Universität Koblenz - Landau
Department für Informatik,
Institute for Web Science and Technologies
Prof. Dr. Steffen Staab
Dr. Chandan Kumar
Lukas Schmelzeisen

4. September 2018

Master Thesis Kandhasamy Rajasekaran

Fake news classification through Wikipedia using recurrent neural networks

1 Introduction

According to Lazer et al.[LBB+18], Fake news is a false information, constructed intentionally (disinformation) or unintentionally (misinformation) and the publishers does not have rigorous news media's editorial norms for making sure of accuracy and credibility. The focus of this master thesis is to classify a claim as true or fake news by using Wikipedia as a ground reality. In this thesis work, the information in Wikipedia will be captured using different approaches and will be evaluated based on how good it classifies fake news.

Fake news is prevalent and a research study from Allcot and Gentzkow [AG17] state that "average American adult saw on the order of one or perhaps several fake news stories in the months around the election, with just over half of those who recalled seeing them believing them". A large-scale empirical study with twitter dataset by Vosoughi et al. [VRA18], reveals that fake news spread longer, faster, deeper and broader than the legitimate news. It also reveals that the effects of fake news are more prominent in a political context than news about terrorism, natural disaster, science and other domains.

Most often fake news is detected by people and organizations through their common sense. There are many facts checking websites such as for e.g. snopes.com, factcheck.org, politifact.com etc. which uses collaborative effort of domain experts. In these websites, each news is tagged with a fact meter to refer its authenticity. The main strategy to handle fake news is to check against a reliable source of information and claim its integrity. Although this is fairly good, since it requires manual effort, it is not available for all domains. Also, it is unmatchable to the rate at which the information is produced because of many social networks, blogging sites etc.

Wikipedia is a free online encyclopedia available in more than 300 languages with a principle that anyone can edit [Wal05]. It covers a wide range of domains whereas many articles are available in each domain. According to Alexa[1] and SimilarWeb[2], Wikipedia is considered to be the fifth most popular website. According to Wikipedia in 2018, the English Wikipedia consists the highest number of articles amounting to approximately 500 million. Also, the frequency of the updates in English Wikipedia is very high and it is approximately equal to 10 updates per second and 600 articles per day. According to a research by Halavais et al. [HL08], it is estimated that domains such as music, geography, science and literature constitute atleast 5 percent of total articles each in Wikipedia. Although it

[1] https://www.alexa.com
[2] https://www.similarweb.com

can be edited by anyone, an investigation carried out by Nature[3], reveals that the quality of content is similar to another encyclopedia such as Britannica[4] [Wal05]. Although there can be malicious users in Wikipedia, the culture and the community ensures most of the high impactful errors are rectified very quickly[PCL+07]. Thus English Wikipedia can be used as a proxy for a reliable source of information since most of its content is true.

In this master thesis, the Wikipedia will be considered as a ground reality or as a source of experts opinion and this knowledge will be used to cross-check claims automatically. The presence or absence of information in Wikipedia will be used as a proxy for information being considered as true or fake. Recurrent neural networks with different configurations will be used to understand Wikipedia and the results of each will be compared against each other to understand better.

The "Related work" section consists a brief summary of different researches done in the past which are relevant to fake news classification using semantics and platform specific features. An introduction of neural networks, different architectures of neural networks to model sequence data and the training process involved to optimize the model parameter values are covered in "Background study" section. The "Approach" section comprises the overall work to be done in the thesis which includes different techniques of modeling the information in Wikipedia to build a good fake news classifier. The "Evaluation" section describes what factors will be considered for evaluting the classifier and what standard test datasets will be used to benchmark against each other.

2 Related work

Many efforts in research have been put in to detect fake news in microblogging platforms such as Twitter[5]. Most of these works classify the news as truth or fake by using the platform/user-specific information such as how popular the post is, the credibility of the user who shared it, diffusion patterns etc [LNL+15] [MGW+15]. Zhao, et al. [ZRM15] have used cue terms such as 'not true', 'unconfirmed' etc in retweets or the comments to detect fake news. The reasoning in their approach is that when people exposed to fake news they will comment or retweet with such words as a response to the post. Other studies focused on using the temporal characteristics of fake news during the spread. Kwon et al.[KCJ+13] used tweet volume in time series and Ma, et al.[MGW+15] measured variations of social context features over time.

