Cosine Siamese Models for Stance Detection

Akshay Agrawal
akshayka@stanford.edu

Delenn Chin
delenn@stanford.edu

Kevin Chen
kchen8@stanford.edu

Abstract

Fake news detection has received much attention since the 2016 United States Presidential election, where election outcomes are thought to have been influenced by the unregulated abundance of fake news articles. The recently released Fake News Challenge (FNC) aims to address the fake news problem by decomposing the problem into distinct NLP tasks, the first of which is stance detection.

We employ a neural architecture consisting of two homologous subnetworks for headline and body processing and a subsequent node for headline/body comparison and stance prediction. Headlines and bodies are represented with a weighted bag-of-words combination of word vectors passed through a ReLU, where the weights are learned. Stance is quantified by computing the cosine similarity of these weighted bag-of-words representations, and the score is regressed to a relaxed, continuous label space in which the true discrete labels are posited to lie.

Our model, which outperforms other recurrent methods, achieves an FNC score of 0.89[1] out of 1.00, a 0.10 increase from the published 0.79 FNC baseline. The cosine similarity function induces a natural geometry among the learned headline and body representations, with unrelated inputs generally orthogonal to each other and agreeing inputs nearly collinear. Our findings implicate the importance of the optimization objective, as opposed to the architecture of the subnetwork models, to success in stance detection, echoing recent work demonstrating the competitiveness of weighted bag-of-words models for textual similarity tasks.

1 Introduction

The rapid growth and availability of Internet news and information in recent years emphasizes the emerging necessity for automated information processing. As the large majority of available text is unstructured, Natural Language Processing (NLP) has become an increasingly important tool for programmatic information analysis. While fact checking and stance detection have long been considered canonical NLP tasks, recent failures of predictive algorithms in mainstream national events have generated a surge of interest in their combination for the detection of fake news.

The Fake News Challenge (FNC) was published in response to these failures to promote the development of automated programs for detecting fake news, and provides a structured dataset and task for the development of a fake news detection system [1]. Stage 1 of the FNC formally defines fake news detection as a supervised stance detection task: determining whether a given headline and text body *discuss* the same topic, *agree* with each other, *disagree* with each other, or are *unrelated* to each other. Each headline-body pair is assigned exactly one such label.

We present a Siamese-like model that uses homologous subnetworks to process and generate headline and body embeddings, which are then used in a single downstream node that uses a similarity function to output a stance classification (Fig. 1). Our model bears resemblance to the Siamese models proposed independently by Bromley et al. and Baldi and Chauvin, but uses distinct weights between the two upstream subnetworks [2, 3].

While Siamese networks have seen usage in sentence similarity tasks, few studies, if any, have directly applied them to stance detection [4]. Recent studies in stance detection have used both simple classifiers and deep learning architectures for classification, which have achieved state-of-the-art results in their respective tasks [5, 6, 7].

[1] NB: Development data was kept separate from training data to prevent information leakage from latter to former.

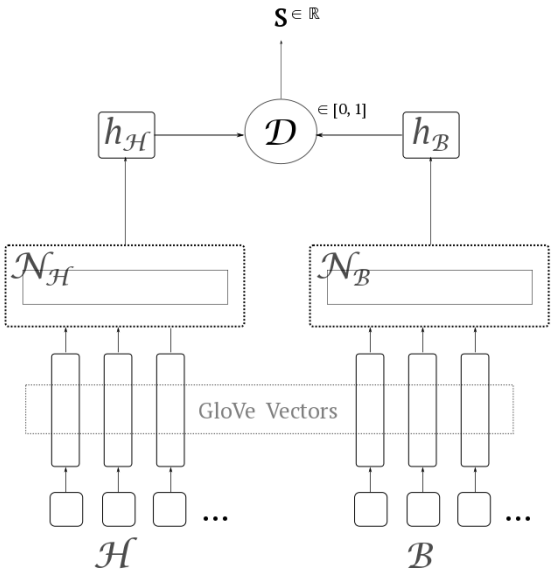


Figure 1: The architecture of our Siamese neural network. Given a headline and body pair, each element of the pair is fed through a subnetwork, $\mathcal{N}_H$ and $\mathcal{N}_B$, respectively, that generates its corresponding hidden vector $h \in \mathbb{R}^+$. The cosine distance $\mathcal{D}$ between the two hidden vectors quantifies the similarity between the input, and is then transformed affinely to obtain a score $s \in \mathbb{R}$, and the loss of the score is the absolute difference between the stance label and s.

