Data Challenges

Extractive Summarization with Discourse Graphs

INF554

Machine and Deep Learning

Team: BattleField4

Ziyu WU **ziyu.wu@polytechnique.edu**

Heyuan LIU **heyuan.liu@polytechnique.edu**

1. Feature selection

There are 2 kinds of features we get from the data, original Diagram and Text.

1.1 Diagram Features

The indexes of the source node, the target node, and the relationship between them are listed in the data file. The features extracted from them are the degree of the node, the degree centrality of the node, and the relations. We also experimented with the depth of the graph as a training feature but found, after testing, that it did not provide any improvement to the model, and in some cases, it even had a negative impact.

Degree of the node and the Degree Centrality of the node

The degree of a node in a graph is defined as the number of edges that are connected to the node. The degree of a node reflects its level of connectivity within the graph. Nodes with higher degrees are more connected as they have more edges incident to them. In a social network, for example, a node with a high degree might represent a person with many social connections.

Relations

To be more specific for the relations, since the relations are the connections to the nodes, so the number of relations should be less than the number of nodes, so, we assign "null" to the relations of the leaf nodes. Meanwhile, the data type of feature "relation" is string, so we created a mapping for the relations. Map the relation strings to the integers. And we use the training data set to find out all the types of relations and their frequency, and we found that the higher frequency, the lower importance for the sentence. So we used the frequency to sort the relation mapping from 1 to 16 and the relations for the leaf nodes were assigned to 0. The details of the relation types can be found at [Appendix I]

1.2 Text Features

We combine the name of the speaker and the sentence together and use the encode( ) function from the SentenceTransformer to encode the text feature, which transforms the combined string to vector representations.

2. Model selection

The comparisons for all the models can be seen at [Table-1].

2.1 Tree Model

We attempted to use decision trees, random forests, and XGBoost in the text section, but the results were not satisfactory. However, in handling graphical data, tree models demonstrated significant advantages. we employ a decision tree model to handle diagram data, as decision trees perform well in capturing non-linear relationships between structured data and features.

2.2 LSTM

The choice of LSTM is motivated by the nature of the input being textual, where there exists a temporal relationship between the context. This temporal dependency makes a sequential model, such as LSTM, more suitable than a Convolutional Neural Network (CNN). Unlike CNNs, LSTMs are designed to capture and remember long-term dependencies in sequential data, making them

well-suited for tasks involving language and contexts where the order of information is crucial. The temporal aspect of LSTMs allows them to effectively model the sequential nature of text, making them a preferred choice when dealing with datasets where the temporal ordering of information is essential for accurate analysis and understanding.

2.3 Two Method

Decision Tree for diagram and LSTM for text, and then use a linear model to combine two models together. Firstly, we employ a decision tree model to handle diagram data, as decision trees perform well in capturing non-linear relationships between structured data and features. This is particularly effective for extracting and analyzing information from diagrams. Secondly, we utilize a Long Short-Term Memory network (LSTM) to process text data. LSTM is a deep learning model suitable for sequential data, excelling at capturing long-term dependencies and contextual information within textual content. This makes it an ideal choice for handling natural language text. Finally, to leverage the strengths of both models comprehensively, we introduce a linear model to merge them. This linear model aids in integrating the outputs of the decision tree and LSTM models, creating a more comprehensive predictive model that considers both diagram and text information. Through this approach, we can analyze and understand complex data that includes both graphical and textual information more comprehensively. After optimization, the score reaches 0.53.

2.4 GNN

Utilize the Graph Neural Network (GNN) model for joint training on both textual and diagrammatic data. Consider the textual information as the node features, while treating the diagram as the source of edge features.

We have tried three models: GCN [1], GAT [2], and GraphSAGE [3]. And we chose the GraphSAGE, which performs best on this particular case.

- ### GCN

Description: GCN is a deep learning model specifically designed for processing graph data. It effectively captures relationships and topological structures among nodes in a graph. Introducing graph convolutional operations, akin to the convolutional operations in traditional Convolutional Neural Networks (CNNs), GCN enables nodes to propagate and update information through their neighboring nodes. In GCN, adjacent nodes share the same weight matrix, aiding in reducing parameter count, enhancing model generalization, and mitigating the risk of overfitting.

Results: Even worse than “TwoMethods”

Analysis: When stacked in multiple layers, GCN may suffer from the issue of over-smoothing, resulting in a loss of fine-grained details. This is a common challenge associated with GCN. Subsequent models such as GraphSAGE and GAT have been designed to overcome this issue.

