The goal of multi-document summarization aims to produce fluent and coherent summaries covering salient information in the documents. The task is generally composed of producing a summary of a cluster of documents that are all describing the same event or topic (for example, multiple news articles that are written about the same story). Many previous summarization systems employ an extractive approach by identifying and concatenating the most salient text units (often whole sentences) in the document. Current state-of-the-art summarizers (Gillick et al., 2009; Haghighi and Vanderwende, 2009; Christensen et al., 2013; Hong and Nenkova, 2015) use graph-based, greedy, ILP, or oracle methods to generate useful summarizations. Our work proposes combining a discourse graph (which we call PDG) with a neural network to extract salient sentences from multiple documents.

Introduction

Terms and Methods

Datasets: In multi-document summarization, the main datasets that are used come from a series of Document Understanding Conferences (DUC) from 2001-2004. These datasets are made up of clusters of articles written on the same topic (typically news articles). These clusters also come with a gold standard reference summary to compare with.

Evaluation Method: In 2004, the ROUGE metric (Recall-Oriented Understanding for Gisting Evaluation) was introduced to evaluate the quality of the summaries. The metric roughly calculates the co-occurrence of n-grams in the reference and candidate summaries.

ILP Summarization: Tries to optimize covering all possible bigrams weighted by their frequencies in the document, which an ILP solver can solve pretty well.

Graph-based Summarization: Creates a graph of sentence relations across the documents, and weights them by macro-level sentence features.

Results

In this paper, we present a novel multi-document summarization system that exploits the representational power of neural networks and graph representations of sentence relationships. On top of a simple GRU model as an RNN-based regression baseline, we build a Graph Convolutional Network (GCN) architecture applied on a Personalized Discourse Graph. Our model, unlike traditional RNN models, utilizes graph representations of sentence relations and demonstrates improved salience prediction and summarization, achieving competitive performance with current state-of-the-art systems.

Acknowledgement

We would like to thank the LILY Lab, as well as all coauthors that were instrumental to this work: Rui Zhang, Ayush Parek, and Professor Dragomir Radev.
Proposed Improvements to ADG (Cont’d)

Specifically, the ADG lacks much diversity in its assigned edge weights. Because weights are discretely incremented, they are multiples of 0.5; many edge weights are 1.0. While the presence of an edge provides a remarkable amount of underlying knowledge on the discourse relationships, edge weights can further include information about the strength — and, similarly, salience — of these relationships. We hope to improve the edge weights by making them more diverse, while infusing more information in the weights themselves. In doing so, we contribute our Personalized Discourse Graph (PDG).

Method to Construct PDG

To advance the ADG’s performance in providing predictions for sentence salience, we apply a multiplicative effect to the ADG’s edge weights via sentence personalization. A baseline sentence personalization score \( s(v) \) is calculated for every sentence \( v \) to account for surface features in each sentence. These features include sentence length, sentence position in document, and number of proper nouns embedded in the sentence, to which we apply linear regression, as per Christensen et al. (2013). Each edge weight in the original ADG is then transformed by this sentence personalization score and normalized over the total outgoing scores. That is, for directed edge \((u, v)\) \( E \), the weight is

\[
\frac{w_{PDG}(u, v)}{w_{ADG}(u, v) \cdot s(v)}
\]

The inclusion of the personalization score of the edge's destination sentence allows the PDG to account for macro-level features in each sentence, improving salience measurements. Because we hope to maintain consistency between graph representations, two modifications are made to the discourse graphs. First, the directed edges of both the ADG and PDG are made undirected by averaging the edge weights in both directions. Second, edge weights are rescaled to a maximum edge weight of 1 prior to being fed to the GCN.

Results & Discussion

Table 1 summarizes the following basic statistics: the number of nodes (i.e., sentences), the number of edges, average edge weight, and average node degree per graph. We include the correlation between node degree and salience, as well. As seen from the table, PDG and ADG have approximately the same number of edges. This is expected since the PDG is built by transforming the edge weights in ADG. The Cosine Similarity Graph has slightly fewer edges, simply due to the implemented threshold. Moreover, note that the ADG has significantly higher average edge weight and node degree as compared to the PDG. These values reflect the discrete nature of the ADG’s edge assignment—further evidence of this can be seen in Figure 1. Because the ADG’s raw edge weight assignment is done by increments of 0.5, the average node degree tends to be significantly large. This motivated the construction of the PDG, which corrects for this by coercing the average edge weight and node degree to be more diverse and, consequently, smaller (after rescaling). The process of including sentence personalization scores in edge weight assignments of the PDG leads to a select number of edges gaining markedly large distinction. This aids the GCN in identifying the most important edge connections along with the affiliated sentences.

Conclusion

We present our Personalized Discourse Graph (PDG), which expands on prior discourse graphs by accounting for macro-level sentence features. Through the use of personalized sentence scores extrapolated via regression, we assign weights by a rough approximation of the graphs stationary distribution. Applying the PDG to a GCN architecture produces promising multi-doc summaries.