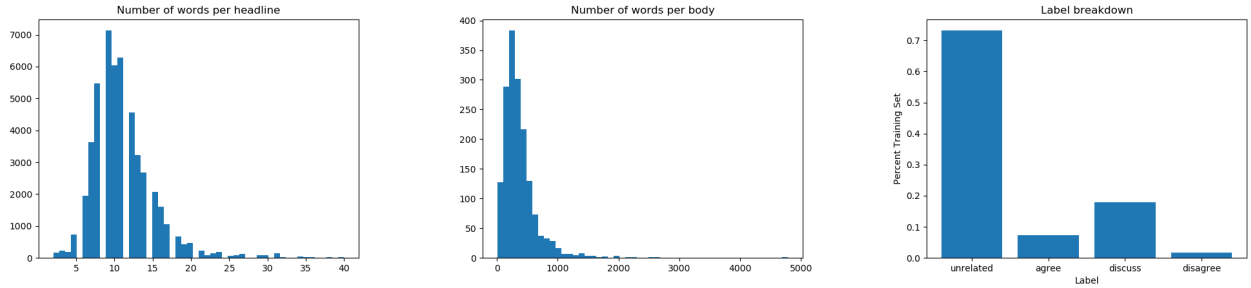
Our findings show that a carefully designed Siamese architecture, fed by two bag-of-words models and coupled with a linear regression from the output of the network to a relaxed (i.e., continuous) label, outperforms RNN-based models. As our approach does not use complex architectures that have previously found success in stance detection and is not supplemented with any additional features, we propose it as a "simple-but-tough-to-beat-baseline" for stance detection and thus the FNC-1 task. Such a baseline is in the spirit of recent work by Arora et al. that demonstrates that weighted bag-of-words models are competitive and often state-of-the-art for textual similarity tasks [17].

The remainder of the paper is structured as follows: In section 2, we analyze the FNC dataset both qualitatively and quantitatively; in section 3, we discuss the performance of naive textual similarity baselines as a proxy for the hardness of the FNC; in section 4, we present our Siamese model in full specificity; in section 5, we present models against which ours was benchmarked along with experimental results; in section 6, we discuss, visualize, and analyze the performance of our Cosine Siamese Model; in section 7, we discuss future work; and in section 8, we give closing remarks on our model and the FNC and stance detection tasks.

2 The FNC dataset

The FNC dataset consists of 49,973 pairs of headlines and article bodies, each of which is labeled with the semantic relationship between the two. The FNC task is to build a model that, given a headline-body pair, correctly ascertains the semantic relationship describing it.

The distribution of labels skews towards *unrelated* (Fig. 2), with approximately 70 percent of input pairs labeled as unrelated to each other. The FNC scoring metric is scaled accordingly, placing higher weight on correct classification on the *related* labels (*agree*, *discuss*, and *disagree*) [1]. In particular, for each evaluation example, the model is rewarded 0.25 points if the true label was *unrelated* and the prediction was the same; 0.25 points if the true label belonged to *related* and the prediction was in *related*, and an additional .75 points if the label and prediction matched. Thus, the maximum score possible is the number of *related* examples plus 0.25 times the number of unrelated examples. We call the sum total of the earned rewards the *FNC score* of a model; unless otherwise stated, all FNC scores will be expressed as a fraction of the maximum score.



(a) Headline Word Counts (b) Body Word Counts (c) Label Split

Figure 2: FNC-1 dataset statistics

Sample headline-body pair and correct label, adapted from Fake News Challenge: [1]

Headline: Robert Plant Ripped up $800M Led Zeppelin Reunion Contract
Body: "... Led Zeppelin's Robert Plant turned down £500 MILLION to reform supergroup ..."
Label: *agree*

Figure 2 shows histograms of the word counts of the headlines and bodies. Note that the long tail of the body histogram: while the median number of words per body is 302, some articles contain thousands of words. A question that we later explore is whether we can simply truncate the articles after a fixed number of words to expedite training and eliminate noise, as the salient semantic content of an article should be present in its opening paragraphs (the answer is yes: we found that capping article lengths at 100 produced good results, as discussed in section 5).

The FNC dataset is small: there are but 49,973 examples. This suggests that deep networks, as well as those with large numbers of parameters or those that retrain their input word embeddings, may over-fit to their data, an intuition that was borne out experimentally. And while there are 49,973 examples, there are many fewer distinct articles in the dataset, as the same body is present in many headline-body pairs. When training and evaluating our model, we take care to ensure that the bodies that are present in the development set are not present in the training set.

3 How hard is the FNC?

Along with their dataset, the FNC released a simple baseline classifier that achieves an FNC accuracy score of 0.795 using a gradient-boosting logistic regression classifier with hand-built features [1]. The released baseline roughly discriminates between *unrelated* and *related*, but fails to disambiguate *related* into *discuss*, *disagree*, and *agree*. Note that a naive classifier that only predicts *unrelated* achieves a score of approximately 0.40 under the weighted scoring scheme.

Unless otherwise stated, in what follows, we assign the integer label 0 to *unrelated*, 1 to *discuss*, 2 to *disagree*, and 3 to *agree*.

3.1 Neural baselines: Siamese-like networks with a softmax classifier

We use a simplification of the Siamese architecture depicted in Figure 1 as our neural baseline. 1-layer networks are used for the homologous networks. The final hidden states for each subnetwork are output as the cumulative representation for headline and body text. Given hidden states $h_{\mathcal{H}}, h_{\mathcal{B}} \in \mathbb{R}^H$, where H represents the embedding dimension, a prediction is generated as $\operatorname{argmax}(\operatorname{softmax}(W[h_{\mathcal{H}}, h_{\mathcal{B}}] + b))$, where $W \in \mathbb{R}^{4\times 2H}$ and $b \in \mathbb{R}^4$.

