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

16. 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 truth. In this thesis work, the information in Wikipedia will be modeled 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 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 by the domain experts to refer its authenticity. The main strategy to handle fake news is to check against a reliable source of information and claim its integrity. The domain experts are believed to have good knowledge on the subject of news. 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]. 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 each of the domains such as music, history, geography, philosophy, science, technology, and literature constitutes at least 5 percent of total articles in English Wikipedia. Also, it is found that, for all domains, the average size

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

of an article is at least 20 kilobytes (excluding images) and the average edits of an article are at least 8. Although it can be edited by anyone, an investigation carried out by Nature[3], reveals that the quality of content is similar to another established 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 a claim information in Wikipedia will be used as a proxy for classifying that claim to be true or fake. Several deep learning techniques [GBC16] will be used to model the information in Wikipedia and these models are evaluated based on how well it can classify fake news. In addition to that, along with the claim input, the information related to a claim will be extracted and used as a context. This context information will either be extracted from Wikipedia or different knowledge sources available online.Our hypothesis is that a model, trained using both input and relevant context information will perform better as a fake news classifier than the model trained only with input.

The "Related work" section consists of a brief summary of different researches done in the past which are relevant to fake news classification using semantics and platform based approaches. The "Background study" section gives all the information required to understand the different approaches that will be used in this thesis. It starts with an introduction of neural networks, the different components involved in it and then to different architectures to model sequence data effectively, the training process involved to optimize model parameter values and the attention mechanism technique to model both input and context information. The "Approach" section comprises the overall work to be done in this thesis which includes different techniques of modeling the Wikipedia and several techniques to build good negative training examples. The "Evaluation" section describes what factors will be considered for testing how good the classifier is and the standard test datasets that will be used to benchmark the performance of different techniques.

2 Related work

There has been much effort to detect fake news in microblogging platforms such as Twitter[5]. Most of these works [LNL+15] [MGW+15]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. 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.

However, the mentioned works only used comparatively smaller datasets. Wang [Wan17] curated a dataset consisting of approximately 13,000 short statements by mining politfact.com covering a decade of information. In this approach, six machine learning models have been built ranging from logistic regression to a convolutional neural network. Along with the text data, metadata such as the

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

speaker, subject, speaker history are also included. The convolutional neural network model used to capture surface level linguistics along with metadata performed better than other models.

Jing Ma et al.[MGM+] efforts have been focused on building a recurrent neural network (RNN) to detect rumors from microblogs such as Twitter and Weibo[6] effectively. The training dataset is obtained by using keywords from fake and true news of debunking services such as Snopes[7] and the Sina community management center[8]. These keywords are used in search APIs of microblogs and the found results are labeled as true or fake 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 are used to capture textual representations and the temporal patterns associated with those posts. An accuracy of at least 80% was achieved by architectures such as RNN, RNN with one layer of LSTM or GRU and RNN with two layers of GRU on both the datasets. This method avoids handcrafted feature engineering efforts which are biased and time-consuming by only considering the text of posts. It produced better results with datasets from Twitter and Sina Weibo than all of the traditional machine learning methods it was compared against. RNNs with two layers of GRU captured long distance dependencies very well and achieved 88% and 91% accuracy on twitter and weibo dataset respectively. An accuracy of more than 80% was achieved within 12 hours, whereas the average time taken by debunking services were more than 50 hours.

However, the mentioned work by Jing Ma et al. is only for microblogging platforms. And also the features specified in one platform will be different from another 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 might improve the chance of achieving same results across platforms.

Ciampaglia et al. [CSR+15] used DBPedia[9], a knowledge graph, for checking computationally whether a given information is factual or not. This work uses some datasets of DBPedia which represent infobox section in Wikipedia. Therefore, it contains only non-controversial and factual information because it stems from human collaboration. 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, that is the number 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 non-generic nodes. This approach exploits the indirect connections to a great extent with distance constraints in a knowledge graph. The approach was evaluated against four datasets such as directors and their movies, US presidents and their spouses, US states and its capital and countries and its capital. The statements were generated of the form 'd directed m', 'p was married to s' and 'c is the capital of r' where d is a director, m is a movie, p is a US president, s is a spouse of US president, c is a city and r is a country or US state. The approach was assigning better scores to true statements than the false statements with probabilities 0.95, 0.98, 0.61 and 0.95 for director, spouse, US capital and world capital datasets respectively.