- ### GAT

Description: GAT introduces an attention mechanism that allows the model to assign different weights to each neighboring node, providing greater flexibility in capturing relationships between nodes. This enables the model to focus on the importance of different parts of the graph, rather than simply averaging or aggregating the information of neighboring nodes with uniform weights. In other words, the attention mechanism permits the model to concentrate on those neighboring nodes

that are more crucial for the current node, reducing dependence on the entire graph. GAT can leverage global information for each node, as the attention mechanism enables the aggregation of information from the entire graph. This facilitates a better capture of the graph's structure and long-range dependencies between nodes.

Results: It is observed that there is not a significant improvement in performance.

Analysis: GAT tends to emphasize global information, employing the attention mechanism, but the task of evaluating dialogue data does not necessarily demand a high reliance on global information.

- **GraphSAGE (Selected)**

Description: GraphSAGE primarily samples neighboring nodes and aggregates their information. It focuses on capturing local information without introducing complex attention weights, unlike GAT. GraphSAGE is specifically designed to concentrate on capturing local neighborhood information, making it more suitable for tasks that require attention to the surrounding structure of nodes. In the case of dialogue text, which heavily relies on context but doesn't necessarily depend on global information, GraphSAGE is expected to outperform GAT in handling such tasks. The initial performance surpassed that of previous models, and considering the beneficial characteristics of the attention mechanism for this specific task, it was evident that the effectiveness of GraphSAGE with attention mechanism was explicitly validated in the experiments and data presented in [4]. Inspired by these findings, we introduced the attention mechanism from GAT (Graph Attention Network) into our model, to be more specific, assigning different attention weights through a linear layer. Resulting in a notable improvement in its performance. The overview of our GraphSAGE with attention mechanisms is depicted in [Figure – 1].

Results: the score reaches 0.55 for the original GraphSAGE. After an optimization for attention mechanism, the score reaches 0.57.

The comparisons for all the models can be seen at [Table-1].

3. Optimization Method

Linear Regression:

For the “TwoMethods” solution, we get two sets predictions from the diagram and text, we considered them as 2 variables for linear regression, and get the model for the final prediction.

Grid Search:

We used the Grid Search to find out the proper parameters for LSTM.

For LSTM, we use the Grid Search to find out the proper parameters, including number of Neurons, learning rate, dropout rate, epochs, etc.

Preventing Overfitting

We observed a trend where, with an increase in epochs, the training loss decreased while the validation set score also diminished. In response, we implemented a proactive strategy: at the conclusion of each epoch, we assessed the model using the validation set originally partitioned from the dataset. We computed a score for each evaluation, keeping a record of the model with the highest score. This approach deviates from the conventional practice of relying solely on the fully trained model and aims to counteract overfitting by selecting the model that demonstrates optimal performance on the validation set during the training process.

Appendix I Relation details

mapping: {'nan': 0, 'Acknowledgement': 1, 'Continuation': 2, 'Elaboration': 3, 'Contrast': 4, 'Comment': 5, 'Question-answer_pair': 6, 'Result': 7, 'Clarification_question': 8, 'Explanation': 9, 'Conditional': 10, 'Alternation': 11, 'Q-Elab': 12, 'Narration': 13, 'Background': 14, 'Parallel': 15, 'Correction': 16}

Appendix II Reference

[1] Kipf, T. N., & Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks. ArXiv.Org. https://doi.org/10.48550/arxiv.1609.02907

[2] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. ArXiv.Org. https://arxiv.org/abs/1710.10903

[3] Hamilton, W. L., Ying, R., & Leskovec, J. (2018). Inductive Representation Learning on Large Graphs. ArXiv.Org. https://doi.org/10.48550/arxiv.1706.02216

[4] Y. Ding, X. Zhao, Z. Zhang, W. Cai and N. Yang, "Graph Sample and Aggregate-Attention Network for Hyperspectral Image Classification," in IEEE Geoscience and Remote Sensing Letters, vol. 19, pp. 1-5, 2022, Art no. 5504205, doi: 10.1109/LGRS.2021.3062944.

