INF 554 : MACHINE AND DEEP LEARNING

Extractive Summarization with Discourse Graphs

8 december 2023

Antoine Martinez, Auguste Crabeil, Julien Gadonneix

ÉCOLE POLYTECHNIQUE
IP PARIS

INTRODUCTION

The main objective of this challenge is to solve an important classification problem inherent in dialogue transcripts: to identify and discern the importance of individual utterances within a conversation. The fundamental task is to build an extractive summarisation system capable of performing binary classification, assigning labels of 1 or 0 to each utterance in order to determine its importance in the dialogue. The aggregation of the important statements identified forms an extractive summary, elucidating the critical elements of the conversation. This report presents the methodologies, techniques and knowledge gained from the rigorous exploration and application of machine learning models to achieve this goal. It examines the difficulties encountered and the strategies employed to mitigate them throughout the project.

1 FEATURE SELECTION

1.1 Embedding

Initially, our approach involved the utilization of pre-trained embedding as a foundational step to initiate our experimentation and model development. We commenced by employing BERT, which was readily available within the provided algorithm. This initial phase yielded promising results, achieving an F1-score of 0.59 solely with textual inputs, signifying a solid start to our endeavor. However, when we wanted to incorporate additional elements such as speaker information and graphical representations, we encountered inefficiencies, leading us to reassess our strategy. Despite exploring alternative pre-trained embedding, including models like RoBERTa, our efforts did not yield improved results. Subsequently, we applied the text vectorization process that was introduced during our lab sessions. Our choice leaned towards this method due to its perceived accuracy. The distinction arose from the fact that while other embedding models were pre-trained, this particular approach involved training and adaptation specific to our data set. As a result, it was presumed to be better aligned with the nuances of our data, potentially offering improved performance. Additionally, our preference for this method stemmed from a deeper understanding of how the text was embedded, allowing us to tailor our algorithms more effectively to suit our needs.

1.2 Graphs

Afterwards, we moved on to incorporate the graph component into our project. This graph had distinct characteristics — it was structured like a tree with labeled edges. Initially, we attempted to integrate it by using an adjacency matrix. However, our approach expanded beyond this basic matrix. We explored different ways to make use of the graph's structure. We tried incorporating the edge features into our models by utilizing edge label that is a categorical variable. Firstly, we experimented with adding this categorical feature as weights of the edges in the adjacency matrix. The results were slightly better but this embedding of the categorical variable is obviously inefficient. A second idea that we implemented is to extract 16 different adjacency matrices (because there are 16 categories), yo independently apply graph convolution networks to the nodes with these matrices and finally to merge the various results. The results were good, but did not exceed the ones we had at that time. Eventually, by searching through the literature, we found a possible way to consider the edge types in a graph neural network. The paper [2] describes Relational Graph Convolution Network.

1.3 Final features

In our algorithm, the features employed encompassed various aspects crucial for comprehensive analysis. These included the textual contentand additional contextual information such as 'before-text' and 'after-text,' providing insight into the preceding and subsequent sentences. Furthermore, we incorporated descriptions of the current and subsequent text , enriching the feature set with contextual details surrounding the dialogue segments. Another inclusion was the *speaker* attribute, a pivotal element in dialogue understanding. To effectively utilize the speaker attribute, we implemented it as a one-hot encoding scheme, like for the *description* feature from the edges. This approach allowed us to represent each unique speaker as a distinct categorical feature, enabling our model to comprehend and leverage speaker information effectively during classification tasks. Integrating these diverse features enabled a more comprehensive representation of the dialogues, significantly contributing to the success of our best-performing algorithm.