The mentioned work by Ciampaglia et al. is a good step towards an automated fact checker system using only the semantics of data. However, the methodology implemented is primitive and can work only for statements which have a simple relationship between the subject and object.

[6] https://www.weibo.com
[7] https://www.snopes.com/
[8] https://www.sina.com.cn/
[9] https://wiki.dbpedia.org/

3 Background Study

In this thesis work, the information in Wikipedia will be learned by developing an advanced system which consists of different neural network architectures such as feed-forward and recurrent neural networks. All the necessary information required to understand the different components of the system is explained briefly in this section. Every sentence of Wikipedia would be fed and the system will be trained to remember it. Several other techniques will be tried to improve the efficiency in learning. One of it is extracting information related to an input sentence, inputting it along with sentence and training the system. This is achieved by using attention mechanism in neural networks.

Neural networks are state of the art models to build learning systems. 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.

Feed-Forward 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

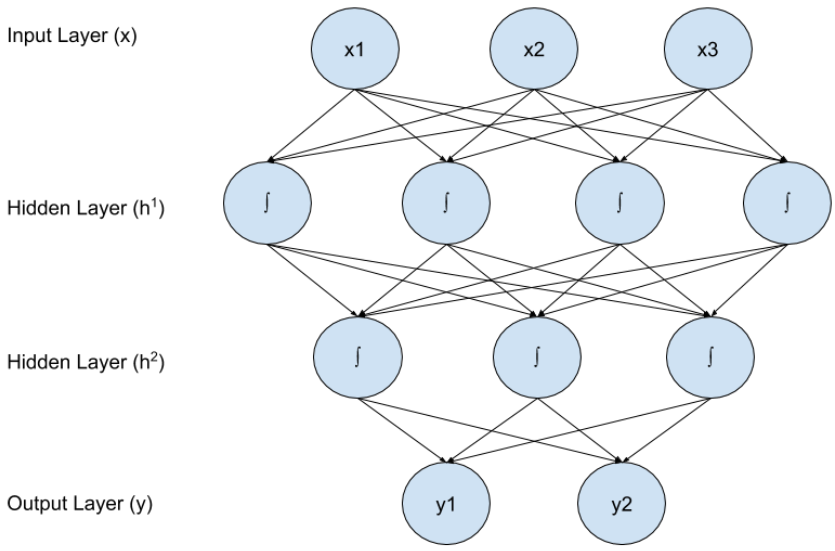

Figure 1: A Feed-Forward neural network.

There are 4 layers in 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 first hidden layer consists of 4 neurons, whereas each neuron takes 3 inputs from input layer. Each input is multiplied by a unique weight variable and added together. Finally, the output is added with a bias variable and will be passed to a non-linear activation function as shown in equation 1. The same process is carried out for the second hidden layer except that it has 3 neurons and each neuron will take 4 inputs from the first hidden layer. The output layer consists of 2 neurons and each will take 3 inputs from second hidden layer as shown in equation 3. Each of the

two neurons represents a class label.

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

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

$$\mathrm{NN}_{\mathrm{MLP2}}(x) = y = g^3(h^2W^3 + b^3) \tag{3}$$

$$x \in \mathbb{R}^{d_{in}}, W^1 \in \mathbb{R}^{d_{in} \times d_1}, b^1 \in \mathbb{R}^{d_1}, W^2 \in \mathbb{R}^{d_1 \times d_2}, b^2 \in \mathbb{R}^{d_2}, W^3 \in \mathbb{R}^{d_2 \times d_{out}}, b^3 \in \mathbb{R}^{d_{out}}$$

Here W^1, W^2, W^3, and b^1, b^2 and b^3 are matrices and bias vectors for first, second and third linear transforms respectively. The functions g^1, g^2 and g^3 can be linear or non-linear. With respect to figure 1, the values of d_{in}, d_1, d_2 and d_{out} are 3, 4, 3 and 2 respectively.

The activation functions 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].

The sigmoid activation function is a S-shaped function which transforms any value between the range 0 and 1. This is not used in internal layers of neural networks 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 hyperbolic tangent functions.

