Entailment in language pragmatics

Prashant Bahuguna

August 2022

1 Introduction

Entailment is a concept that refers to a specific kind of relationship between two sentences. It is a type of semantic dependence that holds between one sentence and another. One sentence can entail another one when the truth of the first guarantees the truth of the second. Moreover, the falsity of the second guarantees the falsity of the first.
For example:
1-The earth goes round the sun. (entails)
2-The earth moves.

Textual entailment has been recently defined as a common solution for modelling language variability in different NLP tasks. Textual Entailment is formally defined as a relationship between a coherent text **T** and a language expression, the hypothesis **H**. **T** is said to entail **H** $(\mathbf{T} \rightarrow \mathbf{H})$ if the meaning of **H** can be inferred from the meaning of **T**. An entailment function **e(T,H)** thus maps an entailment pair T-H to a true value (i.e., true if the relationship holds, false otherwise). Alternatively, **e(T,H)** can be also intended as a probabilistic function mapping the pair T-H to a real value between 0 and 1, expressing the confidence with which a human judge or an automatic system estimate the relationship to hold. For example "Facebook acquired Instagram" entails "Facebook owns Instagram". One of the application of Textual Entailment is in Information Retrieval

The project aims at defining a model that detects possible entailments given an input utterance. In this report I aim to seek answers to what entailment between two sentences fundamentally mean and what it might take to uncover the same.

Hypothesis :
It is always possible to change the state of one sentence to the other sentence given sufficiently large layers of implicit or explicit knowledge. But, in our terminology we use the term entailment if we can reach from one sentence to the other sentence using very few layers of abstraction.
Example:
Sentence-1: I was feeling cold
Sentence2 : I brought a jacket with me.
Sentence2' : I crafted a wooden spear.

Which of the 2 Sentences above entails or agrees with the Sentence1?

With the given line of thoughts, it can be possible that both the sentences (2 and 2') agrees to Sentence1.

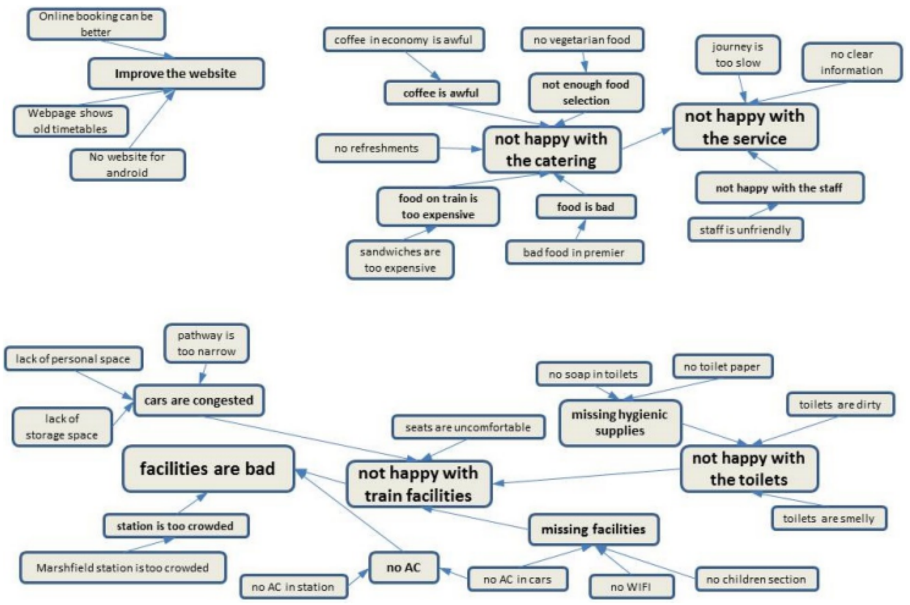
One can make quickly make sense that Sentence-1 entails Sentence-2 with the knowledge:
I was feeling cold $->$ I needed warmth retention $->$ clothes can be helpful $->$ I got a jacket.

