Neural Stance Detectors for Fake News Challenge

Qi Zeng
Department of Physics
Stanford University
qizeng@stanford.edu

Quan Zhou
Department of Physics
Stanford University
quanz@stanford.edu

Shanshan Xu
Department of Physics
Stanford University
xuss@stanford.edu

Abstract

Fake news pose serious threat to our society nowadays, particularly due to its wide spread through social networks. While human fact checkers cannot handle such tremendous information online in real time, AI technology can be leveraged to automate fake news detection. The first step leading to a sophisticated fake news detection system is the stance detection between statement and body text. In this work, we analyze the dataset from *Fake News Challenge (FNC1)* and explore several neural stance detection models based on the ideas of natural language inference and machine comprehension. Experiment results show that all neural network models can outperform the hand-crafted feature based system. By improving Attentive Reader with a full attention mechanism between body text and headline and implementing bilateral multi-perspective mathcing models, we are able to further bring up the performance and reach metric score close to 87%.

1 Introduction

New Media, including online newspapers and social media, not only enables people around the world easily receive news and share information in real-time, but also lets fake news and rumors spread quickly under no verification. These made-up stories can create a great deal of confusion in the public about the facts of current events, thus poses a serious challenge to the news industry as well as the whole of society. It is extremely difficult and even unrealistic to identify rumors and hoaxes in the information age relying only on the traditional human-based fact checkers, due to both the tremendous scale and the real-time nature. Artificial intelligence, specifically machine learning and natural language processing techniques, holds promise for assisting people to win the combat against fake news.

Although the full procedure for news veracity assessment is complicated and unwieldy even for trained experts, AI technology can be leveraged to significantly automating some parts of it. For example, an important building block in fact-checking systems is to understand what other organizations says about one certain topic. In this task, a computer is required to estimate the relative perspective of two given pieces of texts, specifically the stance of news body relative to a headline. While the stance detection of news shares a lot in common with various natural language processing tasks, such as natural language inference [1] and machine comprehension [2, 3], it also has its own distinctness in several aspects. Most natural language inference studies focus on the relation between two sentences of small or even moderate length, comparatively the body text of news can have several or tens of sentences, which makes the sequence quite longer than the headline. Machine

comprehension indeed deals with the scenario of short question and lengthy context, but it aims to point out the answer span rather than to evaluate the overall attitude toward the question.

The key in the task of stance detection is to find out good features to represent the relations between headline and body text. Hand-crafted features of headline-body pair, including bag of words and n-grams matching, have already been used to achieve moderate accuracy for stance classification with the dataset of Emergent [4]. Nonetheless, feature engineering requires a lot of linguistic expertise, and does not fully take advantage of the huge amount of data. Recently, neural-based encoder architectures are introduced to capture these relations [5]. In general, these neural approaches fall into two types of framework: the "Siamese" architecture which encodes headline and body text independently, and the conditional encoder which allows interactions between them in the encoding stage. Augenstein *et. al.* have shown that their bi-directional conditional encoding system can reach state-of-the-art results in SemEval 2016 Task 6 Twitter Stance Detection [5]. However, almost all the researches until now, studied the dataset of texts of moderate length at most [Twitter, Emergent], because of the limited availability of human-annotated data for real news headlines and bodies.

Recently, *Fake News Challenge* (*FNC1*) derives from Emergent a moderate-sized dataset of expert-labeled headline-body pairs for the task of stance detection. Body texts in this dataset can contain up to several thousands of tokens. One baseline model based on handed features is also provided, which achieves a performance of near 80% under a weighted score metric. In this paper, we analyze this dataset and investigate several neural stance detection systems for the task, based on ideas from natural language inference and machine comprehension. We explore a set of Bi-RNN encoders for feature extraction, and find that they are already able to beat the baseline. With appropriate connections between headline and body text, as well as large hidden memory in the RNN cell, this type of models can record a score of about 82%. We further incorporate attention mechanism [2, 3] beyond the vanilla Bi-RNN encoder, where each position in the body text attends either the final encoding (simple attention) or every time stamp of the headline (full attention). Attention brings significant improvements on simple encoders, one of our best attentive reader (full attention) scores near 87%, which indicates the local context matching nature of the current task. With this inspiration, we experiment the bilateral multiple perspective matching [6] with only max-pooling matching, which already reaches a robust score near 85%. More sophisticated matching components might hold promise for further improvement.