All the research indicated before used datasets which are smaller in size. Wang [Wan17] curated dataset which consists approximately 13000 short statements by mining politfact.com covering a decade of information. In this approach, six machine learning models are built ranging from logistic regression to the convolutional neural network and compared. Along with the text data, metadata such as the speaker, subject, speaker history are also used. The convolutional neural network model used to capture surface level linguistics along with metadata performed better than other models.

All the attempts made in the above researches involve handcrafted feature engineering which is critical, biased and very time-consuming.

Jing Ma et al.[MGM+] efforts are focused on building a recurrent neural network (RNN) to detect rumors from Microblogs such as Twitter and Weibo[6] effectively. The training dataset used is obtained

[3]https://www.nature.com
[4]https://www.britannica.com/
[5]https://twitter.com/
[6]https://www.weibo.com

by using constructed keywords in fake and true news from debunking services such as Snopes[7] and Sina community management center[8]. These keywords are used in Search API's of Microblogs and the tweet results of a search are labeled respectively. The social context information of a post and all its relevant posts such as comments or retweets is modeled as a variable-length time series. RNNs with different configurations such as one or two layers of GRU and LSTM are used and achieved very good results in capturing long-distance dependencies of temporal and textual representations of posts under supervision. This method completely avoids all the handcrafted feature engineering efforts which are biased and time-consuming. It produces better results with datasets from Twitter and Sina Weibo than all of the traditional Machine Learning methods. RNNs with two layers of GRU gave the best results and it was also very quick in predicting the rumor than the average time from debunking services.

The above method uses platform-specific features immensely such as retweets, comments in a tweet and the temporal correlation between them to figure out whether the news is true or fake. The features specified in one platform will be different from the other and it is not guaranteed that this methodology will give the same results for the same news in two different platforms. The semantics of the tweets are not used and if used then it would remain the same across platforms.

Ciampaglia et al. [CSR+15] used DBPedia[9] for checking computationally whether a given information is factual or not. The work uses the knowledge graph built from DBPedia which represents infobox section in Wikipedia. This represents only non-controversial and factual information which is analogous to human collected information. The methodology formulates the problem of checking facts into a network analysis problem which is finding the shortest path between nodes (subject and object of a sentence) in a graph. The aggregated generalities of nodes along a path in a weighted undirected graph are used as a metric for measuring the authenticity of information. The more the elements are generic; the weaker the truthfulness is. The genericness of a node is obtained by the degree of that node - no. of nodes connected to that node. The truthfulness of the information is improved if there exists at least one path from subject to an object with minimal non-generic nodes. This approach exploits the indirect connections to a great extent with distance constraints in a knowledge graph. The approach gave promising results when tested with datasets containing simple factual information about history, geography, entertainment, and biography.

The above research is a good initial step towards an automated fact checker system using only the semantics of data. The problem of fake news attempted is very primitive and uses only 'is' or 'type of' relation. The current fake news is very complex and subtle when it comes to ambiguities. In this approach, DBPedia is used and according to their sources, the update/synch frequency is slower than Wikipedia by 6 to 18 months.

3 Background Study

Neural networks are state of the art models to build learning systems. In the beginning, neural networks were inspired by the brain's computational mechanism [MP43]. Neural networks compose many interconnected, fundamental, functional units called neurons. Each neuron in the network takes in multiple scalar inputs and multiplies each input by a weight and then sums them, adds the result with a bias, applies a non-linear function at the end, which gives out a scalar output. There are different architectures of neural networks which vary mostly in how the neurons are connected to each other and how the weights are managed.