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

Instead of an activation function, the function in output layer g^3 can be a transformation function such as softmax to convert values to represent a discrete probability distribution. Each of the converted values will be between 0 and 1 and sum of all of them will be 1.

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

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. The gradient of the errors is

calculated with respect to each weight and bias variable by propagating backward using chain rule of differentiation. 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 an activation function 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]. Some loss functions that exist are hinge loss (binary and multiclass), log loss and categorical cross-entropy loss [Gol16].

The categorical cross-entropy loss is used when predicted output refers to a probability distribution. Let y= y_1, y_2,..., y_n be representing a multinomial distribution over the labels 1,2...n and let $\hat{y}_1$, $\hat{y}_2$,..., $\hat{y}_n$ be the network's output which is transformed by a softmax function. The categorical cross-entropy loss measures the difference between the true label distribution y and the predicted label distribution $\hat{y}$.

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

For hard classification, y is a one-hot vector representing the true class. Here t is the correct class assignment. Training will attempt to set the correct class t to 1 which inturn will decrease the other class assignment to 0.

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

The overfitting in neural networks will cause the trained system to perform well only for trained data but not on the test data. This 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.

In case of text data, the input is arbitrary and sequential where the ordering of words is important. Techniques such as continuous bag of words [MCCD13] can be used with feed-forward networks to convert sequential input into fixed length but it will lose the order of the text information. 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 $x_{1:n}$ 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.

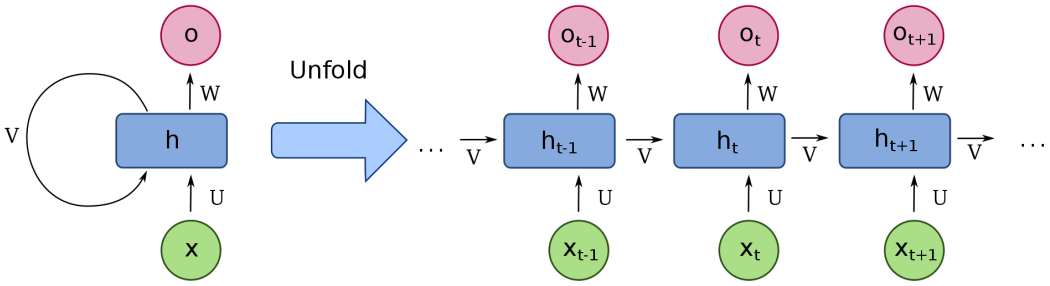

Figure 2: A basic example of RNN architecture [Del13].

$$\begin{aligned} \text{RNN}(h_0, x_{1:n}) &= h_{1:n}, o_{1:n} \\ h_i &= g^1(h_{i-1}V + x_iU + b^1) \\ o_i &= g^2(h_iW + b^2) \end{aligned}$$

$$x_i \in \mathbb{R}^{d_x}, h_i \in \mathbb{R}^{d_h}, o_i \in \mathbb{R}^{d_o}, U \in \mathbb{R}^{d_x \times d_h}, V \in \mathbb{R}^{d_h \times d_h}, W \in \mathbb{R}^{d_h \times d_o}$$

Here the functions g^1 and g^2 are non-linear; W, V, and U are matrices and b^1, b^2 are bias vectors.

To train a RNN, the network is unrolled for a 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 backward through time. After that, the parameters are adjusted to reduce the error in prediction[Wer90]. While training, the error gradients might vanish or explode especially when dealing with long sentences in RNNs. The gradient explosion can be handled by clipping the gradient when it goes beyond the threshold. LSTM networks [HS97] solve the vanishing gradient problem by introducing memory cells which are controlled by gating components. These gating components decide at each time step, what parts of the hidden state should be forgotten and what parts of new input should be included into the memory cells. These memory cells are involved in the computation of hidden states, which in turn are used to compute the output states. This captures the dependency between words even though separated by a long distance by configuring the gates accordingly.

RNN computes the state of current word x_{i} only based on the words in the past such as $x_{1:\text{i-1}}$. However, the following words $x_{\text{i+1:n}}$ will also be useful in computing the hidden state regarding the current word. The Bidirectional RNN (biRNN) (Schuster and Paliwal, 1997) solves the problem by having two different RNNs. The first RNN is fed with the input sequence $x_{1:\text{n}}$ and the second RNN is fed with input sequence in reverse. The hidden state representation h_{i} is then composed of both the forward and backward states. Each state representation consists of the token information along with sentential context from both directions and this is better than uni-directional RNN.