We evaluate the performance of our baseline model through the use 300 dimensional GloVe vectors, 300 dimensional hidden states, and four traditional deep-learning models: bag of words, RNN, GRU, and LSTM, which achieve training/development FNC scores of 0.84/0.57, 0.80/0.52, 0.85/0.55, and 0.86/0.52, respectively [8]. Our deep-learning

models fail to match the performance of even the FNC hand-rolled baseline, prompting a re-examination of the Siamese-like softmax network architecture.

3.2 A convex baseline: Exploiting textual similarity

To further probe the difficulty of the FNC and to gain a sense of the extent to which the FNC reduces to a keyword matching textual similarity task, we built two naive quadratic regression models. The first model maps the Jaccard index of each input pair to a score in $\mathbb{R}$ and rounds it to the nearest integer in $[0, 3]$, while the second regresses upon the (non-binary) cosine similarities (Fig. A.1).

The models were fit and evaluated on the entirety of the FNC data, imposing upper bounds on their generalizability. The former model earned an FNC score of 0.70, while the latter earned a score of 0.82, exhibiting a significant relationship between textual similarity and stance. The performance of such simple models motivates the design of the following model.

4 A Cosine Similarity Siamese Model

In this section, we present our primary modeling contributions: (1) a Siamese network that quantifies a headline-body pair's stance with the cosine similarity of its subnetworks' hidden outputs, and (2) a polynomial regression from the output of the Siamese network to a *relaxed* prediction in $\mathbb{R}$, rounded to the nearest integer in $[0, 3]$ to produce a bonafide prediction.

The network (Fig. 1) is trained by backpropagating through the regression, to subnetworks $\mathcal{N}_\mathcal{H}$ and $\mathcal{N}_\mathcal{B}$. The intuition here is that, by restricting the hidden outputs $h_\mathcal{H}$ and $h_\mathcal{B}$ to non-negative values, the optimization objective (ideally) forces the network to learn sentence and paragraph representations where *unrelated* inputs are orthogonal, inputs that *agree* with each other are collinear, and inputs that either *discuss* the same topic or *disagree* with each other form acute angles, with the angle formed between inputs of the former type larger than those formed between the latter type.

We briefly describe our pre-processing pipeline and the inputs to our Siamese subnetworks $\mathcal{N}_\mathcal{H}$ and $\mathcal{N}_\mathcal{B}$ before describing our model in full specificity.

4.1 Input Pre-processing

Each text input (a headline or a body) is represented as a sequence of tokens. We tokenize[2] all inputs into contiguous strings of alphabetical characters, segmenting words at whitespace or punctuation boundaries (with the exception of apostrophes, which are deleted from the dataset in a first pass), and remove stopwords as per the NLTK English stopwords list [10]. Bodies are truncated to a maximum length, specified by the modeler. All tokens are cast to lowercase.

The sequence of tokens is mapped to a sequence of GloVe vectors, which is provided to the subnetworks as input. Tokens not present in the GloVe vocabulary are discarded (approximately 10% of input vocabulary) [3] Note, however, that many of these are not true "words", such as gibberish strings or hashes from a url.

4.2 Headline and body subnetworks

The subnetworks $\mathcal{N}_\mathcal{H}$ and $\mathcal{N}_\mathcal{B}$ in our model are homologous but untied: each possesses its own set of parameters, as we hypothesize that headlines and bodies are sufficiently distinct in style so as to merit tailored hidden representations. Each subnetwork is a map $\mathcal{N}_k \colon \mathbb{R}^{L_k \times d} \to \mathbb{R}^H$, $k \in \{\mathcal{H}, \mathcal{B}\}$, where L_k is the maximum headline or body length, d is the dimension of the GloVe vectors, and H is the dimension of the hidden representation. Any such map is acceptable.

Our best-performing models use simple weighted bag-of-words subnetworks $\mathcal{N}_\mathcal{H}$ and $\mathcal{N}_\mathcal{B}$. In particular, letting $V_k \in \mathbb{R}^{L_k \times d}$ be the input sequence of GloVe vectors for either a headline or a body, $\tilde{v}_k \in \mathbb{R}^d$ the mean[4] of the GloVe vectors

[2] r"[a-zA-Z]+[']?[a-zA-Z]+"

[3] Other methods of handling unmappable words, such as averaged GloVe vectors of the token characters, or random vector mapping, had no appreciable impact on performance.

[4] The mean is taken with respect to the number of tokens in tokenized headline or body, after truncation; so, if the length l of an example is less than L_k, we obtain V_k by padding it with zero vectors and ignoring said zero vectors in all further computations.

in V_k, and $W_k \in \mathbb{R}^{H \times d}$ a weight matrix,

$$\mathcal{N}_k(V_k) = \text{ReLU}(W_k \tilde{v}_k). \tag{1}$$

We benchmark the weighted bag-of-words subnetworks against standard recurrent subnetworks (RNNs, GRUs, and LSTMs, respectively), where the subnetworks produce $h_{\mathcal{H}}$ and $h_{\mathcal{B}}$ by outputting their final hidden states. See section 5 for an analysis of performance by subnetwork type; as a preview, while [4] use LSTM subnetworks for a similar task, we find that our weighted bag-of-words models are more successful.