2 The Stance Detection Task

The stance detection task introduced by *FNC1* aims to identify perspectives of news body texts toward headlines. Specifically, given a pair of headline and body text, the goal is to predict whether the body text is related to the headline, and what the exact relationship is between them. Four different labels can be assigned to each headline-body pair: *Unrelated*, *Discuss*, *Agree* and *Disagree*. A typical example is illustrated in Figure 1. Typically, a headline can be one sentence or a combination of several short expressions, and a body text consists of several or even tens of sentences.

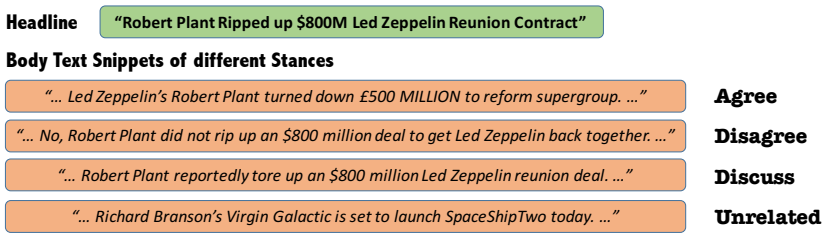

Figure 1: One example of body stances toward a headline

The dataset provided by *FNC1* is derived from Emergent, a digital journalism project for rumor debunking. There are 49972 headline-body pairs in total, with stances labeled by expert journalists. The populations of the four labels are imbalanced, with nearly three quarters belonging to the type *Unrelated*, which is reasonable for real news data but poses challenge for relationship identification. There is a severe overlap of either body text or headline among these pairs, as there are only 1683

distinct body texts and 1648 distinct headlines. Therefore shared body texts must be avoided in data split of training and test set, as suggest by the baseline model provided by the challenge. In our work, 10% of the body texts are selected as hold-out first, where all the pairs with these body texts constitute the test set. The remaining 90% body texts are randomly divided based on $8:1$ ratio every time. The corresponding two sets of pairs are used as training set and validation set respectively. Detailed data statistics is given in Table 1.

# Headline-body pairs			49972
# Headlines			1648
# Bodies			1683
# Bodies in test set			169
# Headline-body pairs in test set			5025
Average # tokens of headline			12.6
Average # tokens of body			427.5
Unrelated	*Discuss*	*Agree*	*Disagree*
73.1%	17.8%	7.4%	1.7%

Table 1: Statistics of *FNC1* dataset

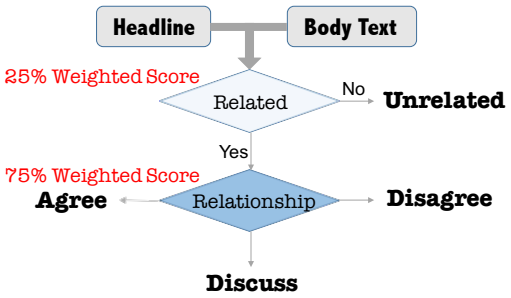

Figure 2: Score Metric for *FNC1*

FNC1 also provides a score metric for the task of stance detection. Since relationship identification is more important in real fake news detection, larger portion of score is given to the correct prediction of this part, while discrimination over related/unrelated is less weighted. More specifically, 25% weighted score is for correct prediction on relatedness, and another 75% weighted score is for correct prediction on relationship, as shown in Figure 2.

3 Methods

There are in general two approaches to the stance detection: hand-crafted feature engineering and neural networks. While the first approach largely depends on human experts with domain knowledge in NLP, the second one is trying to reveal the hidden structure between headlines and body texts using recurrent neural network structures.

The baseline method, provided by the *FNC1* organizer, is following the first approach. Hand-crafted features used by this method includes

- Fraction of overlapped words between headline and body
- Appearnce of refuting words in headline
- Polarity of both headline and body
- Counting of occurence of a token in headline in the body text (with and without stopwords included in title)
- Counting of occurence of an n-gram of the title in both the body text and paragrphs

And then these features are concatenated and fed into a GradientBoosting classifier.

