Homework 3: Coreference Resolution

Florin Cuconasu 1835605

Abstract

Coreference resolution is the task of finding all expressions that refer to the same entity in a text. In this paper, I address the problem at the mention linking level, knowing that the coreference mention will always be a pronoun. All the architectures I have considered take advantage of pre-trained BERT (base and large), which is then specialized with a classification head. The best model improves the baseline by adding complexity to the mentions representation, also ensuring faster convergence.

1 Introduction

Coreference resolution can be formulated as the problem of solving the mention proposal and mention linking tasks. The mention proposal part aims to identify all candidate entities in a text; instead, in the mention linking step the goal is to understand which pairs of mentions correspond to the same entity. In the following work I focus on a specific aspect of the mention linking task, where the objective is to find the correct entity mention for a given ambiguous pronoun.

2 Dataset

The analysis was performed on the Gendered Ambiguous Pronouns (GAP) dataset by Google AI (Webster et al., 2018) using 2999 training and 454 validation samples. The structure of the dataset allows to easily treat the pronoun coreference resolution task as a classification problem, where the model must identify if the ambiguous pronoun coreferences one of the two given candidate mentions (A, B) or neither.

The dataset is pretty clean, therefore I did not apply any special changes.

3 BERT

The backbone of all experiments is BERT (Devlin et al., 2018a) used to learn contextual embeddings. The advantage of using transformer models is that they have already been trained with a huge amount of data, hence they have already encoded many types of linguistic information. Moreover, BERT uses a WordPiece tokenizer, which is very useful with less frequent words, such as proper names, since they are divided into different sub-tokens. Almost all models I tested share the same BERT module, which is then specialized with a classification head. The classification head takes BERT embeddings as input and processes them to output a probability distribution over the label classes. In general, BERT uncased (base and large) was used to fine-tune the models.

The difference between BERT base and BERT large is on the number of encoder layers. BERT base model has 12 encoder layers stacked on top of each other whereas BERT large has 24 layers of encoders stacked on top of each other. Since the large version was trained on a much larger set of data, it also requires more computational power to fine-tune, thus I tried fewer tests on it with respect to the base one.

4 Models Description

This section shows the various architecture I have built. All models use: accuracy as metric, Cross Entropy as loss function and Adam as optimizer. I also tried SGD with different momentum values, but the optimal results were achieved with Adam and a learning rate between 8e-6 and 5e-6.

The input text was tokenized using different approaches:

- Mention tags (Image 1): Mentions (A, B and the pronoun) are delimited using tags ($< A >$, $< /A >$, $< P >$ etc.).

- Truncation (Image 2): The text is truncated at the last mention (A, B, pronoun) encountered; in case the last mention is an entity, the pronoun is appended at the end, in such a way that it will always be after the other mentions.

In the following experiments, I will refer to mention tags and truncation with the above meaning.

4.1 Baseline

The baseline approach follows the guidelines of the CorefSeq model by Chada that faces the problem as a sequence classification task. The input is the text passage with truncation and without mention tags. The sequence features are the concatenation of the token embeddings corresponding to the A, B and the ambiguous pronoun spans. Except for the pronoun that is characterized by a single token, hence a single embedding u_P, the other entities may be composed by different sub-tokens. Therefore, I decided to compute the span embedding $\bar{u}$ as the mean of all the sub-tokens embeddings that make up the entire span.

$$v = [\bar{u_A}; \bar{u_B}; u_P]$$

The sub-token embeddings are the result of the last BERT layer. I have tried to use the sum of the last four layers or their concatenation, but they did not lead to better results, considering the increased training time. The v features are then fed to a feed-forward neural network (FFNN) with a LeakyReLU activation function and dropout.

4.1.1 Training & Hyper-parameters

The baseline architecture was trained using both BERT base and large. In the first case, all layers were fine-tuned; in the second case the first 12 layers were frozen. In the BERT base model, I also considered the cased version of the tokenizer, that outperforms the uncased version even if they use almost the same hyper-parameters. With BERT base, I applied a step scheduler that decays the learning rate after 3 epochs by 0.5. This addition helps in limiting overfitting, such as the dropout of 0.1 applied before the last classification layer.

For what concerns the BERT large version, it reaches even better results with only 3 epochs and no scheduler.

4.2 Multiple Choice

The multiple choice approach (Chada, 2019) exploits the ability of BERT to predict the next sentence given a pair of sentences as input. Each input corresponds to three sequences of the form $S_1|S_2$, where | represents the string concatenation operation. S_1 is the text passage, instead, S_2 is the concatenation of the phrase "[ambiguous pronoun] is" and one of A's name, B's name or the word "neither". For instance, the passage "According to **her** mother, Tatyana Vladimovna, Demkina was a fast learner." corresponds to S_1. Instead, the sequence S_2 starts with the phrase "**her** is" and then continues with one of the options (A, B or "neither").