Appendix III Figures and Tables

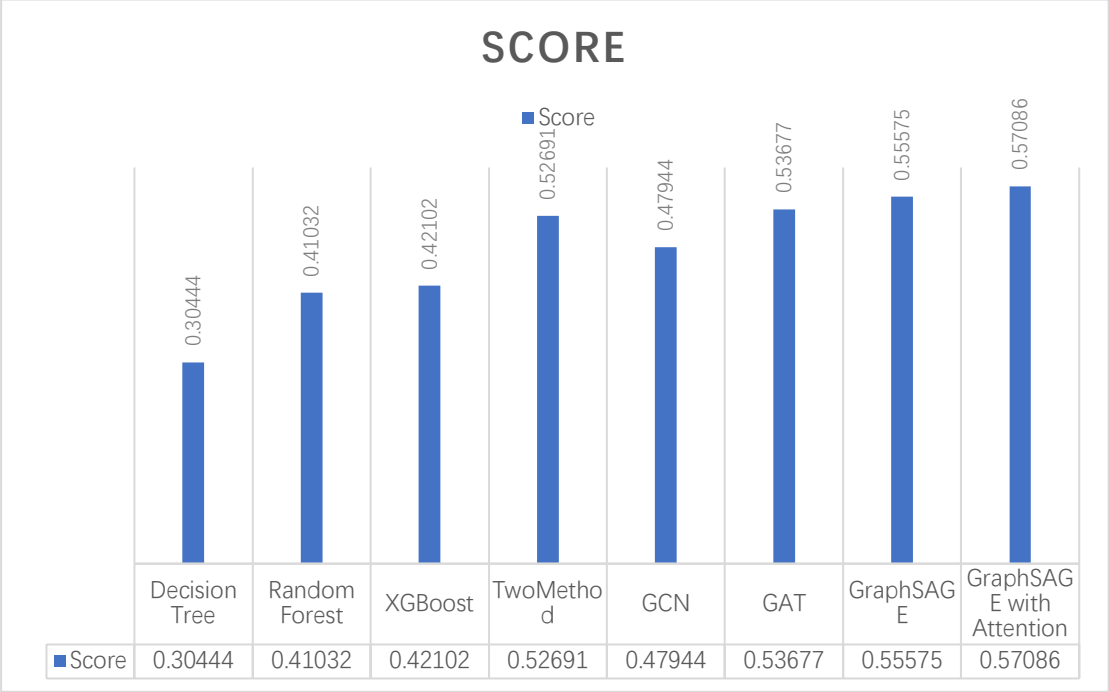

Table – 1 Comparisons among all models

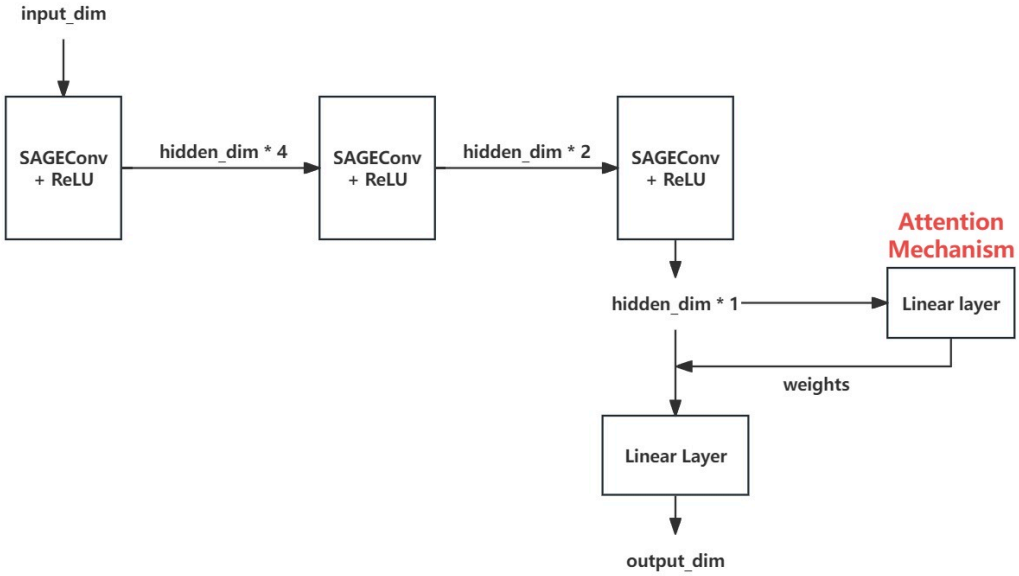

Figure – 1 Overview of GraphSAGE with attention Mechanism