For the neural network approaches, there are two components: feature extractor and classifier. The feature extractors, which would be described in details in Section 3.1, 3.2 and 3.3, are all RNN based, while the classifier is a simple feedforward neural network. For this task, since we believe the feature extractor is the key part, we put most of our focus on it, with the classifier network fixed. Due to the nature of text encoding, only bidirectional RNN is considered in feature extractor.

3.1 Bidirectional Encoder

This is the simplest RNN approach that we start with. In this approach, we encode both headline and body text with bidirectional recurrent neural networks, and the final states of RNN cells are concatenated as features for classifier network. Furthermore, three slightly different architectures are considered as following.

Unconditional Bidirectional Encoder. Headline and body text are encoded using separate bidirectional RNN cells with different weights. Both are initialized with hidden states all set to 0 before processing headline or body text independently. The final states of each RNN encoder are concatenated as feature for classifier.

Conditional Bidirectional Encoder. Headline and body text are encoded using separate bidirectional RNN cells with different weights. The headline is first processed. Then the RNN cells encoding body text is initialized with final states of the headline encoder before processing through body text. The final states of each RNN encoder are concatenated as feature for classifier.

Concatenated Bidirectional Encoder. Headline and body text are first concatenated together and then encoded by common RNN cells. The final states of this RNN encoder are concatenated as feature for classifier.

3.2 Attentive Reader

While the simple bidirectional encoder can build the contextual representation of headline and body, it is hard to identify the most relevant part in body text for a given headline. Therefore, we introduce the attention mechanism, as first proposed in [2]. Similar to the unconditional bidirecitonal encoder, we first build a contextual representation of each word in headline and body text using independent bidirectional RNN cells. Output vector of each body word (each time-stamp) is then compared to the headline in order to determine an attention weight for each word. Based on how we generate attention weight, two different models are explored as following.

Attentive Reader with simple attention. This is illustrated in the left of Figure 3. Each time-stamp in the RNN processing body text is compared with the final states of RNN cells processing headlines using a bilinear term, as proposed in [3], which is further normalized with a softmax nonlinearity to generate the attention weight for each word in body text:

$$\alpha_i = \text{softmax}_i \boldsymbol{q}^T W_s \tilde{\boldsymbol{p}}_i \tag{1}$$

where $W_s \in \mathbb{R}^{h \times h}$ is the bilinear term, $\boldsymbol{q}$ is the final states of headline RNN, $\tilde{\boldsymbol{p}}_i$ is the contextual embedding of body text and α_i is the attention weight.

Then we compute the weighted sum of each time-stamp vector and concatenated it with the final states of headline RNN as features for classifier. This last concatenation step is different from [3] since we find this to give a better performance.

Attentive Reader with full attention. This is illustrated in the right of Figure 3. The major difference from "simple attention" is that we calculate the attention weight by comparing time-stamp vector of body RNN to **each** time-stamp vector of headine RNN, instead of the final states. The maximum over all attention weights calculated for headline time-stamps would be taken as the final attention weight for each time-stamp of body text. We call this approach "full attention" since it is able to perform a comparison at a much finer granularity compared to the "simple attention".

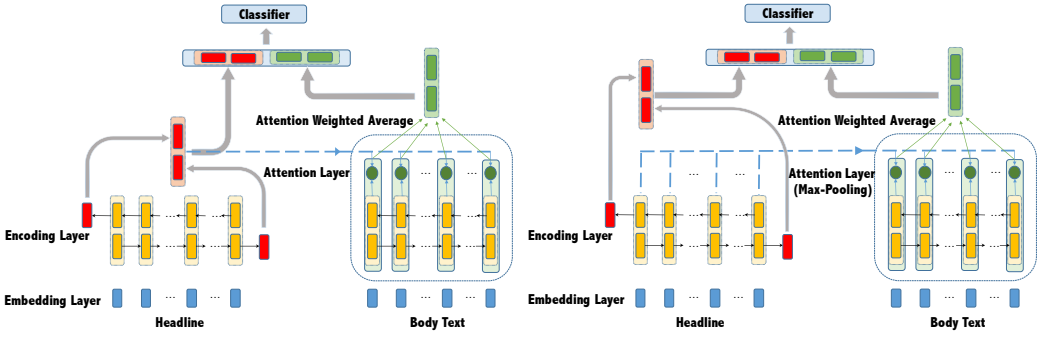

Figure 3: Illustration of Attentive Reader with simple attention (left) and full attention (right)