In this case, the head receives the pooled output from BERT, where the contextual embeddings has passed through a linear and Tanh layer. The probable output is obtain after a Logistic Regression layer, as the one proposed by Devlin et al. to solve the SWAG (Zellers et al., 2018) task.

4.2.1 Training and Hyper-parameters

The multiple choice model was trained using BERT large uncased with the first 12 layers frozen. In this case, the architecture is quite simple and perhaps this explains the worst performances. Indeed, it was trained for 3 epochs with a learning rate of 8e-6.

4.3 Mention Score

The following model is similar to the baseline for the initialization part, however, features are represented in a more complex way. In this case, a span embedding u is computed as the concatenation of the embeddings of the span's start u_s, span's end u_e and the average of all sub-tokens that form the span $\bar{u}$.

$$u = [u_s; u_e; \bar{u}]$$

Then, the span embeddings of A u_A and B u_B are concatenated and passed to a FFNN that computes the entities representation.

$$v_{ent} = FFNN_{ent}([u_A; u_B])$$

The single pronoun embedding u_P also goes through a FFNN and its output is finally concatenated to the entities representation.

$$v_P = FFNN_P(u_P)$$

$$v = [v_P; v_{ent}]$$

The tensor v is fed to a FFNN that terminates with a classification layer which outputs the probabilities over the classes A, B and Neither.

The name of this model is inspired by the score mention functions used by Lee et al. to solve the

end-to-end Coreference resolution task. In fact, the final tensor representation v is computed similarly to the span antecedent score.

4.3.1 Training and Hyper-parameters

The FFNN is composed by a first linear layer, that receives the mentions embeddings, followed by a layer normalization, the GELU activation function and terminates with a dropout layer and a final linear layer. The use of both normalization and dropout mitigate the risk of overfitting from the first epoch. It is also applied a step scheduler that decades the learning rate by 0.8 after the first epoch. Such practices stabilize the model and allow to reach the best performances (Table 1).

5 Extra

I also tried to implement the second part of the coreference resolution pipeline, so entity identification and resolution. In this case, the possible candidate mentions are not provided, therefore the model must first identify them and then select the entity that is coreferenced by the ambiguous pronoun. I attempted to build a generalization of the Mention Score architecture, where entities are recognized by a Named Entity Recognition (NER) model. The problem was formulated as a token classification task, so that the classification head has to output for each entity if it is the one coreferent by the pronoun or not. However, it can happen that different entities are identified as mentions correlated to the ambiguous pronoun, thus the one with highest probability score is selected.

5.1 Process Description

Since the ambiguous pronouns (Table 4) only refer to people, I use the Stanza library (Qi et al., 2020) for person entity recognition. Once entities were detected, I retrieved their offsets, in the tokenized text, by adding mention tags. Then, following an approach similar to the Mention Score one, the model builds entities representation by concatenating all entity span embeddings.

I suppose that the formulation of the problem could have been done following other approaches, since it only reaches 63% of accuracy.

6 Results and Discussion

Table 1 shows the validation accuracy scores of all models. The Multiple Choice approach is surpassed by the Baseline Base model even if it uses BERT large as pre-trained model. I did not want to try more hyper-parameter tuning on it, as it is quite slow to train, not only for the presence of the large model, but for the fact that it includes three copy of the same sentence, i.e., the training set is three times bigger. Instead, the baseline models perform pretty well, specially the cased version. Nevertheless, they required more epochs to be trained and they easily overfit when adding more layers to the head. Also the Mention Score model has problems with overfitting, so it is trained only for two epochs; however it achieves better results both in terms of accuracy and convergence time. In fact, the Mention Score model is about three times faster that the baseline, if we consider the BERT base models.

For what concerns the input tokenization, only the baseline models perform better by applying truncation. Instead, keeping the mention tags did not seem beneficial for any model, although tags could help the model's sensitivity to noise in the input text (Attree, 2019).

References

Sandeep Attree. 2019. Gendered ambiguous pronouns shared task: Boosting model confidence by evidence pooling. *CoRR*, abs/1906.00839.

Rakesh Chada. 2019. Gendered pronoun resolution using BERT and an extractive question answering formulation. *CoRR*, abs/1906.03695.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. *CoRR*, abs/1707.07045.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *CoRR*, abs/2003.07082.

Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. 2018. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *CoRR*, abs/1810.05201.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. *CoRR*, abs/1808.05326.