[7] https://www.snopes.com/

[8] https://www.sina.com.cn/

[9] https://wiki.dbpedia.org/

Feedforward neural networks [SKP97] can have multiple layers and each neuron in one layer is connected with every other neuron in the subsequent layer as given in the Figure 1

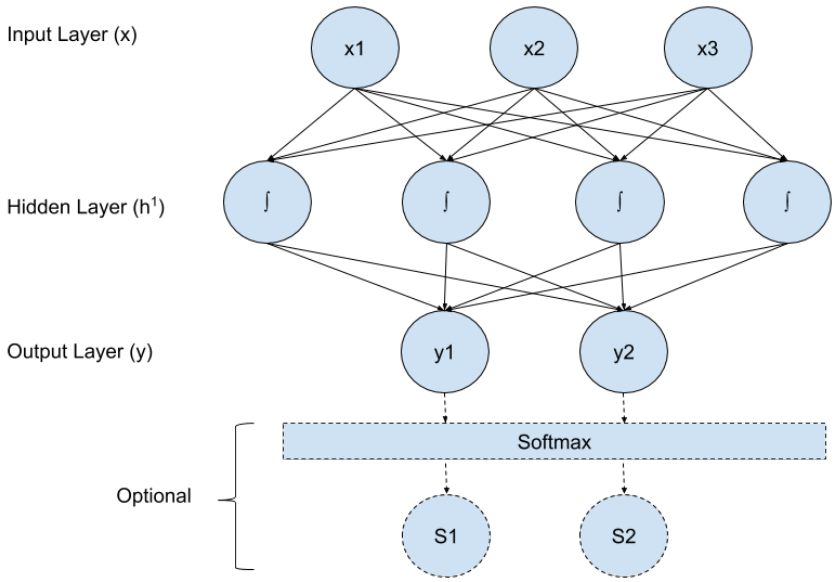

Figure 1: A Feed forward neural network.

$$\mathrm{NN}_{\mathrm{MLP1}}(x) = y \tag{1}$$

$$h^1 = g(xW^1 + b^1) \tag{2}$$

$$y = h^1 W^2 + b^2 \tag{3}$$

$$x \in \mathbb{R}^{d_{in}}, W^1 \in \mathbb{R}^{d_{in} * d_1}, b^1 \in \mathbb{R}^{d_1}, W^2 \in \mathbb{R}^{d_1 * d_2}, b^2 \in \mathbb{R}^{d_2}$$

Here W^1 and b^1 are a matrix and a bias term for the first linear transformation of input, g is a non-linear function that is applied element-wise, and W^2 and b^2 are matrix and bias term for second linear transform. With respect to the figure 1, the values of d_{in}, d_1 and d_2 are 3, 4 and 2 respectively.

There are 3 layers in the figure 1. Each circle is a neuron with incoming lines as inputs and outgoing lines as outputs to the next layer. Each line carries a weight and the input layer has no weights since it has no incoming lines. The input layer consists of 3 neurons and the extracted features of raw data will be sent through these neurons. The hidden layer consists of 4 neurons, where as each neuron takes 3 inputs from input layer. Each input is multiplied with a unique weight variable and added together. Finally, the output is added with 1 bias variable and will be passed to a non-linear activation function as shown in equation 2. The activation function help the neural network models to approximate any nonlinear function. Different activation functions pose different advantages. Some activation functions that will be used in this thesis are sigmoid, hyperbolic tangent and rectifiers [Gol16]. Rectifiers are used most commonly.

The sigmoid activation function is a S-shaped function which transforms any value between the range 0 and 1. This is considered to be depreceated since other functions have been giving better results empirically.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

The hyperbolic tangent function is also a S-shaped function, but it transforms any value between the range -1 and 1.

$$tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

The Rectifier activation function clips values lesser than 0 and it performs faster and better than sigmoid and hybolic tangent functions.

$$ReLU(x) = max(0, x)$$

The output layer consists 2 neurons and each will take 3 inputs from hidden layer as shown in equation 3. If it has no outgoing lines then it will be used as final output. The output layer can use a transformation function such as softmax to convert values to represent a discrete probability distribution. In this figure, 2 neurons are used to refer to 2 labels and this system classifies the input into one of those labels.

$$y = y_1, y_2..., y_k$$
$$s_i = softmax(y_i) = \frac{e_i^y}{(\sum_{j=1}^{k} e^{y_j})}$$