Acknowledgement

We would like to thank the LILY Lab, as well as all coauthors that were instrumental to this work: Rui Zhang, Ayush Parek, and Professor Dragomir Radev.
Graph-based Neural Multi-Document Summarization

Michihiro Yasunaga, Kshitijh Meelu, and Krishnan Srinivasan

LILY Lab, Department of Computer Science, Yale University, New Haven, CT

Introduction

RNN sequence models have been successful in single-doc summarization recently. However, in the multi-doc setting, extending these models by simply concatenating the documents in a cluster does not perform well. Motivated by this, we incorporate graph representation of sentence relations into a simple RNN model via Graph Convolutional Networks, and achieve improved performance.

Proposed Summarization Model

Given the document cluster, our method extracts sentences as a summary in two steps: sentence salience estimation and sentence selection.

1. Salience Estimation (Figure 1)

Graph representation of sentences: we first build a sentence relation graph of a cluster, as explained in the previous subsection. GRUv2, we obtain initial sentence embeddings by applying RNN with GRU on the words in each sentence.

2. Sentence Extraction

The goal of GCN is to learn a function f(H, A) that takes as input the adjacency matrix of the graph, A, and the initial node features H, and outputs high-level hidden features H' for each node that update the initial node features by incorporating the graph structure. Specifically, each layer of GCN takes the following propagation rule:

$$H^{(l+1)} = \sigma \left(W^{(l)} \cdot \text{ReLU}(D^{-\frac{1}{2}} \cdot A \cdot D^{-\frac{1}{2}} \cdot H^{(l)}) + W^{(l-1)} \cdot H^{(l-1)}\right)$$

where H is the input node features, D is the normalization factor for the adjacency matrix A, W is the parameter to learn in this layer, and H(0) is the output node features. σ is an activation function such as ReLU.

Through multiple layers of GCN propagation, we obtain the final sentences embeddings that encapsulate the information of the sentence relation graph.

Cluster embedding: we apply second level RNNs on each document to get document embeddings. The average pooling is our cluster embedding.

Salience estimation: for each sentence s in a cluster, we calculate the salience estimate of s as follows:

$$s(s) = \sum_{v(s)} \text{sal} \left(W \cdot H + W' \cdot H' \right)$$

where C is the cluster embedding, s is the final sentence embedding, W, W' are the parameters to learn in this layer.

Training: the model is trained end-to-end to minimize the following cross-entropy loss between the salience prediction and the correct score of each sentence:

$$L = -\sum_{i} (s(s_i) \log(s(s_i)))$$

where s(s) is the actual ROUGE score and sal(s) is the estimate. They are both normalized across the cluster via softmax.

2. Sentence Extraction

Given the salience score estimation, we apply a simple greedy procedure to select sentences. We sort sentences in descending order of the salience scores. Then, we keep selecting one sentence from the top of the list and append to the summary if the sentence is of reasonable length (5-55 words) and is not redundant, until we reach the length limit (700 words). The sentence is redundant if the f1 of cosine similarity between 440 the sentence and the current summary is above 0.5.

Experiments

1. Data set and Evaluation

We use the benchmark, data sets from the Document Understanding Conferences (DUC) containing clusters of English news articles and human reference summaries (Table 1). Our model is trained on DUC 2001 and 2002, validated on 2003, and tested on 2004. For evaluation, we use the ROUGE-1.2 metric.

2. Experimental Setup

We conduct four experiments on our model: without any graph, with Cosine Similarity Graph, with ADG and with PDG. We apply GCNs with the graphs in the final step of sentence encoding. For the experiment without any graph, we take the paragraph as the input node and simply use the GRU sentence and cluster encoders.

3. Preprocessing

For each document cluster, we tokenize all the sentences into words and generate a graph representation of their relations by the three methods mentioned above. Additionally, we prepare the correct salience scores of each sentence by measuring its ROUGE score with the human-written reference summary.

4. Implementation Detail

- 300-dimensional pre-trained word2vec embeddings for GRU
- All the hidden states are 300 dimensions
- 3 GCN layers

Acknowledgement

We would like to thank the LILY Lab, as well as all coauthors that were instrumental to this work: Rui Zhang, Ayush Parek, and Professor Dragomir Radev.

Figures

Figure 1. Illustration of our architecture for sentence salience estimation.

Results (Table 2)

1. Comparison among our models

First we take our simple GRU model as the baseline of the RNN-based regression approach. As seen from the table, the addition of sentence relation graphs, especially PDG, on top of the GRU clearly boosts the performance. This improvement indicates that the combination of graphs and GCNs processes sentence relations across documents better than the vanilla RNN sequence models.

2. Comparison with other systems

We also compare our result with other baseline multi-document summarizers and the state-of-the-art systems related to our regression method. Our GCN system significantly outperforms the commonly used baselines and traditional graph approaches such as Centroid, LexRank, and G-Flow. This indicates the advantage of the representation power of neural networks used in our model. Our system also exceeds CLAIRSYA, the best peer system in DUC 2004, and Support Vector Regression (SVM), a widely used regression-based summarizer.

Discussion

Our graph-based models outperform the vanilla GRU model, and among our graph-based model, PDG performed the best. While the Cosine Similarity Graph encodes word-level connections between sentences, PDG specializes in representing the narrative and logical relations between sentences. To better understand the results and validate the effect of sentence relation graphs (especially of the PDG), we have conducted the following analysis.

Training Statistics (Table 3)

Table 3: Training statistics for the four different settings in Table 4 in our experiment.

Table 4: Results for the four different settings in Table 4.

Table 5: Comparison among our models and of discourse graphs applied to processing multi-documents.