4.3 The distance function

The similarity function $\mathcal{D}$ (Fig. 1) is a map $\mathcal{D}\colon \mathbb{R}^H \to \mathbb{R}$ that quantifies the extent of semantic similarity between input headline and body. Any such function D is acceptable. Mueller et al. use the inverse exponential of the Manhattan distance as their function $\mathcal{D}$ [4], whereas we propose the cosine similarity

$$\mathcal{D}(h_{\mathcal{H}}, h_{\mathcal{B}}) = \frac{{h_{\mathcal{H}}}^T h_{\mathcal{B}}}{||h_{\mathcal{H}}||_2 ||h_{\mathcal{B}}||_2}.$$

Bromley et al. also use the cosine distance in conjunction with a Siamese network, though their task is a binary classification problem and they do not regress upon their scores [2].

4.4 Regressing distances to relaxed FNC labels

After computing a similarity score $s = \mathcal{D}(h_{\mathcal{H}}, h_{\mathcal{B}})$, the network predicts the stance of the input pair. This prediction is made by regressing from s to the true target labels [5]. In particular, with y_i as the gold label for the i-th input, we choose $m \in \mathbb{R}$ and $b \in \mathbb{R}$ to minimize

$$\mathcal{L} = \sum_i \frac{1}{4} \mathbb{1}[y_i = 0] \cdot |y_i - (m \cdot s_i + b)| + \mathbb{1}[y_i \neq 0] \cdot |y_i - (m \cdot s_i + b)|. \tag{2}$$

The prediction for the i-th input is then obtained by rounding $m \cdot s_i + b$ to the nearest integer in $[0, 3]$.

To motivate and interpret this loss function, note that our assignment of integers to labels is not arbitrary; label magnitude loosely represents the extent of similarity between input pairs. Inputs are assumed to be wholly dissimilar if they are *unrelated*, slightly similar if they *discuss* the same topic, more similar if they *disagree* on the topic (as disagreeing implies discussion), and even more similar if they *agree* on the topic. We go further and assume that the discrete gold labels are but the observations of a latent, relaxed continuous label space. For example, two inputs may both *discuss* the same topic, and so the would both bear the gold label 1, but the extent to which the first pair discusses the same topic may be greater than the extent to which the second pair does; as such, it might be the case that the latent relaxed labels for these pairs are 1.3 and 1.1, respectively. In this manner, our loss function encourages the learning of a spectrum of stance and similarity.

Moreover, note that our loss function both explicitly and implicitly weights certain incorrect predictions as more desirable than others. The explicit weight is in the form of the indicators: we down-weight the penalty for incorrect classifications of the unrelated label by a quarter in order to more directly optimize for the FNC scoring metric. The implicit weighting can be seen through example: if, say, the true stance of an input pair is *agree*, the penalty our loss function assigns to an incorrect prediction is proportional to the distance from 3. In particular, a prediction near 0 (i.e., *unrelated*) will be more penalized than a prediction of *discuss*. Contrast this to the softmax classifier with cross entropy loss, which does not do any such implicit weighting (classifying *agree* as *unrelated* is no worse than classifying it as *discuss* in the softmax/cross entropy scheme).

Regarding prediction, defining $\mathcal{N}_k$ as in Equation 1 and $\mathcal{D}$ as the cosine similarity, note that each s_i will lie in $[0, 1]$ (because $\mathcal{N}_k$ applies a ReLU). Geometrically, our model assigns to each input pair a point on the unit circle, restricted to the first quadrant, and the unit circle is carved into four sections, one per stance. The input prediction is determined by the corresponding section (Fig.5 provides a visualization).

[5] Recall that 0 : *unrelated*, 1 : *discuss*, 2 :*disagree*, 3 : *agree*.

5 Experiments and Results

Unless otherwise stated, all experiments were run with a maximum body length of 100 words, 300 dimensional GloVe and hidden representations, and *no* word vector retraining (given the relatively small size of the dataset). All FNC scores were obtained via best-of-3-fold[6] cross validation, with 80/20 percent training and development splits that enforced the bodies in the development sets to be distinct from those in the training sets. Recurrent models were run with input dropout of 0.30 at each cell, and, with the exception of the sequence-to-sequence models, were run with maximum body lengths of 30. Models were trained for 60 epochs, using Tensorflow's Adam optimizer.

