Generating Natural Language Entailments and Contradictions

Ryan Sie

1 Introduction

This paper explores the use of deep learning methods for a generation task related to Natural Language Inference (NLI). NLI can be described as the general task of determining the particular semantic relationships between multiple sentences - for instance, whether one particular sentence can be inferred from another, or whether two sentences are contradictory. The ability to automate NLI can be very useful for aiding various NLP tasks like information extraction, question answering, and machine translation.

The first task is the generation of entailments and contradictions. Given an input sentence A, we wish to produce a sentence B which A **entails** and a sentence C which A **contradicts**. Informally speaking, sentence a **entails** sentence b if when a is true, it follows that b is true. Conversely, sentence a **contradicts** sentence b if when a is true, it follows that b is false. If neither of these relationships hold, we say that a and b are **neutral**. A more formal representation of these ideas is discussed in section **4**.

2 Previous Work

The models for the generation task are standard sequence-to-sequence models [1] which use the LSTM [2] recurrent unit in encoder and decoder recurrent neural networks (RNNs). The use of sequence-to-sequence models has seen great success in a wide range of NLP tasks, and is a natural choice for a generation task like this. Indeed, this particular model architecture has been previously successfully implemented for the same generation task described here by Kolesnyk et al. [3].

3 Data

For both generation tasks, use the Stanford Natural Language Inference (SNLI) corpus [6], which consists of pairs of human-produced input and hypothesis sentences, along with corresponding classification labels of entailment, contradiction, or neutrality between the two sentences. We train one model to generate entailments, and another to generate contradictions, only differing in the training data used. After splitting into train/dev/test sets there were roughly 180,000 training examples per model.

We also make use of the GloVe [7] pre-trained 200 dimension word embeddings in each model.n

4 Linguistic Background

In the linguistic subfield of semantics, the relationships of **entailments** and **contradictions** between sentences in natural language are well studied phenomena. A common way to formalize the representation of 'declarative' natural language sentences are as sets of possible worlds in which they are true, in the mathematical sense of *set*. A possible world can simply be thought of any particular arrangement of circumstances in the real world that can be imagined. For our purposes it will suffice to describe a possible world w in words. For instance, in one world you might own a goldfish, while in another, you might not. Then a sentence $S = \{w_1, ..., w_k, ...\}$ is a simple set containing possible worlds w_i in which the sentence is true. Note that this set may be infinite (in fact, many are), and even empty. To help illustrate this formulation, below are some example sentences with descriptions of some possible worlds they contain.

(1) I ate breakfast
   a. A world where I ate a bagel for breakfast
   b. A world where I ate cereal for breakfast

(2) The sky is not blue
   a. A world where the sky is purple
   b. A world where the sky is white

(3) The sentence is true or false
   a. All possible worlds

(4) The sentence is both true and false
   a. No possible world

Examples (3) and (4) would correspond to the sets W and $\varnothing$, respectively, where W is the set of all possible worlds. As a convention, for a sentence A we will denote S_A as the set of possible worlds in which A is true. With this formulation setup, we can now introduce the following definitions:

Definition 4.1. Let A, B be two sentences. We say that A **entails** B if when A is true, it follows that B is true. This occurs when every possible world in S_A is a possible world in S_B, or $S_A \subseteq S_B$.

Definition 4.2. Let A, B be two sentences. We say that A **contradicts** B if when A is true, it follows that B is false. This occurs precisely when every possible world in S_A is not a possible world in S_B, which is equivalent to saying every possible world in S_B is not a possible world in S_A. These conditions are satisfied precisely when $S_A \cap S_B = \emptyset$.

Definition 4.3. Let A, B be two sentences. We say that A and B are **neutral** if the truth value of one sentence has no bearing on the truth value of the other. This occurs when some but not all possible world(s) in S_A is a (are) possible world(s) in S_B, and vice versa. This is satisfied when $S_A \cap S_B \neq \emptyset, S_A \nsubseteq S_B$, and $S_B \nsubseteq S_A$.