However, the sentence-2' can as well entail Sentence-1 with some thought process as:
I was feeling cold $->$ I needed warmth retention $->$ I wanted to hunt the animal for its hide that can be used to shield from extreme low temperatures $->$ I crafted a wooden spear.

It is clear that entailment can take many forms, sometimes through the usage of synonyms while other times using the implicit and explicit knowledge in the background that revolves around Sentence1.

2 Textual graphs

Fragmenting a sentence into smaller parts can facilitate in finding semantic similarity and possibly entailment. As described in the paper Textual Entailment Graphs[1], arriving at the entailment through the usage of fragments can be a powerful technique.
To summarize-
1) We start with constructing the fragments of a sentence by removing the modifiers.
2) Later, if 2 smallest fragments of different sentences are entailing or synonymous, they are merged.



Example of a merged Textual Entailment Graph

However, the problem of linking smaller fragments of two different sentences still remains to be a fundamental problem that is central to the topic of this report. Even if we are able to break down the sentences into smaller chunks, it is much less clear what fundamental criterion should be employed to declare one sentence/phrase as an entailment of the other.

3 Approach

The training was conducted using the The Stanford Natural Language Inference (SNLI) Corpus. It consists of 3 classes - neutral, entailment and contradiction. The train dataset comprised of around 55k pair of sentences and validation dataset comprised of roughly 10k pair of sentences. A batch size of 128 was used to fine tune the model described below. The training was executed on Nvidia A-100 google cloud platform .
The major outline of my line of investigation is to check if changing the tokenized input sentences by employing textual graphs could bring any improvement in the finetuning of the pretrained model.

3.1 Tokenized Input

While it is not clear if entailment is inherently commutative in nature, I assumed it to be non commutative.
In the first baseline method the sentences were encoded in the following manner inside the tokenizer of the model

[CLS] *Sentence1* [SEP] *Sentence2* [SEP]

Therefore, sentence1 and sentence2 are used as it is.
whereas, in the second training, the sentences were tokenized as follows thereby altering the sentence2 while keeping sentence1 same so that loss of information can be avoided.

[CLS] *Sentence1* [SEP] *Sentence2'* [SEP]

Sentence2' was obtained using the following algorithm-
For example let the sentence 1 be:

A black race car starts up in front of a crowd of people.
fragments: a) car starts up in front of a crowd of people,
b) A black race car starts up in,
c) car starts up in
entities: a) black race car b) people

Sentence 2 be

A man is driving down a lonely road.
fragments: a) man is driving down a lonely road
b) A man is driving down road
c) man is driving down road
entities: a) man b) lonely road

Each sentence has a ROOT which is not dependent on any token. All the tokens which depend on the other tokens are called as modifiers. I started with identifying the root and then all the tokens which depend on the root. Like this, I go up the chain until I recover the original sentence.

As presented above, if the len of tokens of sentence2 exceeds 1.5 times the length of the sentence1 (whose entailemt is to be found), I proceed to choose the fragments based on the criterion which is, to take the sentence that has the highest intersection and the lowest length. For example if fragment1 and fragment2 has same number of entities then that sentence is chosen which has the lowest length.
The whole premises is based on the assumption that attention mechanism can suffer from long range correlations. Therefore, if the modifiers / fragments which contain the entities are used it may improve the metrics.