Training is an essential part of learning and like many supervised algorithms, a loss function is used to compute the error for the predicted output against the actual output. Some of the loss functions that could be used are hinge loss (binary and multiclass), log loss, categorical cross-entropy loss etc [Gol16].

$$L_{hinge(binary)}(\hat{y}, y) = max(0, 1 - y.\hat{y})$$

$$L_{cross-entropy}(\hat{y}, y) = -\sum_i y_i.log(\hat{y}_i)$$
$$L_{cross-entropy(hardclassification)}(\hat{y}, y) = -log(\hat{y}_t)$$

The gradient of the errors is calculated and propagated back to compute with respect to weights and bias. The values of the weights and bias are adjusted with respect to the gradient and a learning parameter. Typically a random batch of inputs is selected and a forward pass is carried out which involves multiplying weights, adding bias and applying a activation function on top of it to predict outputs. The average loss is computed for that batch and the parameters are adjusted accordingly. This optimization technique is called stochastic gradient descent [Bot12]. A number of extensions exists, such as Nesterov Momentum [SMDH13] or AdaGrad [DHS11]. The overfitting in neural networks can be minimized by using regularization techniques such as L_2 regularization and dropout [HSK+12]. The L_2 regularization works by adding a penalty term equal to sum of the squares of all the parameters in the network to the loss function which is being minimized. The dropout works by randomly ignoring half of the neurons in every layer in each batch and corrects the error only using the parameters of other half of neurons. This helps to prevent the network from relying on only specific weights.

Feedforward networks work very well on structured input data but, in case of text data, the input is sequential. Techniques such as continuous bag of words [MCCD13] can be used to convert sequential input into fixed length but it will lose the order of the text information which is important. Convolutional neural networks (CNN) [Ben97] are good at capturing the local characteristics of data irrespective of its position. In this, a nonlinear function is applied to every k-word sliding window and captures the important characteristics of the word in that window. All the important characteristics from each window are combined by either taking maximum or average value from each window. This captures the important characteristics of a sentence irrespective of its location. However, because of the nature of CNNs they fail to recognize patterns that are far apart in the sequence.

Recurrent neural networks (RNN) accept sequential inputs and are often able to extract patterns over long distances [Elm]. RNN takes input as an ordered list of input vectors such as $x_{i:j}$ with initial state vector h_0 and returns an ordered list of state vectors $h_1,...,h_n$ as well as an ordered list of output vectors $o_1,...,o_n$. At time step t, a RNN takes as input a state vector h_{t-1}, an input vector x_t and outputs a new state vector h_t as shown in the figure 2. The outputted state vector is used as input state vector at the next time step. The same weights for input, state, and output vectors are used in each time step.

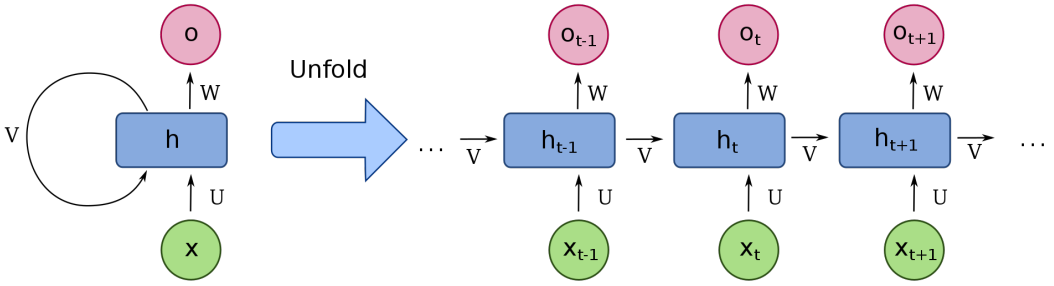

Figure 2: A basic example of RNN architecture [Del13].

$$\begin{aligned}
\mathrm{RNN}(s_0, x_{1:n}) &= s_{1:n}, y_{1:n} \\
s_i &= R(s_{i-1}, x_i) \\
y_i &= O(s_i) \\
x_i \in \mathbb{R}^{d_{in}}, y_i \in \mathbb{R}^{d_{out}}, s_i &\in \mathbb{R}^{f(d_{out})}
\end{aligned}$$