5 Model

5.1 Generation Model

The generation task involves 2 sequence-to-sequence models with the same architecture, differing only in the subset of data used to train each one (pairs labeled 'entailment' vs

'contradiction'). As mentioned above, the architecture involves an encoder and decoder recurrent neural network (RNN), both of which use the LSTM [2] recurrent unit. We use the GloVe pre-trained word embeddings as well for our embedding layer [7].

An LSTM cell at time t takes in a previous cell state $\mathbf{c}_{t-1}$ and hidden state $\mathbf{h}_{t-1}$, as well as the input at time $t, \mathbf{x}_t$, and produces the next cell state $\mathbf{c}_t$ and hidden state $\mathbf{h}_t$:

$$\mathbf{LSTM}((\mathbf{c}_{t-1}, \mathbf{h}_{t-1}), \mathbf{x}_t) = (\mathbf{c}_t, \mathbf{h}_t)$$

5.1.1 Encoder

The encoder RNN can then be described by the following algorithm to process an input sentence $S = w_1 w_2 ... w_n$.

$$\text{Initialize } \mathbf{c}_0, \mathbf{h}_0 \tag{1}$$
$$\text{for i = 1, ..., n} \tag{2}$$
$$\mathbf{x}_i = \text{GloVe}(w_i) \tag{3}$$
$$(\mathbf{c}_i, \mathbf{h}_i) = \mathbf{LSTM}((\mathbf{c}_{i-1}, \mathbf{h}_{i-1}), \mathbf{x}_i) \tag{4}$$

The encoder RNN thus produces n pairs of cell and hidden states $(\mathbf{c}_1, \mathbf{h}_1), ..., (\mathbf{c}_n, \mathbf{h}_n)$, which will be useful in the decoder RNN.

5.1.2 Decoder

The architecture for the decoder is analogous to that of the encoder. We use the final hidden and cell state from the encoder as the decoder's initial ones. We also make use of a prediction MLP, MLP_{pred} for the actual generation at each timestep. During training we use teacher forcing to give the true labels.

The following algorithm describes the behavior of the decoder RNN during training, assuming that $T = u_1 u_2 ... u_m$ is the observed hypothesis sentence corresponding to the premise sentence $S = w_1 w_2 ... w_n$, and t_j is the index of the nonzero entry of the one-hot vector of u_j.

$$\text{Take } \mathbf{b}_0 = \mathbf{c}_n, \mathbf{g}_0 = \mathbf{h}_n \tag{5}$$
$$\text{for j = 1, ..., m} \tag{6}$$
$$\mathbf{x}_j = \text{GloVe}(u_j) \tag{7}$$
$$(\mathbf{b}_j, \mathbf{g}_j) = \mathbf{LSTM}((\mathbf{b}_{j-1}, \mathbf{g}_{j-1}), \mathbf{x}_j) \tag{8}$$
$$\mathbf{y_j} = \text{MLP}_{pred}(\mathbf{g}_j) \tag{9}$$
$$\text{loss}_j = -\log(\mathbf{y_j}_{(t_j)}) \tag{10}$$
$$\text{loss}_{total} = \sum_{j=1}^{m} \text{loss}_j \tag{11}$$

Steps (8) - (10) involve the computation of context vectors $\mathbf{v}_j$ for word-by-word attention, while steps (13) - (15) takes care of the generation, and computing the corresponding loss with negative log likelihood with the original sentence T. We then perform Stochastic Gradient Descent to train the model.

References

[1] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS14, page 31043112, Cambridge, MA, USA, 2014. MIT Press.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):17351780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

[3] Vladyslav Kolesnyk, Tim Rocktäschel, and Sebastian Riedel. Generating natural language inference chains. *CoRR*, abs/1606.01404, 2016. URL `http://arxiv.org/abs/1606.01404`.

[4] Shuohang Wang and Jing Jiang. Learning natural language inference with LSTM. *CoRR*, abs/1512.08849, 2015. URL `http://arxiv.org/abs/1512.08849`.

[5] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. Reasoning about entailment with neural attention. *CoRR*, abs/1509.06664, 2015.

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

[7] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.