LT2326: Project report: Contradictory , my dear Watson

Yoann Morello
University of Gothenburg
gusmoreyo@student.gu.se

CONTEXT

Predicting sentences entailment consists, given a pair of sentences called premise and hypothesis, to assign them a class : Entailment, Neutral or Contradiction, depending on the second being a consequence of the first.
This task was still considered as extremely difficult[1] some years ago. More recently Poliak and Adam proposed it as a general metric to evaluate NLP systems, underlying how fundamental it is in relation to understanding human languages [2]. Bert and all the derived models (Albert, DistilBert...) are successfully used today for entailment prediction on a number of datasets. We decided to use the Kaggle database provided with the competition "Contradictory, my dear Watson", as it seemed to provide challenging pairs while the data was high quality and easily available. We focused on the English subset that is composed of 12000 training pairs.

I. THE CONCEPT

The concept we decided to explore is rooted in [3]. The URN architecture described in the article has shown promising results on formal languages. Could they extend to textual entailment ? However the size of the training set indicates that our model had no chance to see enough occurrences of all the words to embed them correctly. We thus decided to include information about the words embedding extracted from a pretrained Albert model. Subsequently, the URN layer should play the role of a sentence encoder that would aggregate information coming from a sequence of words into one tensor that could serve as input for the classification layer. The question about the architecture became : how to integer a word

embedding from some Bert model with some URN layers in a way to make use of some interesting mathematic properties of the last. These will be detailed in the chapters below, as well as the mentionned properties. Efforts were mainly focus at experimenting with potential architectures. We want to draw attention to this precision because had the objective been to optimise learning for the task at hand, we would have spent time on data engineering, in particular some data augmentation which proved so efficient in [4]. We would also have spent more time on exploring the space of hyperparameters, which we have in this case reduced to a minimum so that our models remain as comparable as possible, and differ essentially only in the architecture.

II. TWO BASE MODELS FOR COMPARISON

We first implemented the smallest version of Albert from huggingface.co with and without fine-tuning. Both versions were trained 4 epochs and scored high on the validation data. The variant without fine-tuning score slightly higher with 82% of accuracy on the validation set, while the fine-tuned model reaches 92%. Both quickly overfits (from the second epoch), and the validation loss increases in the 2 subsequent epochs. Classification was done on the

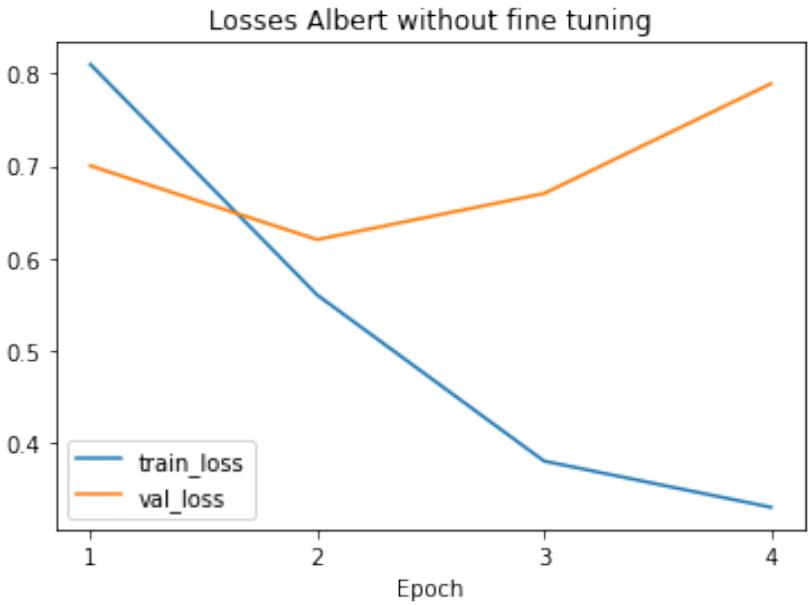

Figure 1. Albert losses

768 dim embedding of the first (separation) token returned in the last hidden layer of Bert. Sentences were fed in pairs and this first separation token is usually used to represent the learned contextual embedding for a sentence (or a couple of sentences in our case).

