Comparing Models for Natural Language Inference on the SNLI Dataset

Montgomery Gole
Dept. of Computer Science
Toronto Metropolitan University
Toronto, Canada
mgole@torontomu.ca

***Abstract*—This paper explores various natural language inference models for the Stanford Natural Language Inference dataset. The models evaluated were: the basic neural network approach from Bowman et al.'s SNLI paper with LSTM sentence embeddings and all-mpnet-base-b2 based sentence embeddings, a conditional LSTM model, and a word-by-word attention model from Rocktaschel, et al. The goal of this paper is to compare these model's performance. The results show that the MPnet sentence embedding model and the conditional LSTM model outperformed the basic LSTM model. Unfortunately, the attention-based model was not evaluated. These results are promising, but further research is needed to confirm their validity with the full SNLI dataset.**

***Index Terms*—Language Model-based sentence embeddings, NLI, word-by-word attention**

I. Introduction

Inference is a fundamental process for human behavior [1] which entails natural language-based inference. Natural language inference (NLI) is a natural language processing (NLP) task that involves determining the relationship between two sequences of text: the premise, and the hypothesis. Specifically, this relationship is whether or not the hypothesis entails the premise–the premise provides enough evidence to conclude that the hypothesis is true. The three classes this relationship can be assigned to are entailment (the premise provides evidence for the truthfulness of the hypothesis), contradiction (the premise provides evidence for the falsity of the hypothesis), and neutral (the premise does not provide sufficient evidence for either classification). The Stanford Natural Language Inference dataset has been widely used as a benchmark for evaluating NLI models [2]. The basic model presented by [2] for NLI with the SNLI dataset uses three sentence-embedding methods in their model–which will have a detailed explanation in the next section. These techniques are the sum of word embeddings, vanilla RNN encoding, and LSTM encoding. This paper will attempt to use the *all-mpnet-base-v2* Language Model-based sentence embeddings [3] with the basic architecture described in [2], and compare it against the LSTM based sentence embeddings, as well compare 2 more models for NLI. These models come from *Reasoning About Entailment With Neural Attention*, one being their conditional LSTM-based encoding model, and the other being their word by word attention model [4]. Overall, this paper attempts to answer the question: *How does the performance of the models above (basic SNLI model with all-mpnet-base-v2 based sentence embedding model, basic SNLI model with LSTM-based sentence embeddings, conditional LSTM model, and word-by-word attention model) compare when applied to the Stanford Natural Language Inference dataset?*

II. Background

Natural language inference has had a low frequency of related research publications in the past. However, near when Bowman et al. released the SNLI dataset, a corpus of 570k high-quality NLI based observations, researchers became more interested in the topic of NLI. Each observation consists of three rows: premise, hypothesis, and label. The SNLI dataset consists of four labels, (-1=inconclusive, 0=entailment, 1=neutral, 2=contradiction). The distribution of said labels are similar, as seen in Figure 1, 2, and 3, the distribution of the labels within the train, test, and validation sets are similar, each of the entailment, neutral, contradiction around 1/3 of the full length, and a very low value for the inconclusive label. Bowman et al. attempted to "force" their dataset to be balanced like this by having Amazon Mechanical Turk workers manually create three hypotheses for each premise. These hypotheses would be *entailment, neutral*, and *contradiction* labels for each premise. The premises were taken from a different crowd-sourced dataset called Flickr30k [5] which is a set of 160k captions from 30k images. Bowman et al. describe their dataset's validation process for a random 10% of their dataset. The purpose of their validation step was to tell whether a given premise-hypothesis pair would have a majority label vote from 5 annotators. If 3/5 annotators agreed on the label, it would be considered a consensus, and this consensus didn't occur in 2% of their validation subset. They found an average Fleiss k score of 0.7 between annotators within this subset [2]. The text length distributions, as seen in *Figure 4, Figure 5*, and *Figure 6* show that the test, train, and validation sets have similarly distributed text lengths as well as mean, min, max, and median values. Furthermore, as word, and sentence embeddings will be used while training the models, a similar distribution of words is necessary within each set in order to allow for our training set to make accurate predictions on unseen testing data. The SNLI dataset also has similar bigram

