April 17th 2019

Baseline Models for Entailment Classification

Otman Benchekroun **Sarah Fernandez**

Abstract

Entailment and contradiction are at the very core of natural language modeling. Being adept at understanding semantic implications or inconsistencies is crucial for tasks like translation, image captioning, and text generation. With the release of the SNLI corpus[1], a large dataset containing upwards of 500 000 sentence contradictions, entailments and non-entailments, more complex models have been trained on this task, while baseline models have further been evaluated as well. Our work focuses on establishing and fine tuning the more simplistic baseline models and proving their effectiveness relative to the state of the art. We achieve particularly good results with a simple logistic regression classifier and lexicalized features, outperforming the best LSTM model developed by Bowman et. al.[1] in the original SNLI paper.

Introduction

Understanding logical entailment and contradiction is crucial for the modeling of any language[2]. In natural language inference tasks, it is desired that a model that takes an input text premise and determines whether or not another piece of text (the hypothesis) is valid, based on the original premise. The following are examples of entailments, contradiction and non-entailment, respectively.

Entailment
Premise: A man is driving his child to school.
Hypothesis: The man and child are in a car.

Contradiction
Premise: A man is driving his child to work.
Hypothesis: The child is playing basketball with friends.

Non-Entailment
Premise: A man is driving his child to work.
Hypothesis: The child's shoes are untied.

With the release of the SNLI corpus [1], an abundance of training resources is now available for modeling such natural language inference tasks. Since the original corpus's release, many new models have outperformed previous state of the art benchmarks, such as [3] [4], and the highest performing so far on this dataset achieving an accuracy over 90% [5]. However all these methods use deep neural networks, LSTM sentence embeddings, and require a huge memory footprint to both train and reproduce the model. In this paper, we will go back to the basics of the original SNLI paper, re-establish some of the paper's baselines, as well as modify the paper's original lexicalized classifier model.

Related Work

The most common contemporary approach relating to this problem seems to encoding the premise and hypothesis sentences into vectors [3][4][5], and then using different types of deep neural networks as classifiers.

In [3], they use a Bi-Directional LSTM to encode the sentence vectors. The nature of the LSTM, it's ability to store previous sequence information, makes it suitable for tasks such as sentence embeddings, where a word may have different meanings based on the words that came before it. A word may also have a different meaning based on the words that come after it, and therefore this Bi-Directional LSTM is even better suited for this task. The Bi-Directional LSTM would encode the words into a sentence vector by ultimately looking at all the words coming before or after it.

In [6], Rockstaschel et. al. go against the grain and instead of using a sentence embedding and then a simple MLP for classification, they keep the word-level embeddings and use an attention network that finds correlations between pairings or sequences of words from the premise and hypothesis. This method is somewhat analogous to the lexicalized cross-bigrams and cross-unigram features described in the original SNLI paper by Bowman et. al.

The Dataset

The SNLI Dataset contains 570 000 sentence pairs total, organized as a large list of lists, the latter of which contains two strings. The first member of each sentence pair is the premise, the second member is the hypothesis. The original dataset has three balanced classes: entailment, contradiction, and non-entailment. Following the lead of Bowman et. al, we flattened this problem to a 2 class approach, collapsing the non-entailment class and contradiction class together. This gives a random baseline of 66%, so any accuracy greater than that is a valuable model.

Our Approach

Our approach closely and critically follows that of Bowman et. al.[1]
We start off by using a simple edit-distance based approach, wherein we analyze the use of multiple sentence distance functions; namely, the Jaro Winkler distance function, the Levenshtein distance function, and the Hamming distance function. The Hamming function is proportional to the number of positions in each string that differ between the strings. The Levenshtein function is a representation of the number of single character edits it would take to transform one string into another, and the Jaro-Winkler function operates by using a ratio between the number of matching letters in each string, the length of each string and modifies it by the number of transpositions (letters present in both strings but in different positions). We then use these edit distance functions as features for multiple classifiers: a K-Nearest Neighbor classifier, and a Logistic Regressor and a Linear SVM. We also try multiple combinations of input features, concatenating the 3 distance functions in various orders to capture more important and flexible distance features.

We go on to analyze further non-lexical features. For each sentence pair, we extract the BLEU score, the tokenized length difference and finally the absolute and relative overlap. The BLEU score is a common metric that can score text that has been translated by a machine. This score has been shown to have a high correlation between the score a human would give to rate the quality of the translation [7]. Details for how this score is computed can be found in the Papineni et. al paper [7]. The tokenized length difference could be seen as another pseudo distance metric which could be used

to eliminate hypotheses that are exorbitantly large compared to the premise. Finally, for the absolute and relative overlap, we computed this value twice: Once for all words and all sentences, and another time for only Nouns, Adjectives, Verbs and Adverbs. This absolute and relative overlap feature could be quite useful, and offers a lot of information regarding the similarity of both the hypothesis and the premise that goes beyond simple character similarities.

Finally, we add on lexical features. First we append a bag of words that includes both bigrams and unigrams of the hypothesis. We limit the size of this bag of words to only include the top 40000 most common words due to hardware memory concerns. This has been shown to be useful for features like text classification using a naive bayes model[8]. However this information alone can't possibly capture relationships between premises and hypothesis, as this information is limited to the hypothesis' sentence only. To mitigate this, we add another feature: cross uni-grams. This feature goes through each pair, finds words that share a part of speech tag (e.g. Noun, or Verb), concatenates the two words together and creates a bag of words all of these cross-unigram features. This information can capture relationships between words in the premise and hypothesis and has been shown to have provided a significant boost in accuracy to Bowman et. al. in the SNLI paper. Finally, we do some chi-squared feature filtering on the unigrams, bigrams and cross-unigrams to mitigate memory errors and we test these features on a Logistic Regressor, a linear Support Vector Machine, and a Naive Bayes Classifier.