The mentioned approaches take only Wikipedia sentence as an input. This can be extended further by feeding information which gives context about the input sentence. This extra information might enable the system to perform efficiently in classifying whether a news is fake or not. Attention mechanism in neural networks is used to take two inputs and convert it into one common representation. This is done by configuring the level of attention that needs to be given to different parts of each input and combine them to form a single output. Most often the inputs will be matrices and the output will be a vector.

Bidirectional RNNs can be used to convert both input and context information into matrices. The fixed output vector is then fed into a deep feed-forward neural network to predict a class. Attention mechanism with RNN gives state of the art results in many NLP applications such as Natural Language Inference [PTDU16], Machine Translation[BCB14] and Document classification[YYD+16].

4 Approach

Recurrent neural networks will be used in this master thesis extensively since they perform well with sequential data. TensorFlow[10] is a python library for building intensive numerical computation applications and deploy them in multiple platforms such as CPU and GPU. TensorFlow has built-in libraries to develop different neural network architectures. Wikipedia consists of a lot of articles; each article consists of a lot of sentences. Each sentence will be fed as input to the neural network. A general architecture of the system is as shown in figure 3

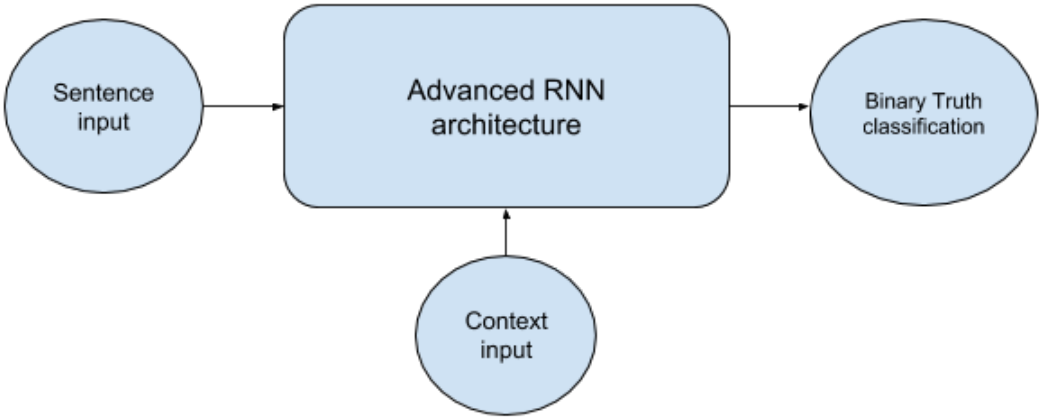

Figure 3: Broad Architecture

An advanced RNN architecture takes in either one or two inputs such as sentence in Wikipedia and context information. The context can be empty. The system processes the input and produces a binary output representing true or fake news. The outputs would contain a probability distribution of how likely it is true or fake news.

The inputs, in general, will be pre-processed to extract tokens, remove common words such as articles, prepositions and perform lemmatization which is converting the words to its root form. Low dimensional and dense word embeddings will be used to represent the inputs instead of one hot vectors. Already available word embeddings such as Word2vec[MCCD] and GloVe[11] will be used to represent the inputs.

The reason behind using context input is that the RNN system would be better equipped with information to make a decision whether it is a true or fake news. And also that, the problem of fake news classification can be seen as a Natural Language Inference [PTDU16] problem, which is about finding entailment or contradictory relationship between two sentences such as hypothesis and premise. Here,

[10] https://www.tensorflow.org/
[11] https://nlp.stanford.edu/projects/glove/

the context input is the premise and sentence input is the hypothesis. If there is an entailment between sentence input and context then it is true news otherwise false.

There are many ways by which context input can be modeled and inputted to the systems. Extracting the context as relevant to sentence input can be challenging and a suitable information retrieval method will be selected for it.