Model	ACC	Epochs	Time
Multiple Choice (Large)	82.2	3	11m 58s
Baseline Base Uncased	83.0	5	16m 16s
Baseline Base Cased	86.3	6	15m 49s
Baseline Large Uncased	86.8	3	8m 2s
Mention Score Base	87.0	2	5m 31s
Mention Score Large	**87.2**	2	5m 44s

Table 1: Accuracy scores on validation set. The Large models were trained on a Colab Tesla T4 GPU; instead the other model were trained using a GeForce RTX 2060.

Hyper-parameter	Value
Batch size	4
Head hidden size	512
Linear hidden size	256
Dropout	0.1
StepLR Scheduler	step size=1, $\gamma = 0.8$
Learning Rate	8e-6
Optimizer	Adam
Activation Function	GELU
Loss Function	Cross Entropy

Table 2: Mention Score hyper-parameters.

Hyper-parameter	Value
Batch size	4
Head hidden size	512
Dropout	0.1
StepLR Scheduler	step size=3, $\gamma = 0.5$
Learning Rate	5e-6
Optimizer	Adam
Activation Function	LeakyReLU
Loss Function	Cross Entropy

Table 3: Baseline Base Cased hyper-parameters.

Pronoun	Train	Dev
his	904 (30.1%)	108 (23.9%)
her	773 (25.8%)	140 (30.8%)
he	610 (20.4%)	93 (20.5%)
she	555 (18.5%)	87 (19.1%)
him	157 (5.2%)	26 (5.7%)

Table 4: Ambiguous Pronoun Distribution.

According to <P>her mother, <A>Tatyana Vladimovna</A>, <B>Demkina</B> was a fast learner.

Figure 1: Mention tags example.

According to her mother, Tatyana Vladimovna, Demkina **her**

The pronoun is appended after the truncation.

Figure 2: Truncation example.