III. THE NAIVE URN+BERT MODEL

The aim with this one was to integer the information from the embedding of each word (without the separation tokens). Each sentence was processed separately, the way a LSTM layer would, but using instead an URN layer. Then, tensors from both sentences are concatenated before being classified by a single linear layer. The architecture is represented in the following diagram:

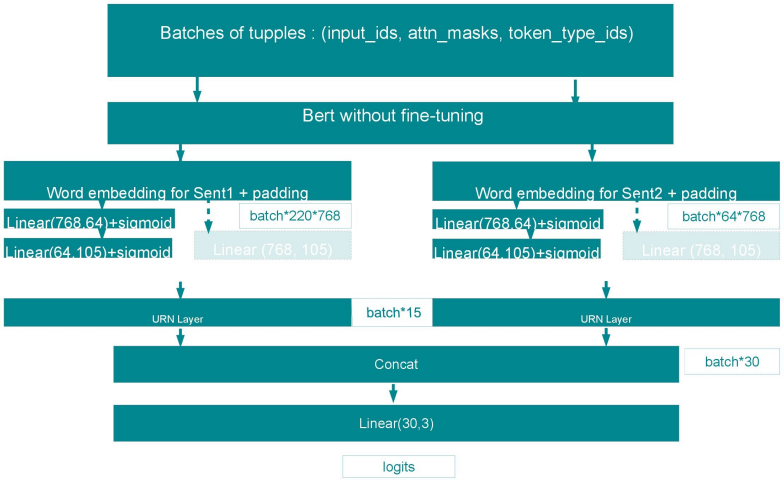

Figure 2. Albert+URN

The dashed lines show a first try, where the information from the Bert embedding was projected in 105 dimensions by a single linear layer. It failed to learn anything. In a second experiment, this layer was replaced by a couple of layers separated by a sigmoid. This time, learning happened. Our model showed a relatively high accuracy on the validation set (75%)

IV. THE URN LAYER(S)

The basic layer takes in input a sentence where the words are embedded in $n(n-1)$ dimensions and a n vector h. Each word in the sentence is then mapped to an orthogonal matrix of dim n*n in three steps:

1. the n(n-1) coordinates of the vector are spread out in an upper triangular matrix T of size n*n
2. $M = T - T^t$ transforms it in an antisymmetric matrix (which eigenvalues are purely imaginary)
3. $exp(M)$ returns an orthogonal matrix as its eigenvalues are the exponential of those of T, but the exponential of a purely imaginary number has norm one. The figure below illustrates the process.

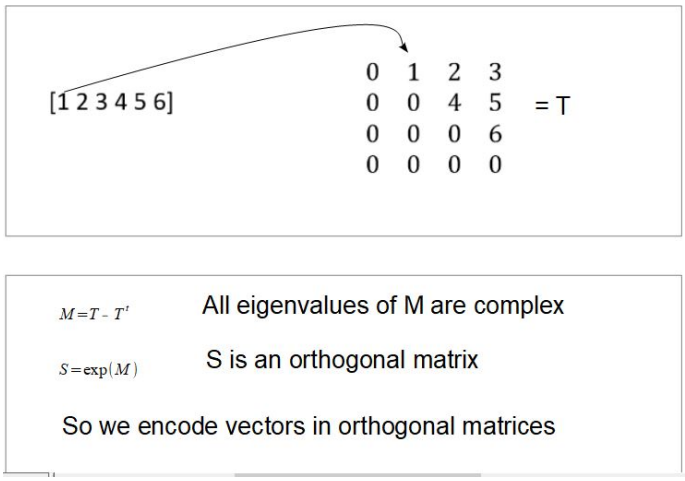

Figure 3. URN principle

This way each word in the sentence can be viewed as a transformation applied to some initial word/semantic state. It allows to encode a sequence in a very straight forward manner (by just multiplying the matrices representing the words that compose it). Furthermore URN encoding presents 2 attractive mathematical properties:

- The first, that we had no time to take profit of with this project, is that matrices learned are easily interpretable.

- The second, which we tried to use in our next architectures is that the cosine similarity of two vectors h and h', $\langle h, h' \rangle$ stays the same after both these vectors are processed through an URN (and a same sentence)

Our next models (several variations were tested) encode one after the other each word in both sentences by projecting the Bert embedding to a small dimension (15) space and processing it through the rest of the sentence via an URN. We describe them in the next section.