3.3 Bilateral Multiple Perspective Matching (BiMPM)

Another similar approach is to check the matching between headline and body text word by word in various perspectives. In the work of [6], a bilateral multi-perspective matching architecture is proposed, on which we do some slight modification for this specific task. The model is illustrated in Figure 4. We again first build the contextual representation of headline and body text using separate bidirectional RNN, as done in unconditional bidirectional encoder. Then, we calculate the cosine similarity between headline and body contextual representation vectors:

$$m_k = \text{cosine}(W_k \circ \boldsymbol{v}_1, W_k \circ \boldsymbol{v}_2) \tag{2}$$

where $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ represents the two vectors of dimension d to be compared, $\boldsymbol{W} \in \mathbb{R}^{l \times d}$ is a trainable parameter, W_k is the k-th row of $\boldsymbol{W}$.

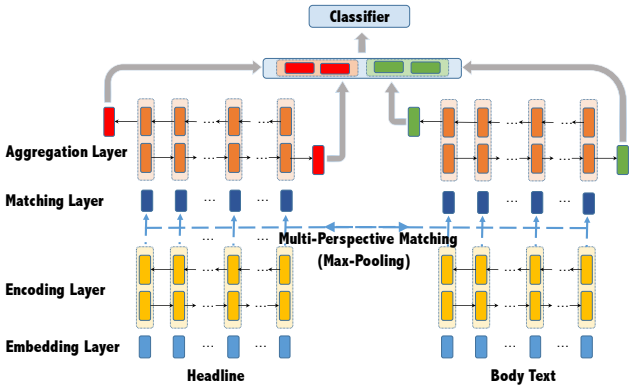

Figure 4: Illustration of Bilateral Multiple Perspective Matching

Different from the attention mechanism mentioned in Section 3.2, the output of comparison here is a l-dimension vector, with each dimension representing a certain perspective and thus assigning different weights to different perspectives. For a given time-stamp vector in body text, it compares with every time-stamp vectors in the headline and takes the maximum over all time-stamp vectors in headline for each perspective dimension. Furthermore, as suggested by [6], we adopt bilateral architecture here such that the same matching procedure is also repeated for headline time-stamps as well. As a result, one would eventually get a list of l-dimension vectors, one for each time-stamp in the headline and body text.

In addition to the l-dimenion matching vectors in matching layer, we apply an aggregation layer on top of it, which is another bidirectionl RNN layer for headline and body text respectively. Then the final states of healine RNN cells and body text RNN cells in the aggregation layer is concatenated as feature for classifier.

It should be noticed that in the original paper [6], four different kinds of mathing layers are applied and concatenated, while we only take one of them here (the maxpooling-matching). This is mainly due to the consideration of computing time.

4 Experiments

4.1 Experiment Setup

The raw text is first tokenized using "nltk" package, and then further embedded using 50-dimension word vectors using GloVe 6B. The embedding parameters are non-trainable during our experiments here since we do not think the given dataset in this task is large enough. For the out-of-vocabulary words, we initialize them with random noise. Headline texts are padded with length of 50 while body texts are padded with length of 200. In case the text length is larger than padding size it is truncated at padding size.

For all the bidirectional RNN, we use GRU cell with 100 hidden size for both kinds of Attentive Readers and BiMPM models, but 200 hidden size for three kinds of simple bidirectional encoders. Please be noticed that the hidden size mentioned here is for GRU cell in **one** direction, thus the number of dimensions of either time-stamp vector or final states is twice of it since we have GRU cells for both directions. For BiMPM model, we take number of perspective to be 40 ($l = 40$).