Our Cosine Siamese Model with weighted bag-of-words subnetworks (henceforth referred to as CS+WBOW) was benchmarked against the following models:

1. (CS+RWBOW) Our Siamese model with *random* 300 dimensional vectors instead of GloVe vectors as inputs.
2. (CS+ARORA) Our Siamese model with Arora-weighted embeddings [17].
3. (CS+IDF) Our Siamese model with embeddings weighted by inverse-document frequency.
4. (M+WBOW) Our Siamese model with an inverse exponential manhattan distance function, as in [4].
5. (SM+WBOW/RNN/GRU/LSTM) Our weighted-bag-of-word / recurrent neural baselines with softmax classifiers, described in section 3.1.
6. (SM+RNN/GRU/LSTM+CSF) A modification to the neural baselines in which the cosine similarity between the raw inputs is appended as a feature to the concatenated vector of hidden states before passing it through the affine transformation and softmax.
7. (CS+GRU/LSTM) Cosine Siamese Models with recurrent subnetworks (GRU, LSTM), as described in section 4.2.
8. (S2S+ATTN) A sequence-to-sequence model with attention, in which the body was reversed and fed through a recurrent subnetwork composed of GRU cells, and the final hidden state of the body subnetwork was provided as the initial state to a recurrent subnetwork composed of GRU cells. In the latter subnetwork, each cell was given attention to all the outputs of the former subnetwork, and the score of each output was computed as the cosine similarity of the output with the current hidden state. The final state of the headline subnetwork was put through an affine transformation and classified via a softmax with cross entropy loss.

The results are compiled in Table 1.

Method	FNC Score (Train)	FNC Score (Dev.)
FNC published baseline	?	0.79
CS+WBOW	0.99	**0.89**
CS+RWBOW	1.00	0.83
CS+ARORA	0.99	0.86
CS+IDF	0.99	0.86
M+WBOW	0.88	0.78
SM+WBOW	0.84	0.57
SM+RNN	0.80	0.52
SM+GRU	0.85	0.55
SM+LSTM	0.86	0.52
SM+WBOW+CSF	0.9	0.87
SM+RNN+CSF	0.95	0.88
SM+GRU+CSF	0.99	**0.89**
SM+LSTM+CSF	0.99	0.88
S2S+ATTN	0.99	0.83
CS+S2S+ATTN	0.72	0.49
CS+GRU	0.99	0.84
CS+LSTM	0.99	0.84

Table 1: Scores for variants of each model. The models with the best performance are displayed in boldface.

[6]We use best-of instead of the mean score to be consistent with the released FNC baseline [1].

Note that our choice of the cosine similarity for $\mathcal{D}$ improves over Mueller et al.'s method by 0.11 points [4], and a 0.32-0.37 increase over the softmax Siamese models.

5.1 Using the raw cosine similarity as a feature significantly improves the softmax models

None of the models managed to outperform CS+WBOW, though SM+GRU+CSF (that is, the GRU softmax augmented with the cosine similarity of the raw textual inputs) tied it. Interestingly, simply providing the cosine similarity feature to the softmax models increased their scores by approximately 0.30 points each. This finding agrees with the one made in section 3.2, where we note the strong relationship between cosine similarity and the stance. The softmax models augmented with the cosine similarity feature significantly outperform the raw cosine similarity regression in section 3.2, suggesting that using word vectors does indeed boost performance.

5.2 Sequence-to-sequence models either over-or-under-fit

The sequence-to-sequence with attention models did not perform well at all. The softmax version earned a score of 0.83; for context, the CS+RWBOW method (i.e., initialized with *random* 300 dimensional vectors instead of GloVe vectors) earned the same score. Swapping the softmax for the cosine similarity resulted in abysmal performance; further investigation is needed to determine why the sequence-to-sequence models performed so poorly, but one hypothesis is that they are simply much harder to train well; the softmax model overfit while the cosine/regression model underfit.

Experimental results suggest that recurrent networks see less benefits from the usage of the cosine similarity and regression for stance prediction. While the performance of the bag of words model increased substantially, the results in Table 1 show that the improvements of recurrent subnetwork models see less performance gains.

5.3 Truncating bodies improves performance

The maximum body length in the provided training set is 4,788 (with stop-words), bottle-necking computation time. Including too many words of the body also runs the risk of polluting the hidden representation with noise, as the majority of semantic information relevant to stance should be captured in the opening paragraphs or "nut graph" of the article. Figure A.2 plots the performance of the CS+WBOW classifier as the maximum body length was varied and all other variables held constant. In contrast to the CS+WBOW model, we found that the recurrent models performed optimally with 20-30-word bodies.

5.4 Regularization and dropout do not improve generalizability

A number of models in Table 1 appear to overfit to the training data, reaching training scores of ≥ 0.98. Unfortunately, as shown in Figure A.3, standard techniques like ℓ_2-regularization and dropout did not improve generalizability.

5.5 WBOW models train much faster than recurrent models

We find that our Siamese models generally converge within 60 epochs. Model development accuracy typically stabilizes after around 40 epochs (Fig.A.5, A.6, A.7). Beyond 40 iterations, training accuracy continues to increase while development performance remains constant, a hallmark sign of over-fitting.

In terms of computation time, our WBOW models take roughly 8 minutes to train on a 2012 Macbook Pro (16GB of RAM, 2.7 GHz Intel core i7) running a Tensorflow binary compiled with AVX, SSE4.1, and SSE4.2 instructions. The recurrent models take on the order of hours to train on the provided Azure GPU instance.