3.2 Model

DEBERTA: Decoding Enhanced Bert with Disentangled Attention
DEBERTA is a Transformer-based neural language model pretrained on large amounts of raw text corpora using self-supervised learning. Like other PLMs, DeBERTa is intended to learn universal language representations that can be adapted to various downstream NLU tasks. DeBERTa improves previous state-of-the-art PLMs (for example, BERT, RoBERTa, UniLM) using two novel techniques:
i) a disentangled attention mechanism: Where each word is represented using two vectors that encode its content and position, respectively, and the attention weights among words are computed using disentangled matrices on their contents and relative positions, respectively
ii) an enhanced mask decode: It is used to incorporate absolute positions in the decoding layer to predict the masked tokens in model pre-training. The disentangled attention mechanism already considers the contents and relative positions of the context words, but not the absolute positions of these words, which in many cases are crucial for the prediction

THE DEBERTA ARCHITECTURE
For a token at position i in a sequence, we represent it using two vectors,$\{H_i\}$ and $\{P_{i|j}\}$ which represent its content and relative position with the token at position j, respectively. The calculation of the cross attention score between tokens i and j can be decomposed into four components as:

$$(A_i) = \left(H_i, P_{i|j}\right) X \left(H_j, P_{j|i}\right) \tag{1}$$

$$= \left(H_i H_j^T\right) + \left(H_i P_{j|i}^T\right) + \left(P_{i|j} H_j^T\right) + \left(P_{i|j} P_{j|i}^T\right) \tag{2}$$

That is, the attention weight of a word pair can be computed as a sum of four attention scores using disentangled matrices on their contents and positions as content-to-content, content-to-position, position-to-content, and position-to-position

That is, the attention weight of a word pair can be computed as a sum of four attention scores using disentangled matrices on their contents and positions as content-to-content, content-to-position, position-to-content, and position-to-position.
Taking single-head attention as an example, the standard self-attention operation can be formulated

as:

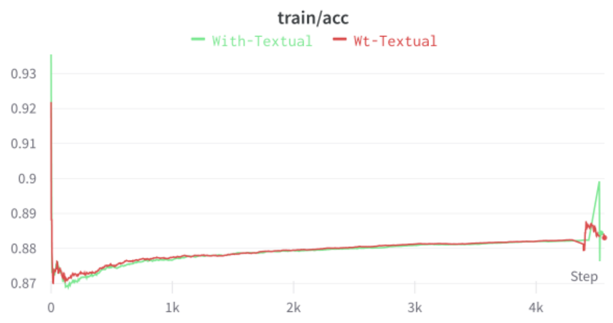
$$Q = HW_q, K = HW_k, V = HW_V, A = \frac{QK^T}{\sqrt{d}} \tag{3}$$

$$H_o = softmax\,(A)\,V \tag{4}$$

where $H \in R^{N\times d}$ represents the input hidden vectors, $H_o \in R^{N\times d}$, the output of self-attention, $W_q, W_k, W_v \in R^{d\times d}$ the projection matrices, $A \in R^{N\times N}$ the attention matrix, N the length of the input sequence, and d the dimension of hidden states.

3.3 Metrics

To test the hypothesis, just one epoch was enough to obtain the conclusion and the drawbacks of the current approach. Therefore, after running the training cycle for1 epoch in each case I obtain the following metrics.



Experiment	Train(acc)	Val(acc)
With Textual	0.88030	0.8914
Without Textual	0.88293	0.8916

4 Conclusion and Further developments

While only roughly around 11% of sentences differed by length more than 1.5 therefore, the hypothesis can be better tested on bigger sentences. The result instead of having an improvement suffered a very slight decrease in performance.
The task of uncovering meaning of entailment continues to be a hard topic. Other directions where answers might reside is:
1) Using category theory $(https://golem.ph.utexas.edu/category/2018/02/linguistics_using_category_the.html)$ to map the objects by constructing the morphisms. However, based on brief reading of the blog mentioned, I came to conclusion that it would require knowledge of monoids along with some basic knowledge of category theory which couldn't be accomplished in the time frame.
2) To associate each sentence with a diagram. Diagram consists of finite objects. The benefit of this approach would be that, we can employ graph based vision techniques to find similarities in different sentences because now we dont have tokens but the visual objects in this case. Although

it is still a speculation but I did make some progress in coming up with general rules to build the diagrams. It will be interesting to see how far this idea can go.

5 References

1) Textual Entailment Graphs, Lili Kotlerman and others.