For the classifier, we use a 1-hidden layer forward NN with 50 dimensions on hidden layer for both kinds of Attentive Readers and BiMPM models, but 100 for three kinds of simple bidirectional encoders. The final output layer of this forward NN is a 4-class softmax representing probability of 4 classes we are predicting here. A dropout with keep probability 0.9 is applied **only** on the hidden layer of classifier.

The loss used is cross entropy, optimized using Adam optimizer [8], with 0.001 learning rate. For each model we train for 10 epoches with 32 batch size. Our experiments show that 10 epoch is enough for all models to converge without running into overfit. Thus the maximal score evaluated on development set is reported for each run. For a fixed model / configuration, we repeat the training several times, and take average to deal with randomness.

The setup above is the optimal one within all different configurations that we have scanned. More details on hyper-parameter fine tuning can be found in Section 4.2. The final evaluation results with this set of hyper-parameters can be found in Section 4.3.

4.2 Hyper-parameter Tuning

Due to the extreme large hyper-parameter space, it is impractical to scan over all possibel combinations, given our limited computing resources. Thus our strategy is to start with the configuration mentioned above, and vary each hyper-parameter one-by-one. For some hyper-parameters that are not sensitive to the choice of model, we only scan them on some certain model.

Loss Function. We notice that our dataset is very imbalanced between four classes. One way to address this would be to assign different weights to different classes during the training. In particular, we comes up with a customized loss called "conditional cross-entropy" that can easily tune such weights with only one hyper-parameter.

The standard cross-entropy for a single data point can be written as:

$$CE = \sum_k q_k \log p_k \stackrel{\text{def}}{=} \sum_k CE_k \tag{3}$$

where $\boldsymbol{q}$ is the one-hot truth label vector, $\boldsymbol{p}$ is the prediction probability. Now for classes (index k) belonging to *discuss*, *agree* and *disagree* (i.e. *related*) this can be re-written as:

$$\begin{aligned} CE_k &= -q_k \log p_k \\ &= -q_k \log(\frac{p_k}{p_1 + p_2 + p_3} \times (p_1 + p_2 + p_3)) \\ &= -q_k \log \tilde{p}_k - q_k \log(1 - p_0) \end{aligned} \tag{4}$$

where we are denoting index 0 as *unrelated* class. $\tilde{p}_k$ represents the conditional probability of class k given it is *related*. As one can see, this split cross entropy into two parts: the part for being *unrelated* and the part for being otherwise, given it is *related*. Then we are able to tune the relative weights between these two parts by applying a hyper-parameter λ on one of them:

$$CE'_k = \begin{cases} -q_k \log p_k, & k = 0 \\ -q_k \log(1 - p_0) - \lambda \cdot q_k \log \tilde{p}_k, & k \neq 0 \end{cases} \tag{5}$$

Apparently, when $\lambda = 1$, this will get back to the standard cross-entropy. When $\lambda = 0$, this will reduce to a binary classification between *related* and *unrelated*. For our problem here, we only need to scan λ around 1.

In our experiments we find that when we set $\lambda > 1$, the training will overfit rapidly. This suggests putting too much weight on *related* dataset is dangerous, possibly due to the lack of statistics in these categories. So we instead try $\lambda < 1$, as shown in Table 2. The result indicates that $\lambda = 1$, i.e., the standard cross-entropy, is still preferred, depsite of the imbalance of dataset.

	Bidirectional Encoder (concatenated)	Attentive Reader (simple attention)	Attentive Reader (full attention)	Bilateral Multiple Perspective Matching
$\lambda = 0.5$	80.8%	81.1%	80.1%	83.5%
$\lambda = 1.0$	82.7%	82.4%	83.7%	84.1%

Table 2: Average score on development set for various λ values and models

Embedding training. We did a quick scan on "Attentive Reader with simple attention" allowing embedding parameters to be trainable. The result shows that average score on development set would drop from 82.4% to 79.3% by including embedding parameters in the training. This is not too surprising given our dataset is not very big. Thus we always keep embedding non-trainable through out experiments.

Choice of RNN Cell. We find that there is no essential difference between using LSTM and GRU cells. Therefore we fix on GRU since it is slightly faster in terms of computing.