1. The context can be empty. The system would be trained to understand Wikipedia and use this information alone to make a decision.
2. Top 10 Wikipedia articles relevant to the sentence will be passed as context input. The Wikipedia articles will be long and so will increase the size of context input. The system may face long distance dependency issues and a suitable hyper-parameter selection is critical. Since both the sentence and context input are text from Wikipedia, the system might find it easy to align one with respect to other and find patterns among them.
3. Nearly 100 triples of knowledge graph will be passed as context input. Resource Description Framework (RDF) triples of DBPedia can be used. Extracting such information from DBPedia will be relatively easier. The structure of context input will be simple and straightforward and the length of context input will be shorter than using Wikipedia articles.
4. Information such as parts of speech, named entity and dependency graph of a sentence will be passed as context input. These metadata will bring more clarity about the sentence structure, which will help the system to extract patterns efficiently.

A detailed architecture of an advanced RNN system is shown in the figure 4. The sentence input and context input will be encoded into a rich and convenient representation separately. Bidirectional RNNs can be used which will take a sequential input and convert it into a matrix. The number of rows of the matrix will be equal to number of tokens in a sentence and number of columns can be controlled. Each row in the matrix will represent a token with sentential context. Both the sentence and context representation are then combined to form an efficient common representation. The arbitrariness of both sentence and context representation will be controlled and converted into a representation of fixed length. Attention modules can be used which will figure out how much attention needs to be given to both the representations to produce a common representation. It usually takes two matrices and outputs a fixed vector. But the attention mechanism can be a pure reduction process, where it takes only one matrix and converts into a fixed vector. This can be used when the context input is empty. The predictor takes in a fixed vector as input and predicts which one of the binary class label(true or fake) is most likely the final output. A feed-forward neural network will be a good choice to be used as a predictor.

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 different natural language processing (NLP) methods will be used to generate false news. Spacy[12], a python library for NLP tasks will be used for extracting parts of speech, named entity and dependency relations. Some techniques are

[12]https://www.spacy.io

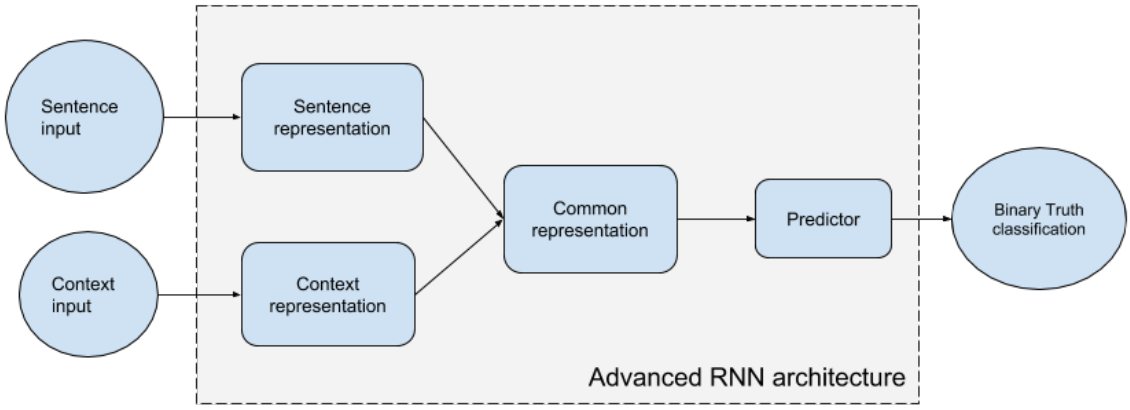

Figure 4: Detailed Architecture

1. Swap the words randomly in each and every sentence extracted from Wikipedia. The RNN models used will remember the sequence information and so a randomly shuffled sentence will be considered as a different sentence by the system. However, the syntax and semantics of the sentence created will be poor. This method is primitive but it will serve as a good baseline.

2. Replace the random words picked in a sentence with a random word from the corpus. This method would retain some of the words as it is and changing only few words will bring a subtle change in sentence. This is similar to random swapping but the syntactic structure of the sentence will be preserved to some extent.

3. Replace all or randomly picked words in a sentence based on their parts of speech. In this method, all the words in the corpus will be tagged with a parts of speech label. This information will be used and randomly a word will be picked from the corpus which has the same parts of speech as the word to be replaced. This method will preserve the syntactic structure of the sentence better than the previous methods.