To train a RNN, the network is unrolled for the given input sequence and the loss function is used to compute the gradient of error with respect to parameters involved in every time step by propagating backwards through time. After that the parameters are adjusted to reduce the error in estimation[Wer90]. While training, the error gradients might vanish or explode especially when dealing with RNNs. The gradient explosion can be handled by clipping the gradient when it goes beyond the threshold. LSTM networks [HS97] solves the vanishing gradient problem by introducing memory cells which remembers gradients across time steps. These memory cells are controlled by gating components which decides at every time step, on what parts of the hidden state should be carried over and what parts of new input should be included.

$$\begin{aligned}
s_j &= R_{LSTM}(s_{j-1}, x_j) = [c_j; h_j] \\
c_j &= c_{j-1} \odot f + g \odot i \\
h_j &= tanh(c_j) \odot o \\
i &= \sigma(x_j W^{xi} + h_{j-1} W^{hi}) \\
f &= \sigma(x_j W^{xf} + h_{j-1} W^{hf}) \\
o &= \sigma(x_j W^{xo} + h_{j-1} W^{ho}) \\
g &= \sigma(x_j W^{xg} + h_{j-1} W^{hg})
\end{aligned}$$

$$y_j = O_{LSTM}(s_j) = h_j$$

$$s_j \in \mathbb{R}^{2.d_h}, x_i \in \mathbb{R}^{d_x}, c_j, h_j, i, f, o, g \in \mathbb{R}^d_h, W^{xo} \in \mathbb{R}^{d_x * d_h}, W^{ho} \in \mathbb{R}^{d_h * d_h}$$

4 Approach

You should rewrite this complete section. Start with the general architecture diagram, that we talked about in the last meeting (i.e. a black box taking a sentence and a context as input and outputting a binary output). Then discuss the different context definitions that we plan to compare. In the following roughly scetch out the planned architecutre of the model (i.e. RNN for encoding the input sentence, RNN for encoding the context sentence(s) (when applicable) and attention mechanism to reduce to singular vector followed by feed-forward net for prediction). At the end discuss training, i.e. how negative example sentences are generated from positive ones, and that you will need to compare several hyperparameter configurations.

1. General Architecutre diagram - sentence and context giving a binary output

2. How different models of context are available - None, 10 relevant wikipedia sentences, 100 triples of knowlege graph, knowledge graph as it is, POS + Named Entity + dependency graph
3. Get into the details of how the sentence and context are going to be modeled, attention mechanism to reduce to a sngular vector followed by feed forward network for prediction
4. Talk about training - how negative sentences are generated from the positive ones and also compare several hyper parameter configurations

We will use Recurrent neural networks in the master thesis extensively since they are good with sequential data. TensorFlow[10], a python library for building intensive numerical computation applications and deploy it in multiple platforms such as CPU, GPU etc, will be used to develop RNN architecture. Wikipedia consists of a lot of articles; each article consists of a lot of sentences. Each sentence could be fed as input to the neural network. The different configurations of the neural network will be used and the performance of each will be compared against the other. Some of the configurations of the neural network are

1. Different layers
2. Learning parameter
3. Batch size
4. Activation functions such as Rectified Linear units (ReLu), hyperbolic tangent (tanh) etc;
5. Epoch number (how long the training should happen)
6. One hot vector vs word embeddings
7. Single/Multiple layer LSTM

The dataset will be split and used for training, validation, and testing of the system. A general rule of thumb is to divide the dataset into 60-20-20 for training, validation, and testing respectively. But if the datasets are in order of millions then it is good enough to have approximately 10000 entries each for validation and testing. The validation dataset is used to test different parameter configurations and select the optimal values.

It is important to do pre-processing on the dataset which involves a series of steps.