Training Bigrams	Validation Bigrams	Test Bigrams
(('two', 'men'), 23146)	(('two', 'men'), 438)	(('two', 'men'), 490)
(('man', 'wearing'), 19043)	(('man', 'wearing'), 369)	(('man', 'wearing'), 445)
(('group', 'people'), 16598)	(('group', 'people'), 342)	(('group', 'people'), 335)
(('two', 'people'), 15621)	(('two', 'people'), 283)	(('woman', 'wearing'), 253)
(('two', 'women'), 10988)	(('young', 'man'), 270)	(('two', 'people'), 253)
(('woman', 'wearing'), 10445)	(('man', 'woman'), 258)	(('young', 'boy'), 243)
(('young', 'boy'), 10346)	(('young', 'boy'), 231)	(('young', 'man'), 196)
(('little', 'girl'), 9387)	(('two', 'women'), 220)	(('two', 'women'), 181)
(('man', 'woman'), 9085)	(('woman', 'wearing'), 205)	(('young', 'girl'), 172)
(('young', 'man'), 8886)	(('little', 'girl'), 183)	(('man', 'woman'), 159)
(('young', 'girl'), 8562)	(('man', 'black'), 167)	(('wearing', 'blue'), 148)

TABLE I
TOP 11 BIGRAMS PER SUBSET

frequency distributions with respect to each subset of data as seen in *Table 1*.

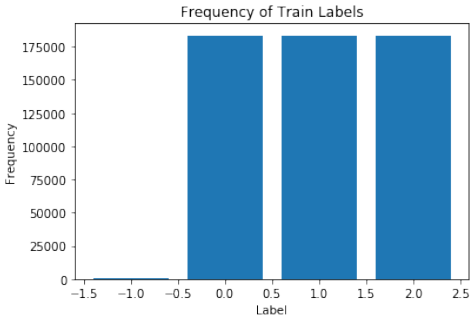

Fig. 1. Train set label distribution (1: 33.3% 2: 33.3% 0: 33.3% -1: 0.1%)

A. SNLI basic models

In order to detect entailment with their SNLI dataset with respect to sentence embedding based models, Bowman et al., for each observation, took its respective premise, and hypothesis, inputted each to a sentence embedding model to create 100-dimensional sentence embeddings, and put the concatenated embeddings through three 200-dimensional *tanh* layers, until finally using a softmax layer for a 3-way classification. Their sentence embedding models were characterized as: 100d sum of words, 100d RNN, and 100d LSTM RNN. This paper

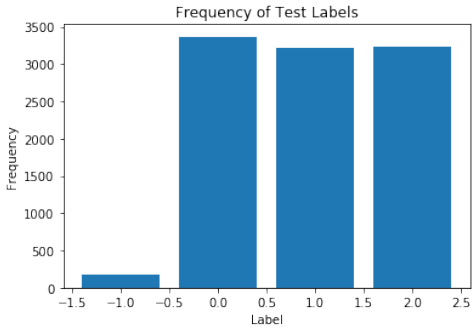

Fig. 2. Test set label distribution (1: 32.2% 0: 33.7% 2: 32.4% -1: 1.8%)

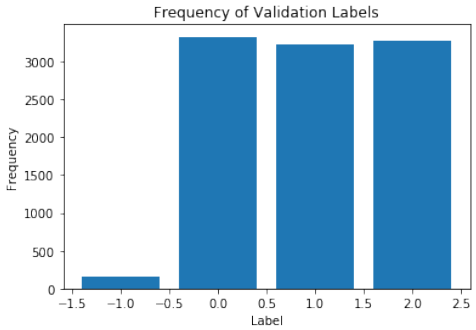

Fig. 3. Validation set label distribution (1: 32.35% 0: 33.29% 2: 32.78% -1: 1.58%)

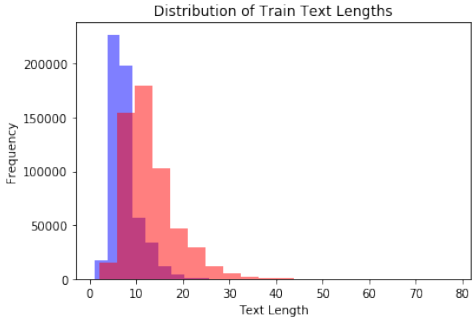

Fig. 4. Train set text length distribution (Mean: 10.1, Median: 9, Max: 78, Min: 1)