6 Discussion

6.1 Error analysis of best model

Overall recall was high, but there was a clear weak point in the precision for *disagree*, where we guessed *disagree* on almost equal numbers of all of the *related* labels (Fig. 3, 4). This could be a consequence of our loss function.

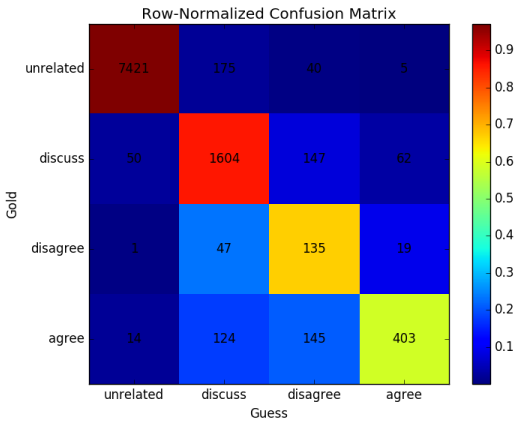


Figure 3: Recall confusion matrix for CS+WBOW.

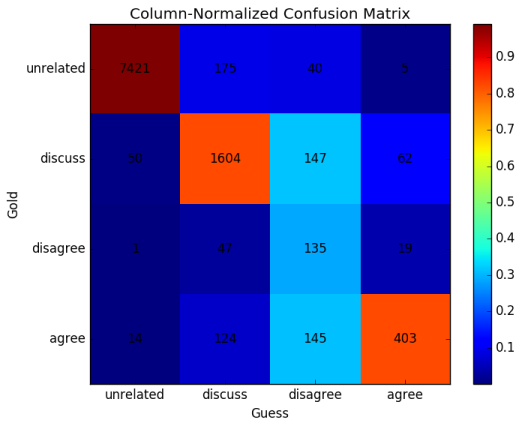


Figure 4: Precision confusion matrix for CS+WBOW.

We hypothesize that when our model encounters headline and body pairs that are hard to predict, it maps them to the central region of the output space to minimize incurred loss.

Consider the following *unrelated* example taken from the FNC dataset, which our model misclassifies as *discuss*:

Headline: *US probing claims ISIS fighters seized airdropped weapons meant for Kurds*

Body: *Eleven commercial jets have reportedly been stolen in recent weeks in Libya, and Western intelligence agencies have begun warning they could be used in terror attacks on Sept. 11, the anniversary of the devastating Osama bin Laden-orchestrated attacks on New York and Washington that left nearly 3,000 dead. According to a report in the Free Beacon, the jets were taken by Islamist militias in Libya, and reports distributed within the U.S. government included a warning that one or more of the aircraft could be used in an attack on the date marking the anniversary of the Sept. 11, 2001, terrorist attacks*

It is clear that this example is difficult to classify even for humans skimming over the texts. We therefore suspect that beating our model's current results might take a more nuanced approach, possibly incorporating specific linguistic features.

6.2 Analysis of learned representations

Figure 5 visualizes the regression from (cosines of) angles to labels that the model learns. Note the implied geometric structure: the hidden state vectors of inputs that are *unrelated* end up orthogonal to each other, and the angle between inputs decreases as the related-ness increases.

This relationship is made more tangible in Figure 6, in which we project select headlines into two dimensions using PCA. Note that headlines tend to cluster collinearly by topic. We chose the selected keywords by skimming the FNC dataset bodies for topics and keywords that occurred often (and found that the dataset is strongly skewed toward terrorism and technology).

6.3 Why a Siamese-like WBOW works well

This leads us to the root of our discussion: why does a Siamese-like weighted bag of words model work well? While the use of Siamese networks for similarity detection tasks is well documented, our choice to use homologous untied subnetworks stems from the inherent differences between article headlines and article bodies [3, 2, 4]. Headlines are meant to represent one-line summaries, and thus can have distinct structure and semantics from bodies. This modification to the original Siamese network allows each subnetwork to better fit to its distinct input type, in contrast to previous Siamese network usages where inputs were not inherently different.

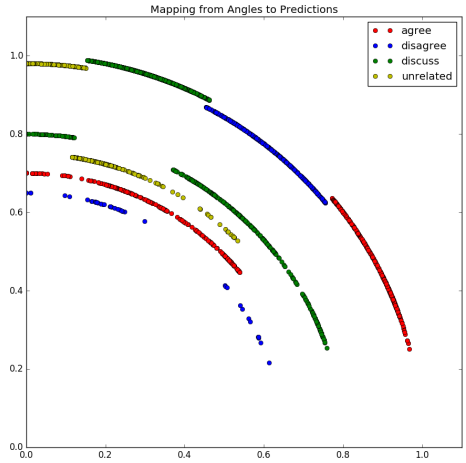


Figure 5: Map of angle between angle and final hidden state headline and body vectors, as learned by CS + WBOW. Outer and inner rings contain all and incorrect predictions, respectively

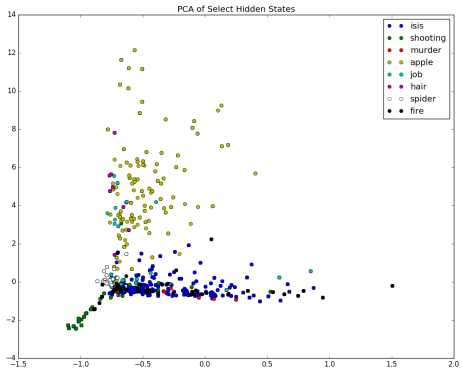


Figure 6: PCA to two dimensions from the final headline hidden states as learned by CS+WBOW, for headlines with select keywords. Similar topics cluster roughly collinearly, a property encouraged by the cosine similarity function.