1. removal of common words such as articles for e.g a, the etc;
2. lemmatization of verbs which involves bringing the conjugated verbs into its root form

Since we use Wikipedia as a proxy for authentic information, each sentence in the Wikipedia is considered as true news. There is hardly any false news present. For training to be successful to build a good fake news detector, it should be ensured that more or less equal amount of data should be present for both true and fake news. The construction of false news is tricky and we will use different natural language processing (NLP) methods and compare which one is giving good results. Spacy[11], a python library to do NLP tasks will be used for extracting parts of speech, named entity and dependency relations. Some techniques are

[10] https://www.tensorflow.org/
[11] https://www.spacy.io

1. Swap the words randomly in each and every sentence extracted from Wikipedia
2. Replace the random words picked in a sentence with a random word from the corpus
3. Replace the words in a sentence based on their parts of speech. In this method, a vocabulary of words is maintained for each Parts of Speech section and randomly one of it is chosen
4. Replace the words in each sentence with their opposites extracted from WordNet[12]
5. Extract the dependency tree for each sentence and select an appropriate one from other sentences to replace it

The above methods involve modeling the content of Wikipedia in such a way that the system can understand it better. These procedures are completely automatic in building the training examples and none of the features are manually crafted. The RNN and different configurations of the system along with unique methods in constructing the training data will be able to understand Wikipedia better. This will be an improved semantic understanding of Wikipedia and by using this fake news detection can be carried out in a platform agnostic manner.

5 Evaluation

The evaluation can be done both internally and externally. The system can be evaluated based on how well it is able to understand Wikipedia. This is done by inputting random sentences or the sentences taken from Wikipedia and check whether it is present or not.

In addition to that, the system can be evaluated as a fake news detector. There are two ways by which the test dataset can be prepared.

1. There are two comprehensive open datasets available. One of it is published in Kaggle[13] which consists of structured data of 13K size. The LIAR dataset is published by Wang [Wan17] which is mined from politifact.com and it consists approximately 13K entries.
2. Fake news can be obtained directly from facts checking websites such as snopes.com, politifact.com, etc., by crawling or using the API's provided.

The baseline system is a very simple RNN and the improvements are added on top of it. The following are the factors which are used to compare the baseline with the improved system

1. How good the system is able to classify the non-wikipedia news / fake news?
2. How fast the model is build up?

The goodness of the system can be measured by metrics such as Precision, Recall, F1 measure, and Accuracy. The F1 measure is preferred since it includes both false positive and false negatives. The speed of the system can be measured iteratively by calculating the results of the loss function. In this master thesis, the focus is mainly on improving the accuracy of the system and the speed of the system can be improved by using good hardware.

[12] https://wordnet.princeton.edu/

[13] https://www.kaggle.com/mrisdal/fake-news

6 Organizational matters

Duration of work:	01-July-2018 – 31-Dec-2018
Candidate:	Kandhasamy Rajasekaran
E-Mail:	kandhasamy@uni-koblenz.de
Student number:	216100855
Primary supervisor:	Prof. Dr. Steffen Staab
Supervisor:	Dr. Chandan Kumar
Secondary supervisor:	Lukas Schmelzeisen

7 Time schedule

- Introduction and Literature: 01-May-2018 – 30-June-2018
- Initial phase: 01-July-2018 – 15-Sep-2018
  - Prototyping: 01-July-2018 – 30-July-2018
  - ML pipeline implementation: 01-Aug-2018 – 15-Aug-2018
  - Baseline implementation: 16-Aug-2018 – 30-Aug-2018
  - Testing and refining: 01-Sep-2018 – 15-Sep-2018
- Development phase: 16-Sep-2018 – 30-Dec-2018
  - RNN with different configuration: 16-Sep-2018 – 30-Oct-2018
  - Comprehend benchmark results: 16-Oct-2018 – 21-Oct-2018
  - Analyse and figure out improvements: 22-Oct-2018 – 30-Oct-2018
  - Improvisation using NLP techniques: 01-Nov-2018 – 30-Dec-2018
  - Comprehend benchmark results: 16-Dec-2018 – 30-Dec-2018
- Final phase: 01-Jan-2019 – 01-Feb-2019
  - Comprehend Benchamark results: 01-Jan-2019 – 07-Jan-2019
  - Revision: 08-Jan-2019 – 22-Jan-2019
  - Thesis report: 01-Jan-2019 – 30-Jan-2019

A meeting with Lukas Schmelzeisen will happen approximately once in two weeks to discuss about the progress made and set the targets and milestones for subsequent weeks.