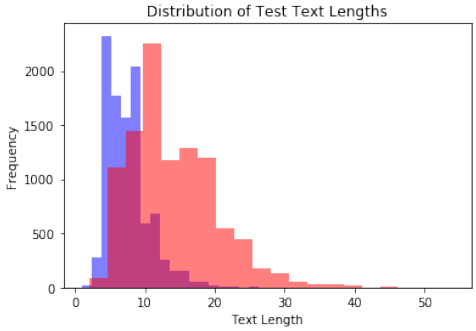

Fig. 5. Test set text length distribution (Mean: 10.72, Median: 9, Max: 54, Min: 1)

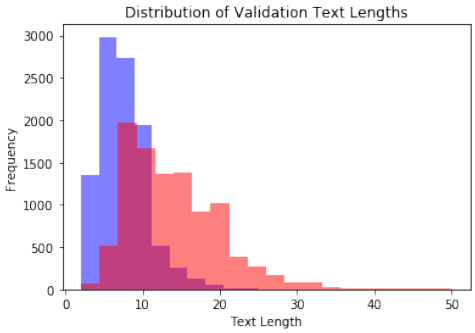

Fig. 6. Validation set text length distribution (Mean: 10.73, Median: 9, Max: 50, Min 2)

initialized each model with "300d reference GloVe vectors (840B token version, Pennington et al. 2014) and fine-tuned as part of training" [2]. Their sum of words method simply took the sums of a given sentence's word embeddings, which were projected to a lower dimensionality with a 100 *tanh* layer. Their sum of words method achieved a testing accuracy of 75.3%. Their RNN and LSTM methods both use recurrent layers with 100 units, leading to 100d sentence embeddings. These methods achieved 72.2%, and 77.6% testing accuracy, respectively.

B. Rockschetal conditional model

In Rockschetal, et al.'s Reasoning About Entailment With Neural Attention, they propose a different rationale to Bowman et al.'s original paper. They first describe the usage of conditional LSTMs, where the first cell state of the hypothesis' sentence encoding LSTM is the final cell state of the premise's sentence encoding LSTM. This basic model uses word2vec word representations, and are then inputted to a 3-way softmax layer, not using the three *tanh* layers as seen in Bowman et al.'s model. This model acheieved a testing accuracy of 80.9%.

C. Rockschetal neural attention model

To further improve the performance of their conditional encoding rationale, Rockschetal et al. utilized word-by-word attention to verify entailment or contradiction of individual word- and phrase-pairs. This is accomplished using an attention mechanism similar to [6]. The method can be described with the following:
k represents the embedding space,
L represents the maximum sequence length,
$W^y, W^h, W^x, W^P \in \mathbb{R}^{kxk}$ represent trainable weight matrices,
$w \in \mathbb{R}^k$ represents a weight vector,
$e_L \in \mathbb{R}^L$ represents a vector of 1s, of length L, $Y \in \mathbb{R}^{kxL}$ represents a matrix consisting of the premise LSTM's hidden states ($[h_1^p, h_2^p ... h_N^p]$). At some timestep t, a final representation of the hypothesis, premise pair is computed as $h^* = tanh(W^P r_N + W^x h_N^H) \in \mathbb{R}^k$. The following equations are used to calculate the weighted representation of the premise based on the current hypothesis token, attention weight vector, and alignment:

$$r_t = Y\alpha_t^T + tanh(W^t r_{t-1}) \in \mathbb{R}^k$$
$$\alpha_t = softmax(w^T M_t) \in \mathbb{R}^L$$
$$M_t = tanh(W^y Y + (W^h h_t + W^r r_{t-1}) \otimes e_L)$$

This final representation is then inputted into a 3-way *softmax* Dense layer for the output prediction. This model achieved a testing accuracy of 83.5%

III. Methodology

A. Data Cleaning

While a typical ML project would involve a large amount of data cleaning, as seen above, the dataset is balanced which allows researchers to focus more on their model creation, and less on data cleaning. However, as seen in the SNLI analysis above, there is 1% of each subset(train, test, validation) that has the label -1. Before model training, this small portion of observations was removed. Furthermore, typical textual cleansing techniques were left out for cleaning this dataset. These techniques include: removing stopwords, and lemmatization/stemming. This is because, intuitively, the nature of NLI is sensitive to single words, and the overall semantic structure of the given sentences. This sensitivity can be imagined with example stopwords like "all", and "not". Also, the sensitivity is showcased with an example premise like "the man gave the boy the 10 dollar bills". If "the" was removed, and "bills" was stemmed to "bill", the premise would read "man gave boy 10 dollar bill". Now it has become unclear whether the man gave the boy 10 bills, each with a denomination of one dollar, or a singular 10 dollar bill, and could legitimately falsify the given premise's observation's label. This potential cause for ambiguity when implementing the removal of stop words, lemmatization, and stemming makes it seem much safer to not implement them.