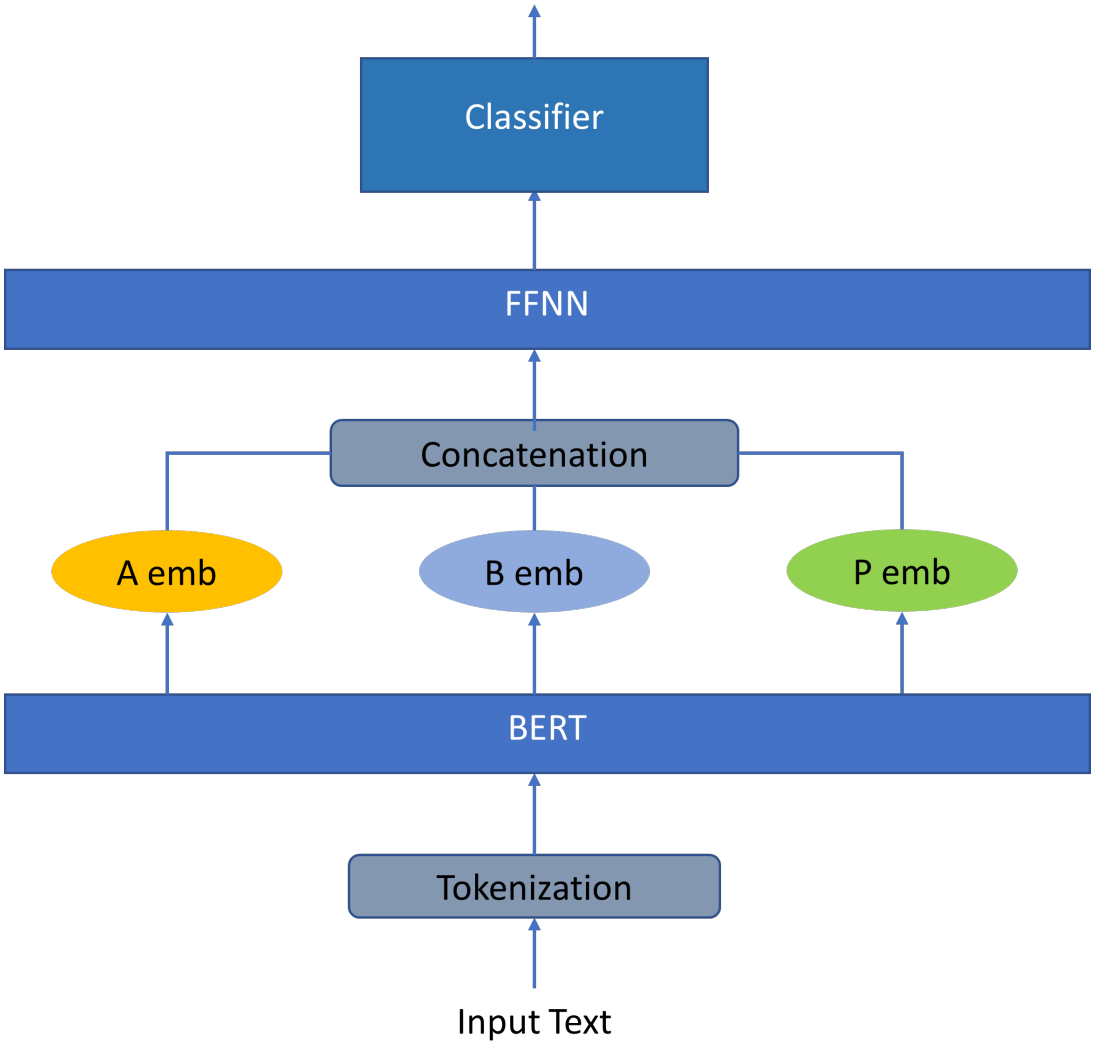

Figure 3: Baseline Architecture.

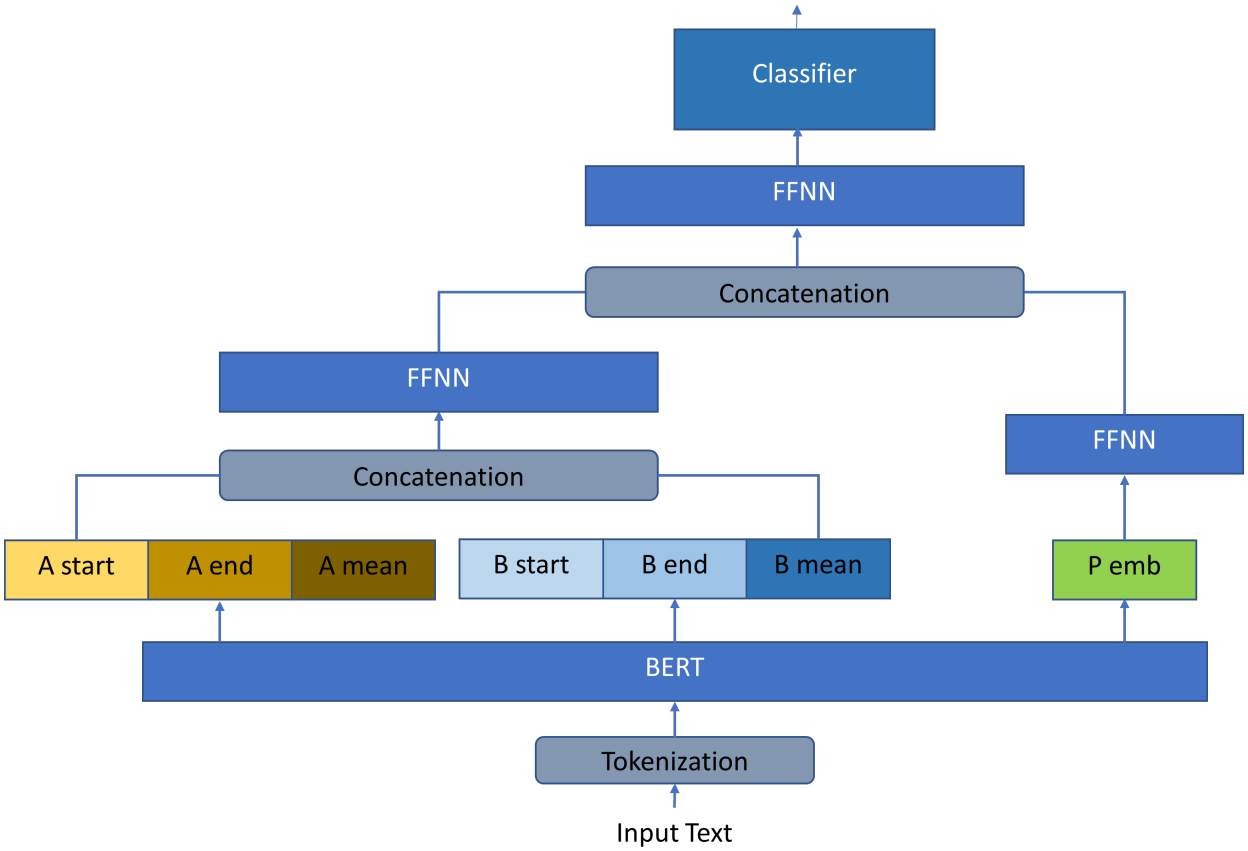

Figure 4: Mention Score Architecture.