Hidden Size. A scan over different hidden size can be found in Table 3. It turns out for relatively simple models such as a direct bidirectional encoders, larger hidden size is preferred, while a smaller hidden size is favored for relatively more complex model as Attentive Reader.

	Bidirectional Encoder (concatenated)	Attentive Reader (simple attention)	Attentive Reader (full attention)
100d (RNN), 50d (classifier)	78.5%	82.4%	83.7%
200d (RNN), 100d (classifier)	82.7%	77.7%	81.2%

Table 3: Average score on development set for different RNN cell hidden size

Padding Size. For the headline part, we fix our padding length to be 50 since it is enough to cover most of cases. For the body text, a scan of padding length can be found in Table 4. It shows that 200 or 400 padding size would give roughly the same result, with difference less than 1%, while a 800 padding size might damage the performance probably because this is too large for GRU to establish a long-term memory. We eventually fix on 200 since it would be much faster than 400.

	Bidirectional Encoder (concatenated)	Attentive Reader (simple attention)	Attentive Reader (full attention)
200	82.7%	82.4%	83.7%
400	83.0%	83.2%	83.3%
800	–	80.7%	–

Table 4: Average score on development set for different padding length on body text

Dropout.In all our models, dropout is applied only at the hidden layer of classifier, hence playing a limited role. A quick scan suggests that using lower dropout value could only decrease final score by less than 1%. Therefore we fix on 0.9 keep probability throughout all models.

4.3 Evaluations

Table 5 shows the evaluation results on both development set and test set using various models with hyper-parameter fixed as described above. All neural network approaches outperform hand-crafted feature based baseline significantly.

The direct bidirectional encoding approach is performing worst among all neural network models. This is not too surprising since a simple encoding can hardly capture the matching information between headline and body text. On ther other hand, it is interesting to see that if we concatenate headline and body text first and just apply one bidirectional RNN on it, it works moderately as good as one of our Attentive Readers.

Attentive Readers, especially with full attention, and the BiMPM achieves the highest score. This is mainly because the architecture design of these two models are specific for the matching purpose between headline and body text.

Models	Ave. Dev. Score	Max Dev. Score	Ave. Test Score	Max Test Score
FNC Baseline	–	–	79.2%	–
Bidirectional Encoder (unconditional)	80.1%	80.5%	79.9%	80.1%
Bidirectional Encoder (conditional)	79.5%	81.2%	80.2%	82.0%
Bidirectional Encoder (concatenated)	82.7%	82.9%	82.0%	83.5%
Attentive Reader (simple attention)	82.4%	83.4%	81.4%	82.6%
Attentive Reader (full attention)	83.7%	85.4%	85.2%	86.5%
Bilateral Multiple Perspective Matching	84.1%	84.8%	84.6%	85.6%

Table 5: Evaluation results on both development set and test set for various models

To get some intuition on what Attentive Reader learns, Figure 5 shows two examples of correct prediction given by Attentive Reader with full attention. In case of *unrelated* prediction, the attention weight is random distribution across the words. On the other hand, for *agree* prediction, a localized region of body text that can match headline gets a significantly higher attention weights than other regions.