B. Data Preprocessing

In order to preprocess the SNLI dataset into a format acceptable for each of the four model's implemented, this paper used two methods. The first method was to use Keras's built in preprocessing methods to tokenize and pad each sequence in the dataset. Tokenization allows for textual data to be represented as numerical data in a bag-of-words style. Furthermore, due to the fact that this paper uses neural network based architectures, it is necessary to pad each sequence of text with 0s as neural networks, whether MLP or RNN based, expect the data inputted into it to be of the same length. Therefore, when padding with 0s, each sequence is made into an array of length m where m = the maximum sequence length in the cleaned dataset. Furthermore, due to memeory constraints, only 100 dimension gloVe embeddings [7] were used, even though the Rocktaschel et al. paper called for word2vec embeddings to be used for their conditional, and word by word attention models [4]. As a single model uses *all-mpnet-base-v2* sentence embeddings, the above preprocessing

steps were not taken, as the pretrained model tokenizes each sequence automatically, and encodes 768 dimension sentence embeddings from a list of strings [3].

C. Model Architecture

The four models described below were created with Keras.

1) Basic Bowman et al. Model with all-mpnet-base-v2 Sentence Embeddings: This model uses the basic MLP based model seen in Bowman et al. [2] except instead of using gloVe embeddings and either finding a sequence/sentence's embedding by adding each word embedding, or using LSTM layers to process premise and hypothesis, this proposed model uses Microsoft's MPNet model [8] [3] to find the premise's, and hypothesis' sentence embeddings, both of dimension 768. These are then concatenated, and inputted into three, consecutive, 200-dimension *tanh* layers as seen in *Figure 7*.

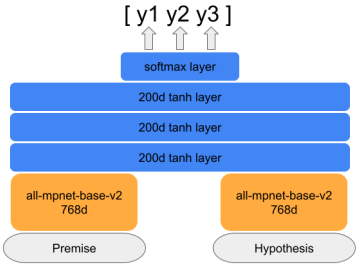

Fig. 7. Basic Bowman et al. model with MPnet Sentence embeddings

2) Basic Bowman et al. Model with LSTM Sentence Embeddings: The model used with LSTMs doing the sentence embedding is a replica of that performed in [2] as seen in *Figure 8*. The only difference is the dimensionality of the word embeddings is 100 instead of 300d.

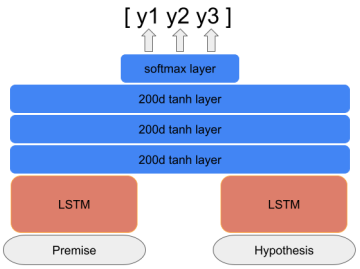

Fig. 8. Basic Bowman et al. Model with LSTM Sentence Embeddings

3) Rocktaschel et al. Model with Conditional LSTM Sentence Embeddings: Rocktaschel et al. propose an intuitive model where the sentence embedding of the hypothesis is conditioned upon the premise's sentence embedding. Here, the final cell state of the premise's LSTM is the first cell state of the hypothesis' LSTM, as seen in *Figure 9*.

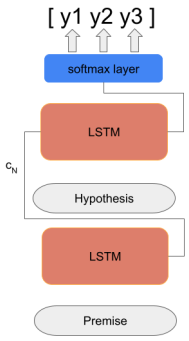

Fig. 9. Conditional LSTM

4) Rocktaschel et al. Model with gloVe word embeddings and Word-by-Word attention: As mentioned previously, the word by word attention model from [4] attempts to align each word in the hypothesis to the premise. This is done by first inputting the premise's sequence through an LSTM, and then for each token in the hypothesis, creating "attention weight-vectors" [4]. This process can be seen from the equations mentioned above, and also in Figure 10. Unfortunately, while this model's replication was attempted, time and the author's knowledge of Keras constraints led to it not being finished. However, one of the author's main goals in this paper was to understand this model, which was accomplished.