Results

For our first set of experiments, we trained a logistic regressor and a 10-NN search classifier using the edit distance based features (Table 1). All values here are based off the average validation accuracy of a 3-fold cross validation training. The Levenshtein distance metric proved to be most useful for this classification. However this is no news to jump at, as the accuracies remain quite low compared to the random baseline of 66.7%. This is an underperformance, especially compared to Bowman et. al's reported 71.9% supposedly using the exact same method.

Edit Distance Function	Logistic Regression accuracy	10-NN Search accuracy
Levenshtein	**66.40%**	65.80%
Jaro Winkler	66.10%	66.20%
Hamming	66.15%	66.10%
All 3	65.80%	65.60%

Table 1: Results of experiments regarding different edit distances. The Levenshtein distance function trained on a logistic regressor yields the best results.

Next, we analyze the effectiveness of the non-lexicalized classifiers, using the BLEU score, length difference, and overlap scores. Table 2 shows the results of our

features trained on a Logistic Regressor, a Linear Support Vector Machine and a simple feedforward neural network (MLP). Thankfully, our results show a little promise. They have improved past the random baseline of 66.7 percent, however still lag behind Bowman et. al's 72.2% Non-Lexicalized classifier. The accuracy recorded is once again a 3-fold cross validated average, save for the MLP, which was tested on a held out set of 10 000 sentence pairs. The MLP was a simple series of 4 fully connected layers with 100 neurons each, with one last layer consisting of one neuron for the output.

Classifier	Non-Lexical Features
Logistic Regression	71.80%
Linear SVM	71.65%
MLP	**73.20%**

Table 2: Accuracy of non-lexical features trained on different models. Our simple MLP outperforms all the others.

Finally, let's look at table 3 for the results of our third experiment. Finding the cross bi-grams is quite memory consuming and as a result, a lot of chi-squared filter selection needed to be used to reduce the size of these features considerably. In the end we ended with roughly 400 unigram and bigram features, and another 400 cross-unigram features. As we can see, the addition of both features the performance of our classifier. Going against first intuition, only cross unigrams features did not seem to improve the base accuracy more than unigram, bigram features. This may be because the cross-unigram bag of words matrix is much sparser than the unigram, bigram matrix, and is thus more vulnerable to noise in the dataset. Another interesting result is that while the last two results were lagging behind Bowman et. al's [1] benchmarks, this result, using logistic regression and both cross-unigrams *and* unigrams and bigrams, is one full percentage point *ahead* of Bowman's findings, despite Bowman et. al's larger hardware and memory resources. This may be a result of our feature selection, that strips the features of their least relevant parts and allows the classifier to focus on the meat of the features. It could also be because Bowman et. al also use cross-*bigrams*, not just cross-unigrams, and that result could be bogging down the classifier as well. Bowman et. al attributed much of the accuracy boost to their cross-bigram features, but if an ablation study were to be done, it could be that the boost was largely due to cross-unigrams, with cross-bigrams offering little more than more features for the classifier to process. It is also worth mentioning that this result outperforms Bowman et. al's LSTM network that peaks at 77.6% accuracy.

Classifier	Cross-Unigram	Uni/Bigram	Both
Logistic Regression	74.70%	77.55%	**79.70%**
Linear SVM	74.40%	76.90%	77.80%
Naive Bayes	73.90%	76.05%	76.80%

Table 3: results of our third experiment, Accuracy of models running our lexicalized features.

Conclusion

In conclusion, we found that a simple classifier, given appropriate input features, can perform quite impressively on Natural language inference tasks, given enough training data. We found that by combining a classical bag of words approach with the novel concept of cross-unigrams, a simple logistic regression model could outperform basic LSTM networks on NLI tasks, and offers a solid baseline for any state of the art contemporary model. It is also our belief that a team with more hardware resources could find even better results, laxing the constraints on the chi-squared filter could allow for more features to be analyzed, and could also improve the model's performance.

Statement of Contributions

Otman Benchekroun wrote the code to create the classifiers for each task, implemented manually the cross unigram features as well as ran the final tests for each experiment. Sarah Fernandez implemented the edit distance functions, as well as the data preprocessing. Both members contributed equally to the writing of the report.

References

[1] Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large annotated corpus for learning natural language inference. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. doi:10.18653/v1/d15-1075

[2] Jerrold J. Katz. 1972. Semantic Theory. *Harper & Row*, New York.

[3]Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., & Inkpen, D. (2017). Enhanced LSTM for Natural Language Inference. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. doi:10.18653/v1/p17-1152

[4]Melamud, O., Goldberger, J., & Dagan, I. (2016). Context2vec: Learning Generic Context Embedding with Bidirectional LSTM. *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. doi:10.18653/v1/k16-1006

[5] Chen, D. (2019). Multi-task learning deep neural networks for automatic speech recognition. doi:10.14711/thesis-b1514768

[6]Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. (2015). Reasoning about entailment with neural attention. CoRR abs/1509.06664. http://arxiv.org/abs/1509.06664.

[7]Papineni, K., Roukos, S., Ward, T., & Zhu, W. (2001). Bleu. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL 02*. doi:10.3115/1073083.1073135

[8]Sida Wang , Christopher D. Manning. 2012. Baselines and bigrams: simple, good sentiment and topic classification, *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers*, July 08-14, 2012, Jeju Island, Korea

[9]Lu, Z., Yu, H., Fan, D., & Yuan, C. (2009). Spam Filtering Based on Improved CHI Feature Selection Method. *2009 Chinese Conference on Pattern Recognition*. doi:10.1109/ccpr.2009.5344010