We found body truncation to also play a large factor in improving our score. Our original rationale for truncation was that an article body is more likely to discuss the headline content at the beginning of the article. Thus, by taking only the first n tokens, the body subnetwork is given a more distilled version of the body text, which is more likely to pertain to its true topic, and thus headline. This 'concentration' of body topic then allows the bag of words model to more effectively capture topic in article representations, which also benefits stance prediction through cosine regression.

That the WBOW models are competitive with the recurrent architectures supports a hypothesis that the *order* of the words in a headline/input is hardly, if at all, relevant to stance detection. This is certainly intuitive for discriminating between *unrelated* and *related*, and somewhat intuitive for disambiguating *related*; at the very least, a human would likely be able to disambiguate *related* even if tokens were permuted.

We previously noted the greater performance gains associated with cosine similarity function as opposed to the Manhattan one. It is likely that the increased accuracy of cosine similarity results from inherent normalization during computation and the natural geometric structure it induces (orthogonality and collinearity). This increased expressivity allows for more nuanced distinctions between headline-body pairs, and thus higher performance.

7 Future Work

Future work to achieve higher FNC scores will likely require more sophisticated architectures designed to capture nuanced semantic information lost to our model in order to better disambiguate the tough *related* examples. One line of future work is to tune our sequence-to-sequence models to combat over-and-under-fitting exhibited by our experiments. We also plan to apply CS+WBOW to the standard NLP benchmarks for textual similarity, so that we can more conretely compare our model with, say, the Manhattan model proposed by [4].

8 Conclusion

We explore the use of a Siamese network, obtaining competitive results through the use of a cosine similarity distance function and linear regression. Our improved model achieves an FNC accuracy score of 0.89, validating the use of a Cosine Siamese Model with weighted bag of words subnetwork models, where the weights are learned, for stance detection. Given its elegance, simplicity, and performance, we propose this architecture as a difficult-to-beat baseline for future deep-learning models in the FNC-1, and more broadly, for other stance and similarity detection tasks.

Acknowledgments

We thank our mentor, Iganacio Cases, for his guidance and support in this study, Stanford CS224N course staff for administrative and educational help, and Microsoft Corp. for providing academic use of the Azure platform.

References

[1] Fake News Challenge, "Fake News Challenge," `http://www.fakenewschallenge.org` 2017.

[2] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. Lecun, C. Moore, E. Sckinger, and R. Shah, Signature Verification Using a Siamese Time Delay Neural Network, Int. J. Pattern Recognit. Artif. Intell., vol. 7, no. 4, pp. 669688, 1993.

[3] P. Baldi and Y. Chauvin, Neural Networks for Fingerprint Recognition, Neural Comput., vol. 5, no. 3, pp. 402418, 1993.

[4] J. Mueller and A. Thyagarajan, Learning Sentence Similarity with Siamese Recurrent Architectures, no. 2012, 2015.

[5] W. Ferreira and A. Vlachos, "Emergent: a novel data-set for stance classification," Naacl2016, no. 1, pp. 11631168, 2016.

[6] I. Augenstein, T. Rocktschel, A. Vlachos, and K. Bontcheva, "Stance Detection with Bidirectional Conditional Encoding," Empir. Methods Nat. Lang. Process., no. 2010, pp. 876885, 2016.

[7] W. Wei, X. Zhang, X. Liu, W. Chen, and T. Wang, "pkudblab at SemEval-2016 Task 6: A Specific Convolutional Neural Network System for Effective Stance Detection," Proc. 10th Int. Work. Semant. Eval., pp. 396400, 2016.

[8] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," Proc. 2014 Conf. Empir. Methods Nat. Lang. Process., pp. 15321543, 2014.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," Nips, pp. 19, 2013.

[10] E. Loper and S. Bird, NLTK: The natural language toolkit NLTK: The Natural Language Toolkit, Proc. COLING/ACL Interact. Present. Sess., no. March, pp. 6972, 2016.

[11] W. Yih, K. Toutanova, J. Platt, and C. Meek, Learning discriminative projections for text similarity measures, Proc. Fifteenth Conf. Comput. Nat. Lang. Learn., no. June, pp. 247256, 2011.

[12] J. Mitchell and M. Lapata, "Vector-based Models of Semantic Composition.," Acl, vol. 8, no. June, pp. 236244, 2008.

[13] J. Mitchell and M. Lapata, "Composition in distributional models of semantics.," Cogn. Sci., vol. 34, no. 8, pp. 13881429, 2010.