References

[AG17] Hunt Allcott and Matthew Gentzkow. Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives*, 2017.

[Ben97] Y Bengio. Convolutional Networks for Images, Speech, and Time-Series Parsing View project Oracle Performance for Visual Captioning View project. 1997.

[Bot12] Léon Bottou. Stochastic Gradient Descent Tricks. In Geneviève B. Montavon Grégoire and Orr and Müller Klaus-Robert, editors, *Neural Networks: Tricks of the Trade: Second Edition*, pages 421–436. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[Com06] Wikipedia Commons. Feed forward network, 2006. `https://en.wikipedia.org/wiki/File:Feed_forward_neural_net.gif`; CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0).

[CSR+15] Giovanni Luca Ciampaglia, Prashant Shiralkar, Luis M. Rocha, Johan Bollen, Filippo Menczer, and Alessandro Flammini. Computational fact checking from knowledge networks. *PLoS ONE*, 2015.

[Del13] Francois Deloche. Recurrent neural network unfold, 2013. [Online; accessed April 27, 2013; `https://commons.wikimedia.org/wiki/File:Recurrent_neural_network_unfold.svg`; CC BY-SA 4.0 (https://creativecommons.org/licenses/by-sa/4.0).

[DHS11] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[Elm] Jeffrey L Elman. Finding Structure in Time. *COGNITIVE SCIENCE*, 14(1):179–21.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. www.deeplearningbook.org.

[Gol16] Yoav Goldberg. A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[HL08] Alexander Halavais and Derek Lackaff. An analysis of topical coverage of Wikipedia. *Journal of Computer-Mediated Communication*, 13(2):429–440, 2008.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[HSK+12] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. 2012.

[KCJ+13] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent features of rumor propagation in online social media. In *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2013.

[LBB+18] David M. J. Lazer, Matthew A. Baum, Yochai Benkler, Adam J. Berinsky, Kelly M. Greenhill, Filippo Menczer, Miriam J. Metzger, Brendan Nyhan, Gordon Pennycook, David Rothschild, Michael Schudson, Steven A. Sloman, Cass R. Sunstein, Emily A. Thorson, Duncan J. Watts, and Jonathan L. Zittrain. The science of fake news. *Science*, 359(6380):1094–1096, 2018.

[LNL+15] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time Rumor Debunking on Twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management - CIKM '15*, 2015.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[MGM+] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. Detecting Rumors from Microblogs with Recurrent Neural Networks.

[MGW+15] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. 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 - CIKM '15*, 2015.

[MP43] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 1943.

[PCL+07] Reid Priedhorsky, Jilin Chen, Shyong (Tony) K. Lam, Katherine Panciera, Loren Terveen, and John Riedl. Creating, destroying, and restoring value in wikipedia. In *Proceedings of the 2007 international ACM conference on Conference on supporting group work - GROUP '07*, 2007.

[SKP97] Daniel Svozil, Vladimir Kvasnieka, and Jie Pospichal. Chemometrics and intelligent laboratory systems Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, 39:43–62, 1997.

[SMDH13] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, (2010):8609–8613, 2013.

[VRA18] Soroush Vosoughi, Deb Roy, and Sinan Aral. The spread of true and false news online. *Science*, 359(6380):1146–1151, 2018.

[Wal05] Jimmy Wales. Internet encyclopaedias go head to head, 2005.

[Wan17] William Yang Wang. "Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection. 2017.

[Wer90] Paul J. Werbos. Backpropagation Through Time: What It Does and How to Do It. *Proceedings of the IEEE*, 1990.

[ZRM15] Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 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, Republic and Canton of Geneva, Switzerland, 2015. International World Wide Web Conferences Steering Committee.

8 Signatures

Kandhasamy Rajasekaran	Prof. Dr. Steffen Staab
Dr. Chandan Kumar	Lukas Schmelzeisen

9 Declaration of Authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

I am aware that the thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the thesis as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future theses submitted. Further rights of reproduction and usage, however, are not granted here.

This paper was not previously presented to another examination board and has not been published.

Koblenz, on September 4, 2018

_____

Kandhasamy Rajasekaran