Yale

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

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.

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-occurence of n-grams in the reference and candidate summaries.

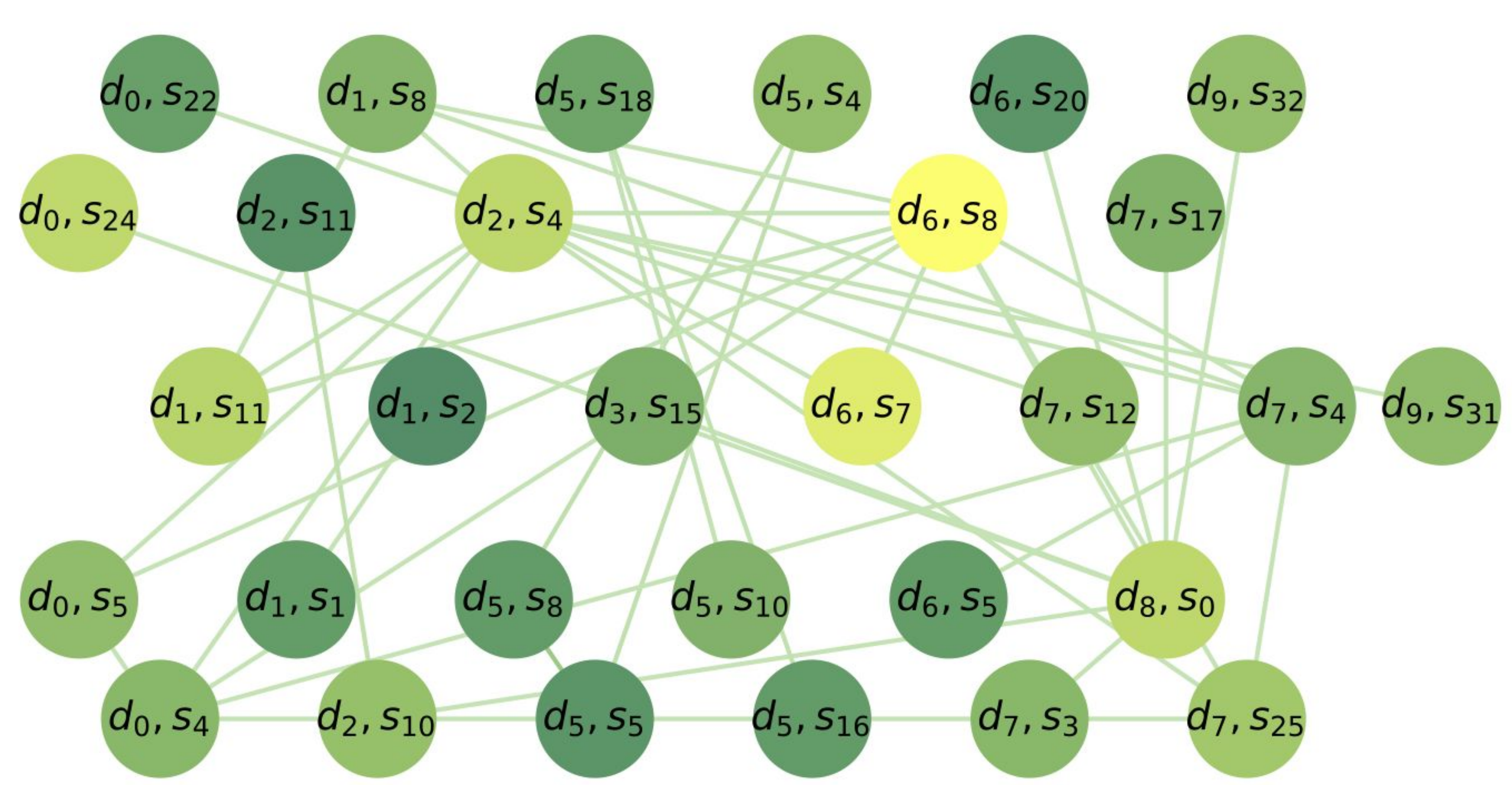
ILP Summarization: Tries to optimize covering all possible bigrams weighted by their frequencies in the document, which an ILP solver can solve pretty well. This is why ICSISumm does particularly well on ROUGE-2.