2
MODEL CHOICE, TUNING AND COMPARISON

2.1 Deep Learning model

After initially trying out basics Machine Learning models at the beginning such as SVM, random forest and XG-boost, we decided to move on Deep Learning models. We tried several different types of models, first it was pretty simple ones, but we added many steps and it became more and more complex. We tried several types of Deep learning models such as Linear, LSTM, GRU that we discovered in the literature [1], convolution model and GNN,(we explain in the additional part each type of model). For our best models, we mixed these models with optimized weights for each of them in order to get the best F1-score, and we achieve having a score of 0.614 with it.

2.2 Threshold optimization

At the beginning, we had a classic threshold equal to 0.5 for the binary classification from the regression. But we understood that as we aimed optimizing the F1-score which is different than accuracy, it might be smarter to have more 1 predictions than in accuracy optimization. Moreover, the input data is unbalanced because most of the utterances are not important and are labeled 0. This fact makes model to predict 0 with greater ease. Therefore, we decreased the threshold and tried different values to have the best F1-score. At the end we used a threshold more or less equal to 0.3, and this has improved a lot our score.

2.3 Avoid Overfitting

Overfitting poses a significant challenge that can negatively impact our model's performance. This occurs when the model becomes excessively tuned to the training data, resulting in poor predictions when applied to new test data. To tackle this problem, we explored various strategies. Initially, we incorporated dropout and weight decay into our model, which showed effectiveness. However, as the number of training epochs increased, the F1-score started declining due to overfitting. To address this, we designed an algorithm to retain the best-performing model throughout training. This algorithm continuously monitored the model's F1-score, preserving the iteration that achieved the highest performance before the onset of overfitting due to extensive epochs. This approach helped in maintaining the model's generalization ability and prevented it from becoming overly tailored to the training data.

CONCLUSION

This project provided us with the initial opportunity to independently tackle a real-world machine learning problem. It immediately exposed us to the challenges inherent in such tasks, ranging from data preprocessing to parameter optimization. This experience served as a valuable learning opportunity, allowing us to glean insights and knowledge firsthand.

REFERENCES

[1] Junyoung Chung et al. “Empirical evaluation of gated recurrent neural networks on sequence modeling”. In: *arXiv preprint arXiv:1412.3555* (2014).

[2] Michael Schlichtkrull et al. “Modeling relational data with graph convolutional networks”. In: *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*. Springer. 2018, pp. 593–607.

INFORMATION

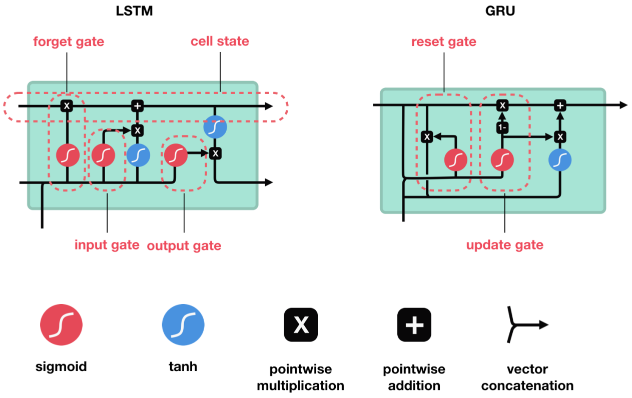

Figure 1: Graphic Explanation of different gate

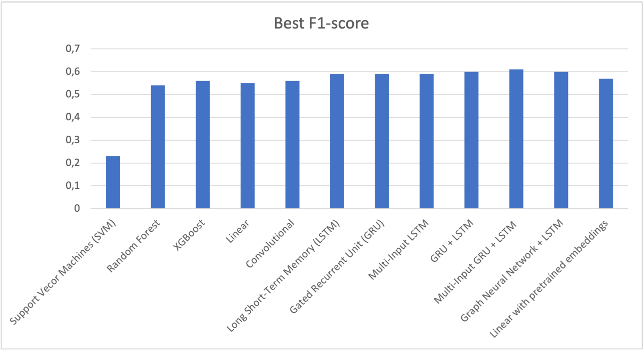

Figure 2: F1-score of our models