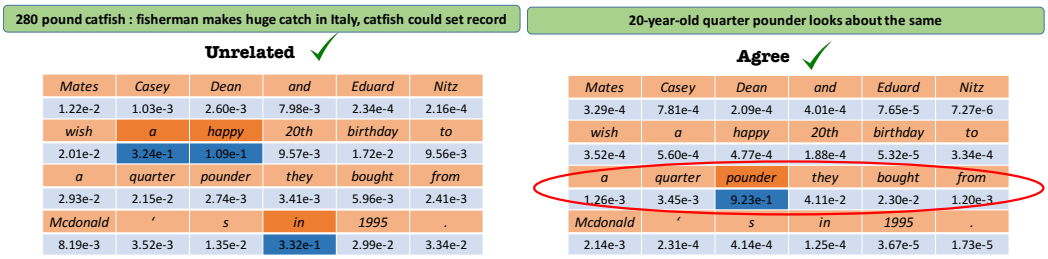

Figure 5: Examples of headline and body text alogn with attention weights

Figure 6 shows how the confusion matrix is improved step by step by using various models. Although *FNC* model is able to predict *unrelated* and *discuss* correctly, it totally mess up with the *agree* and *disagree* classes. A direct bidirectional encoder is able to achieve more accurate prediction for *agree* category, though there is a lot of leaakge of misclassification in *discuss* category. By using Attentive Reader with full attention, the recall on *agree* category gets improved significantly, though the precision is still not high enough. In all cases, we notice that *disagree* category is poorly predicted. For one thing this comes from very limited statistics of *disagree* category, while on the other hand it suggests we need some more dedicate treament of "negation" semantics between headline and body text.

5 Conclusions

In this work, we apply various recurrent neural network based models on the stance detection task for fake news challenge. Direct bidirectional encoding, Attentive Reader and Bilateral Multi-Perspective Matching models are explored. Our result suggests that all neural network based method can outperform the hand-crafted feature based approach. In addition, we improve the existing Attentive Reader with a full attention mechanism between words in body text and headlines. Evaluation result shows that this gives the best performance over other explored methods.

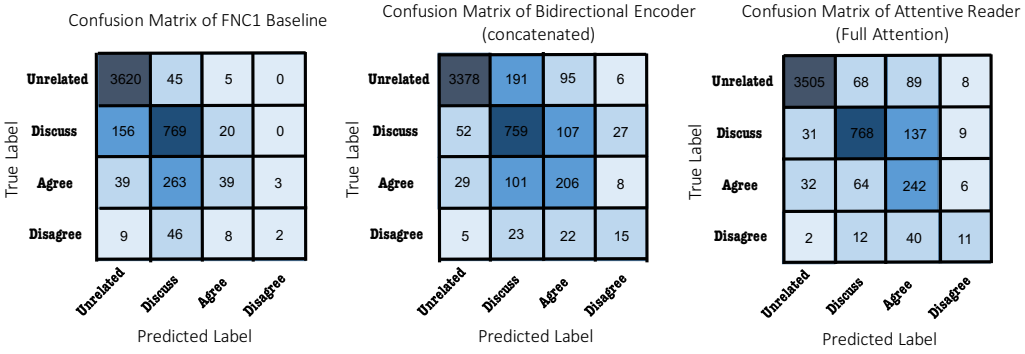

Figure 6: Confusion matrix on test set using *FNC1* Baseline (left), Bidirectional Encode (concatenated) (middle) and Attentive Reader with full attention (right)

Future Work. To further improve the performance, one needs to first improve the precision on *agree* category, and further improve the accuracy on *disagree* cateogry. One possible approach is the combination of hand-crafted features and hidden features learnt by RNN. A full implementation of the BiMPM models with all different matching layers is also worth trying. Finer tuning of hyper-parameter, e.g., embedding dimensionality, might also improve performance. Methods to handle the imbalance of current dataset would also be important for any further improvement.

Contributions of Team Member

Qi Zeng and Quan Zhou are equally involved in the implementation and improvement of all models explored in this work, hyper-parameter tuning, evaluation and analysis of results and summary report writeup. Shanshan Xu is involved in test of baseline model, bidirectional encoder and discussion.

References

[1] Samuel R. Bowman, Gabor Angeli, Christopher Potts & Christopher D. Manning. 2015. A large annotated corpous for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[2] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman & Phil Blunsom. 2015. Teaching Machine to read and comprehend. In *Neural Information Processing Systems (NIPS)*.

[3] Danqi Chen, Jason Bolton & Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In *Association for Computational Linguistics (ACL)*.

[4] William Ferriera & Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

[5] Isabelle Augenstein, Tim Rocktaschel, Andreas Vlachos & Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[6] Zhiguo Wang, Wael Hamaza & Radu Florian. 2017. Bilateral multi-Perspective matching for natural language sentences. *arXiv preprint, arXiv: 1702.03814v1*.

[7] Jeffrey Pennington, Richard Socher & Christopher D. Manning. 2014. GloVe: global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

[8] Diederik P. Kingma & Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. In *arXiv preprint, arXiv: 1412.6980*.