4. Replace the words in each sentence with their opposites extracted from WordNet[13]. This method will not only preserve the syntax but also likely to have right semantics. The sentence created will more likely to have a meaning which is opposite of the original sentence.

5. Extract the dependency tree for each sentence and select an appropriate one from other sentences to replace it. The dependency relationship will contain information about which words are dependent on which other words. Replacing based on this similarity may create better examples.

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 try out different hyper-parameter configurations and select the optimal values for each. The hyper-parameters of the advanced neural network system that will be configured are number of layers, learning parameter, batch size, activation functions and epoch number(number of times the training dataset should be iterated).

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

The proposed methods involve modeling the content of Wikipedia in different ways such that the system can understand it better. These procedures are completely automatic in building the training examples and none of the features will be manually crafted. The different configurations of the system along with unique methods in constructing the training data is expected to understand Wikipedia better and the fake news detection can be carried out in a platform agnostic manner.

5 Evaluation

The system will be evaluated as a fake news classifier. The test data should be different from the training or validation data which is used by the system. There are two ways by which the test dataset can be prepared.

1. There are two comprehensive open datasets on fake news detection available. One of it is published in Kaggle[14] which consists of structured data of 13,000 rows and 20 columns. Some of the important information in this dataset are title, text and spam score. The LIAR dataset is published by Wang [Wan17] which is mined from politifact.com and it consists of 12,836 short statements. In addition to that, it also contains information about context, labels with 6 classes and justification for the label classification.

2. Fake news can be obtained directly from fact checking websites such as snopes.com and politifact.com, by crawling or using the APIs provided.

The mentioned datasets might be too diverse to compare the performance of the system at the beginning. It might require training from the whole of Wikipedia. Research work from Ciampaglia et al. [CSR+15] used many simple datasets to evaluate network analysis techniques on a knowledge graph to build fake news classfier. The datasets are US politicians and their party affiliation, directors and their movies, US president and their spouses, US states and their capitals and world countries and their capitals. It also involves a dataset from Google Relation Extraction Corpus (GREC) which is about education degrees and institutional affiliations of people. These datasets will help to compare the performance of fake news classifier in specific domains of knowledge against their achieved results.

In addition to that, the system can be evaluated based on how well it is able to understand Wikipedia. We use the presence and absence of a information in Wikipedia as a proxy for labeling that information as true or fake news. So, it is good to evaluate our system based on how well it is able to classify whether a given information is present in Wikipedia or not. This is done by inputting either random sentences or the sentences taken from Wikipedia and check whether it is present or not. If the system performs very well in this test and fails to perform well with the mentioned standard datasets for fake news classification then it might help to gain insights on how good the coverage of news in Wikipedia is.

The baseline system involves only RNN and the improvements that will be added are RNN with LSTM blocks, bidirectional RNNs, and inclusion of context input along with sentence input. The following factors 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?

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

The quality 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. 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.

6 Organizational matters

Duration of work:	01-Aug-2018 – 31-Jan-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-Aug-2018 – 15-Oct-2018
  - Prototyping: 01-Aug-2018 – 30-Aug-2018
  - ML pipeline implementation: 01-Sep-2018 – 15-Sep-2018
  - Baseline implementation: 15-Sep-2018 – 30-Sep-2018
  - Testing and refining: 01-Oct-2018 – 15-Oct-2018
- Development phase: 16-Oct-2018 – 30-Dec-2018
  - RNN with different configuration: 16-Oct-2018 – 30-Oct-2018
  - Comprehend benchmark results: 16-Oct-2018 – 30-Oct-2018
  - Analyse and figure out improvements: 22-Oct-2018 – 30-Oct-2018
  - Improvisation using NLP techniques: 01-Nov-2018 – 30-Nov-2018
  - Attention mechanism implementation: 01-Dec-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.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. pages 1–15, 2014.

[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.

[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.

[MCCD] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality arXiv : 1310 . 4546v1 [cs . CL] 16 Oct 2013. pages 1–9.

[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.

[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.

[PTDU16] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A Decomposable Attention Model for Natural Language Inference. 2016.

[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.

[YYD+16] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical Attention Networks for Document Classification. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.

[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 16, 2018

_____

Kandhasamy Rajasekaran