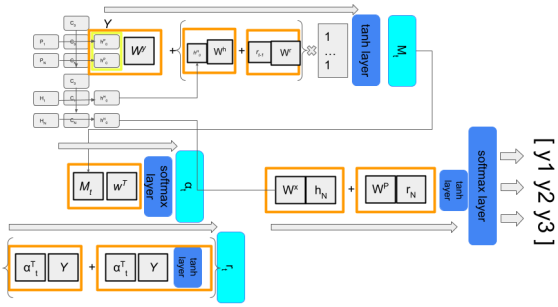

Fig. 10. Word by Word Attention Model

D. Model Training

Unfortunately, due to memory constraints, only 6% of the training set could be used to train each model. Each model was trained using the Keras API for TensorFlow. The loss function used was categorical cross entropy, as the final predictions were of 3 logit format.

1) Basic Bowman et al. Model with MPnet Sentence Embeddings: The MPnet based sentence embedding model was trained for 10 epochs, using the Stochastic Gradient Descent learning algorithm.

2) Basic Bowman et al. Model with LSTM Sentence Embeddings: The LSTM based sentence embedding model was trained for 10 epochs, using the Stochastic Gradient Descent learning algorithm with a batch size of 16.

3) Conditional LSTM model: The conditional LSTM model was trained for 10 epochs, using the Adam learning algorithm.

IV. Results

The models trained for this paper were tested on only 1k test observations (10% of the standard testing set) due to memory constraints.

Model	Train	Test
	Acc	Acc
MPnet SentEmb	0.63	0.63
LSTM SentEmb	0.33	0.33
Conditional LSTM	0.61	0.61

TABLE II
Training, and Testing Accuracies from this paper's experiments

Model	Train	Test
	Acc	Acc
LSTM SentEmb [2]	0.844	0.776
Conditional LSTM [4]	0.835	0.809

TABLE III
Training, and Testing Accuracies from the replicated papers' experiments

Model	Testing F1	Testing F1	Testing F1
	Entail	Neutral	Contradict
MPnet SentEmb	0.67	0.58	0.65
LSTM SentEmb	0	0	0.5
Conditional LSTM	0.6	0.61	0.63

TABLE IV
Testing F1 Scores

V. Discussion

As previously mentioned, only 6% of the training set (34200 observations) was used due to memory constraints. Nevertheless, the final results are interesting. There are two reasons for this. The first is the fact that despite a small training sample, both the conditional LSTM, and MPnet sentence embedding models began to learn, both with testing accuracies over 60%, but the basic LSTM sentence embedding model wasn't able to learn, and was stuck at random guessing (33%). This is interesting due to the small changes made from the basic Bowman et al. model, where the conditional LSTM model removed the 3 *tanh* Dense layers, and connected the final cell state of the premise's LSTM to the hypothesis' first cell state. While it may seem as though the removal of the 3 *tanh* layers, and the inclusion of a conditional LSTM would perform better than the basic model, the high-quality sentence embeddings from MPnet allowed the basic model's architecture to outperform the conditional LSTM.

VI. Future work & Limitations

A glaring limitation of this work is the usage of a small training subset. The extremely small training subset makes it difficult to generalize the results from this paper, but it would be extremely interesting to see how this difference in training would truly affect the performance of the MPnet sentence embedding model. Another next step for this work is to implement the word-by-word attention model with Keras and TensorFlow.

VII. Conclusion

The purpose of this paper was to see how the four models described in this paper compared to each other. While conclusions can't be drawn from the attention-based model as it wasn't created, it is clear that the MPnet sentence embedding model presented in this paper, and the Conditional LSTM model presented from [4] are more performant than the basic LSTM sentence embedding model from [2] due to the fact that they began to learn, and create more accurate predictions when being trained on 6% of the SNLI dataset. Overall, while they are interesting, these results can only be treated as preliminary to future training on the full SNLI dataset.

References

[1] R. Bartolo and B. B. Averbeck, "Inference as a fundamental process in behavior," *Current opinion in behavioral sciences*, vol. 38, pp. 8–13, 2021.

[2] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," *arXiv preprint arXiv:1508.05326*, 2015.

[3] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 11 2019. [Online]. Available: http://arxiv.org/abs/1908.10084

[4] T. Rocktäschel, E. Grefenstette, K. M. Hermann, T. Kočiskỳ, and P. Blunsom, "Reasoning about entailment with neural attention," *arXiv preprint arXiv:1509.06664*, 2015.

[5] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 67–78, 2014.

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[7] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," pp. 1532–1543, 2014.

[8] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mpnet: Masked and permuted pre-training for language understanding," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 857–16 867, 2020.