, 1:62–98.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

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

Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. 2016. News verification by exploiting conflicting social viewpoints in microblogs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

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

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

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Gang Liu and Jiabao Guo. 2019. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337:325–338.

Ray Oshikawa, Jing Qian, and William Yang Wang. 2018. A survey on natural language processing for fake news detection. *arXiv preprint arXiv:1811.00770*.

Syed Taha Owais, Tabrez Nafis, and Seema Khanna. 2015. An improved method for detection of satire from user-generated content.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2931–2937.

Victoria Rubin, Nadia Conroy, Yimin Chen, and Sarah Cornwell. 2016. Fake news or truth? using satirical cues to detect potentially misleading news.

Kai Shu, Suhang Wang, and Huan Liu. 2019. Beyond news contents: The role of social context for fake news detection. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 312–320.

Niraj Sitaula, Chilukuri K Mohan, Jennifer Grygiel, Xinyi Zhou, and Reza Zafarani. 2020. Credibility-based fake news detection. In *Disinformation, Misinformation, and Fake News in Social Media*, pages 163–182. Springer.

Eugenio Tacchini, Gabriele Ballarin, Marco L Della Vedova, Stefano Moret, and Luca de Alfaro. 2017. Some like it hoax: Automated fake news detection in social networks. *arXiv preprint arXiv:1704.07506*.

Vaibhav Vaibhav, Raghuram Mandyam Annasamy, and Eduard Hovy. 2019. Do sentence interactions matter? leveraging sentence level representations for fake news classification. *arXiv preprint arXiv:1910.12203*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Svitlana Volkova, Kyle Shaffer, Jin Yea Jang, and Nathan Hodas. 2017. Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on Twitter. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 647–653, Vancouver, Canada. Association for Computational Linguistics.

Fan Yang, Arjun Mukherjee, and Eduard Dragut. 2017. Satirical news detection and analysis using attention mechanism and linguistic features. *arXiv preprint arXiv:1709.01189*.

Jiawei Zhang, Bowen Dong, and S Yu Philip. 2020. Fakedetector: Effective fake news detection with

deep diffusive neural network. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1826–1829. IEEE.

Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23.

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2019. Reasoning over semantic-level graph for fact checking. *arXiv preprint arXiv:1909.03745*.

Acknowledgements

The team would like to thank Xue Fuzhao, our mentor, for his guidance and support throughout the problem formulation of our project.

We would also like to thank the professors Chris and Min for taking us through the core concepts of natural language processing and inspiring us to step into this field.

The team apologizes in the delay of the submission of this report, due to the technical difficulties experienced by one of the team members (Xihao). We would like to extend our utmost gratitude to the teaching team for accommodating with our request for extension.

Statement of Independent Work

1A. Declaration of Original Work. By entering our Student IDs below, we certify that we completed our assignment independently of all others (except where sanctioned during in-class sessions), obeying the class policy outlined in the introductory lecture. In particular, we are allowed to discuss the problems and solutions in this assignment, but have waited at least 30 minutes by doing other activities unrelated to class before attempting to complete or modify our answers as per the class policy.

1B. Exception to the Class Policy. We did not follow the CS4248 Class Policy in doing this assignment. This text explains why and how we believe we should be assessed for this assignment given the circumstances explained.

Signed, A0189345R, A0206191M, A0235962X, A0236008M, A0236053M.

A Appendix

A.1 Feature Engineering

To evaluate the usefulness of the features extracted in subsection 3.1, we trained two models with identical hidden layers for fairness, in the form of a feed-forward neural network with one hidden layer, differing only in their input layers. The two input layers are: 1. GloVe word embedding, 2. Combination of GloVe and the 6 extracted features, resepectively. The results are shown in Table 3.

Input	Accuracy	F1
GloVe embedding	0.74	0.73
GloVe embedding + features	0.62	0.62

Table 3: Performance with feature engineering