Natural Language Inference – Project 10
Team 25 – YeePM

Introduction

Understanding the project

The concept of Natural Language Inference (NLI) centers around whether we can identify whether one text's information can be inferred from another text. Depending on the initial 'premise', we have to classify a 'hypothesis' into one of three types, based on the confidence score of entailment: Entailment (Logically Follows), Contradiction, and Neutral inference. There are many different implementations to this problem, each varying in the feature vector of the sentence, learning procedure, and score. We will ideally be implementing at least one of the approaches described in the implementation section.

Understanding the scope

Textual entailment is a promising part of natural language processing, and can be used in human readable information extraction, smart document summarization, question answering (as we explain in the additional dataset that we have included) etc. Typically, NLI and TE are parts of larger NLP systems, such as word processors and fact checkers.

Datasets

We will be using four datasets for evaluating our model. The provided SNLI, Sick, and MultiNLI datasets have highly similar formats, where pairs of sentences are provided, as well as their class and entailment, per line. The SciTail dataset contains multiple choice questions with the correct answer provided.

- SNLI: 550152 training vectors, 10000 test vectors, 10000 validation vectors
- Sick: 4439 training vectors, 4906 test vectors, 495 validation vectors
- MultiNLI: 392702 training vectors, 10000 test vectors, 10000 validation vectors
- SciTail: 23587 training vectors, 2126 test vectors, 1304 validation vectors

Implementation

Pre-processing

The pre-processing steps of the data are dependent on the type of final vectors that we use for classification, and thus this implementation can be more fine-tuned or generalized based on the type of vectorization/language model that we use. Some steps that we will consider in pre-processing are:

1. Tokenization
2. Punctuation, Stopwords, and Keywords
3. Stemming and Lemmatization
4. Phrasal Verbs and Idioms

Models

Neural Network Approach: A common approach taken towards textual entailment is deep learning, via a neural network. For this, we first vectorize our pre-processed data, and the GloVe (Global Vectors) representation is very suitable for the task as it can aggregate words and their co-occurrences from the corpus and generate linear substructures that help capture context more effectively than TF-IDF or word2vec.

A recurrent neural network would be ideal in this situation since it reuses input along with a memory to be passed ahead to the next input's calculations, hence remembering context. Even this, however, drowns out the original input, and a more sophisticated node within the RNN could be required, the LSTM, which has the flexibility of keeping information in short- and long-term memory. Using a Bi-directional RNN will allow the model to see the input pair vectors separately as well as within the context of one another.

BERT Approach: A prominent framework that is used in textual entailment is the BERT (Bi-directional Encoder Representations from Transformers) language model. It is preferred over neural networks as it isn't sequential in terms of training the data, which allows for a much larger train size, and thus better results. The BERT LM is more effective than GloVe representation, used above, as it splits the corpus into word-pieces that are completely within the language's vocabulary, thus reducing ambiguity and noise generated in GloVe.

BERT works by understanding the context of the word with respect to the sentence, by considering all neighbouring words rather than looking at them individually. It employs the concept of 'self-attention mechanism, that allows a word to alter its meaning based on different references in a sentence. In addition, as it is bi-directional, it can refer to context of a word that occurs later, in advance. All of these factors suggest that the BERT model can outperform the LSTM model, given sufficient training data. Further, optimized BERT models, such as RoBERTa, or DeBERTa, can be considered to optimize our performance.

Alternative Approach: While most methods depend on the usage of deep text representations or LSTM networks, an alternative method we have in mind to implement the project is the paper 'A Decomposable Attention Model for Natural Language Inference' by Ankur P. Parikh et al. Given that entailment needs to be done in a short context, the problem can be tackled by a process of aligning local text substructures followed by aggregation. Each sentence is represented as a list of tokens (which in turn are word embedding vectors), and the premise and hypothesis are sent down the following pipeline:

- **Attend:** The sentences are soft aligned using an attention mechanism. Pairwise alignment scores are computed using feed-forward net followed by a dot product. Using weighted sums of these scores, alignment degree of subphrases can be extracted out.
- **Compare:** The aligned subphrases are compared with their counterparts using another feed forward net. For every word in the two sentences, a single vector is received as output.
- **Aggregate:** The two sets of comparison vectors from the previous steps are summed over, to get a single aggregate vector for both sentences. They are sent to a final classifier which is a combination of feed forward network and a linear layer. The output is a score for each of the three classes (contradiction, neutral, entailment) and consequently the prediction is the argmax.

The merit of this model is that it is computationally cheaper compared to LSTMs. Further, the paper also presents that with considerably fewer parameters, it achieves state-of-the-art results. In conclusion, the model is built on the intuition that pairwise comparisons of the words from both sentences gives more insight than considering the two as sentence level entities.