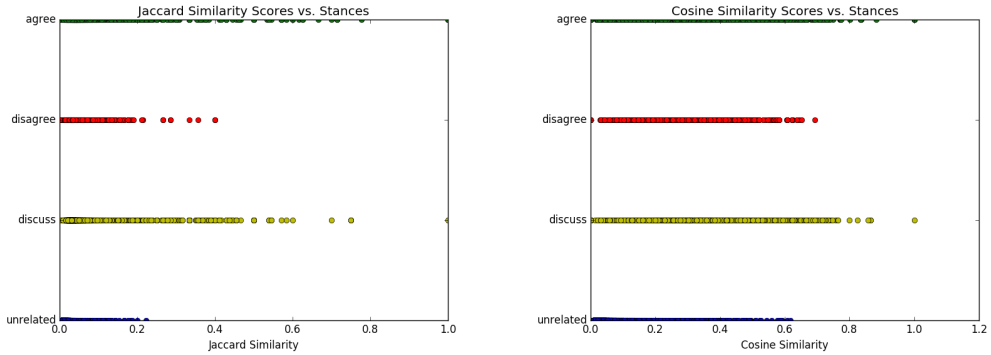
[14] W. Blacoe and M. Lapata, "A Comparison of Vector-based Representations for Semantic Composition," Proc. 2012 Jt. Conf. Empir. Methods Nat. Lang. Process. Comput. Nat. Lang. Learn. (EMNLP-CoNLL 12), no. July, pp. 546556, 2012.

[15] R. Socher, E. Huang, and J. Pennington, "Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection.," Adv. Neural Inf. Process. Syst., pp. 801809, 2011.

[16] K. S. Tai, R. Socher, and C. D. Manning, "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks," Proc. 53rd Annu. Meet. Assoc. Comput. Linguist. 7th Int. Jt. Conf. Nat. Lang. Process., pp. 15561566, 2015.

[17] S. Arora, Y. Liang, and T. Ma, "A Simple but Tough-to-beat Baseline for Sentence Embeddings," Iclr, pp. 114, 2017.

A Appendix



(a) Jaccard Index vs. Stances

(b) Cosine Similarity vs. Stances

Figure A.1: Regressing from similarity scores to stances was on par with the FNC baseline and the neural baselines, suggesting that the stance of a body with respect to a headline is closely related to the keyword similarity between the two.

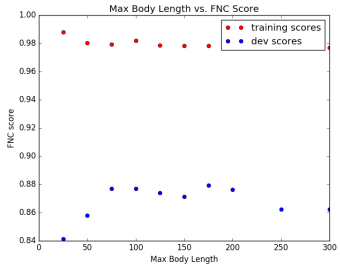


Figure A.2: Training and development scores of CS+WBOW as the maximum body length was varied. Truncating bodies to lengths of 100-200 words seems to result in the best performance, which agrees with intuition.

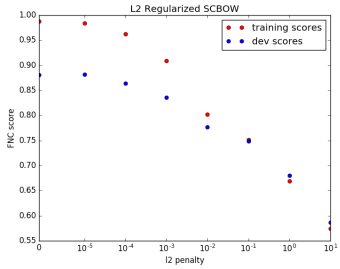


Figure A.3: Training and development scores of CS+WBOW as ℓ_2 regularization penalty was varied. Regularization does not seem to improve generalizability.

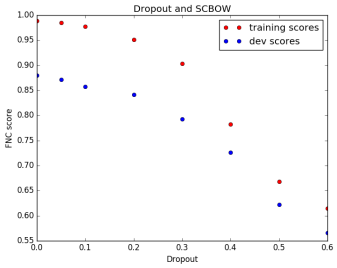


Figure A.4: Training and development scores of CS+WBOW as dropout was varied. Dropout does not seem to improve generalizability.

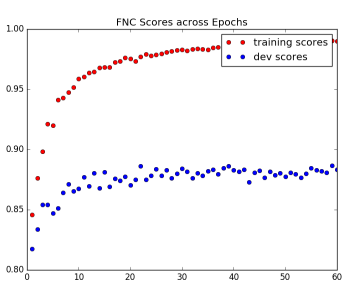


Figure A.5: CS+WBOW

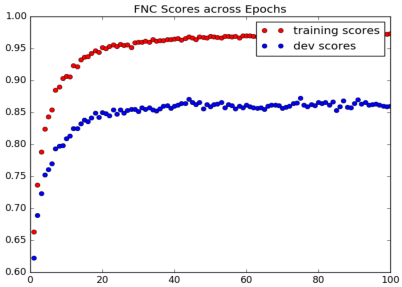


Figure A.6: SM+GRU+CSF

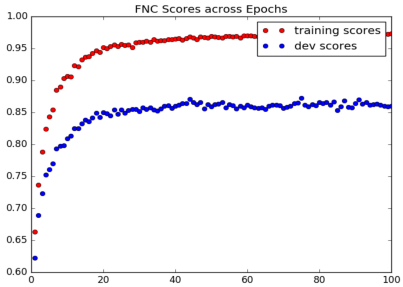


Figure A.7: SM+GRU+CSF

Development scores level off after 40 epochs, while training scores continue to increase, a sign of over-fitting.