V. ALBERT+URN REFINED

(One of) the issues with training the URN on such a database is that numerous words will never be encountered in the training set. Nevertheless we would like our model to use the cosine similarity of these new worlds with some known ones to generalise the heuristic it may have learned. For example, if the model has learned that in the pair : "All birds fly. Some bird fly." the second is consequence of the first, we want it to identify the new hypothesis : "all ravens fly." as a rightful consequence of the same premise. One way to do this would be to encode all words of each sentence but one in orthogonal matrices, so that the encoded sentence would become $O_1, ..O_{k-1}, w_k, O_{k+1}...O_m$ where O_i are orthogonal matrices, w_k is a vector, m is the number of words in the sentence and k is the position of the untouched (not transformed in an orthogonal matrix) word in the sentence. Then, to benefit from the property that 2 vectors h and h' $\langle \prod_{i=k+1}^{m} O_i h, \prod_{i=k+1}^{m} O_i h' \rangle = \langle h, h' \rangle$, w_k needs to be projected onto a space of the same dimension as the orthogonal matrices. This would be repeated for each word of both sentences.

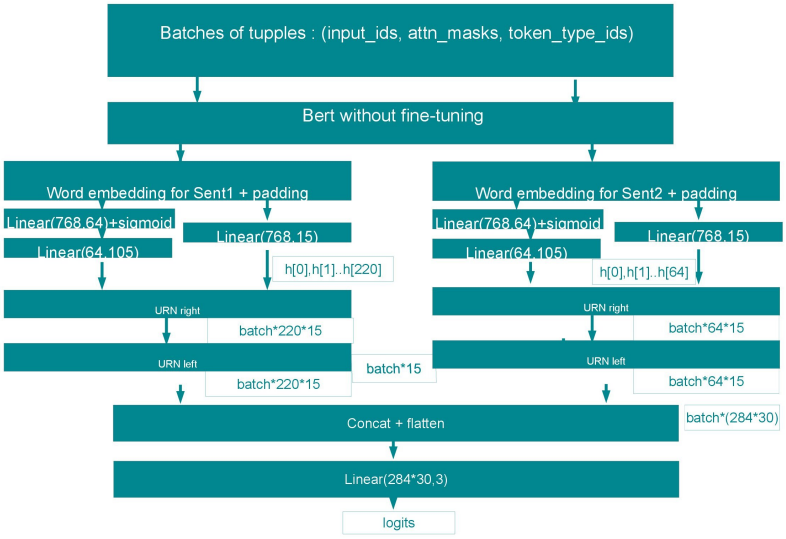

Figure 4. Albert+URN refined

URNright and left are adaptation of the URN layer. They take in input a (batch of) sequence of words in 105 dim (the sentence) and a sequence of vectors h of the same length and of dim 15. Each word in h is then processed through all the words in the sentence (transformed in orthogonal matrices) to its right, returning an output of size: batch*SentLen*15. URNleft does the same on the left. The information from both sentences is then either concatenated or added, before being classified by a single linear layer or several separated by sigmoids.

VI. RESULTS AND DISCUSSION

Many variants of the last model failed to learn anything. One of them seems nevertheless to be doing so, while slowly. However, the double "for loop" inside the URNright and left and the matrices exponential result in the training being so long that a single epochs takes more than 2 hours. Some repeated disconnections from the servers in the recent days, paired with a frequent full occupation of the GPUs discouraged us to explore further this last model in the time limits. Despite its failure to approach Bert's results, the creation of these models has been a very instructive process that has allowed us to reflect and experiment on the ways in which models are constructed and the motivations that can lead to the choice or construction of an architecture. We hope to pursue this exploration in the future. On the other hand, the

ability of the naive Albert + URN model to learn, even if not so good as Bert alone, seemed very interesting from the point of view of possible interpretations. Some more work would be needed in this direction to try and interpret what it has learned. Among other questions, we are left wondering if the Bert contextual embedding would just propagate into each word the information about the sentence in a way that our model was able to reuse, or if it learned some ways that a word impacts a sentence. To check this an easy step that we had no time to implement would be to replace Bert embedding by some bag of vectors like Glove and check if the results hold.

REFERENCES

[1] J. Bos and K. Markert, Recognising textual entailment with logical inference, in *Human Language Technology - The Baltic Perspectiv* (2005).

[2] A. Poliak, A survey on recognizing textual entailment as an NLP evaluation, in *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems* (Association for Computational Linguistics, Online, 2020) pp. 92–109.

[3] *Algebraic Structures in Natural Language*, Chap. Unitary Recurrent Networks:Algebraic and LinearStructures for Syntax.

[4] N. M. G. D. G. A. S. A. Lingam V, Bhuria S, Deep learning for conflicting statements detection in text, .