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

	R-1	R-2
SVR (Li et al., 2007)	36.18	9.34
CLASSY11 (Conroy et al., 2011)	37.22	9.20
CLASSY04 (Conroy et al., 2004)	37.62	8.96
GreedyKL (Haghighi and Vanderwende, 2009)	37.98	8.53
TsSum (Conroy et al., 2006)	35.88	8.15
G-Flow (Christensen et al., 2013)	35.30	8.27
FreqSum (Nenkova et al., 2006)	35.30	8.11
Centroid (Radev et al., 2004b)	36.41	7.97
Cont. LexRank (Erkan and Radev, 2004)	35.95	7.47
RegSum (Hong and Nenkova, 2014)	**38.57**	**9.75**
GRU	$36.64_{\pm 0.11}$	8.47
GRU+GCN: Cosine Similarity Graph	$37.33_{\pm 0.23}$	8.78
GRU+GCN: ADG from G-Flow	$37.41_{\pm 0.32}$	8.97
GRU+GCN: Personalized Discourse Graph	$\mathbf{38.23}_{\pm 0.22}$	**9.48**

Table 1. ROUGE Recall on DUC 2004.

Figure 1 Our Personalized Discourse Graph



Reference Summary (truncated): Malaysian Prime Minister Mahathir Mohamad ruled adroitly for 17 years until September 1998 when he suddenly reversed his economic policy and fired his popular deputy and heir apparent, Anwar Ibrahim. Anwar organized a political opposition, leading Mahathir to arrest him. (...) Anwar remained in custody as lawyers appealed. (...)

SentID (6,8): Anwar was ... after two weeks of nationwide rallies at which he called for government reform and Mahathir's resignation, he was arrested ....

SentID (6,7): The two had differed over economic policy and Anwar has said Mahathir feared he was a threat to his 17-year rule.

SentID (2,4): Mahathir and Anwar had differed over economic policy and Anwar says Mahathir feared him as an alternative leader.

SentID (0,22): Before his arrest, Anwar designated his wife, Azizah Ismail, as the leader of his new ``reform'' movement.

Figure 2. Sentences highlighted by salience

	DUC'01	DUC'02	DUC'03	DUC'04
# of Clusters	30	59	30	50
# of Documents	309	567	298	500
# of Sentences	24498	16090	7721	13270
Vocabulary Size	28188	22174	13248	18036
Summary Length	100 words	100 words	100 words	665 Bytes

Table 2. DUC Statistics

Results

As Michihiro mentioned, our greedy approach towards selecting the most salient sentences is how summaries are generated by the model. In Figure 2, we inspect the salient sentences as they are detected by our model. One of the relationships we found in our model was how salience of a sentence was positively correlated with the degree of the node, indicating that the structure of the PDG is useful to find good sentences to evaluate for summarization. Figure 1 visualizes the salient sentences (which are listed in Figure 2) that correspond to cluster d30011t in the DUC 2004 dataset. Finally, we note that the correlation strength for PDG is superior to the strength of both ADG and Cosine Similarity Graphs.

Of the various experiments we tried, one was to normalize sentence embeddings in order to improve the score by normalizing the input ranges to the neural network, which has shown to work for word embeddings. While we could not initially get it to work in time for the paper deadlines, this is a valid next step or consideration as we attempt to improve our current implementation.

Conclusion

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.

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

Broadly, graph-based multi-doc summarization models have traditionally employed surface level (Erkan and Radev, 2004; Mihalcea and Tarau, 2005; Wan and Yang, 2006) or deep level (Antiqueira, 2007; Antiqueira et al., 2009; Leite et al., 2010) approaches based on topological features and the number of nodes (Albert and Barabasi, 2002). Efforts have been made to improve decision making of these systems by using discourse relationships between sentences (Radev, 2000; Radev et al., 2001). In our work, we build on the approximate discourse graph (ADG) model (Christensen et al., 2013) and account for macro level features in sentences to improve sentence salience prediction.

Providing a departure from previous cosine similarity sentence connections, G-Flow (Christensen et al., 2013) identifies discourse relationships via an approximate discourse graph (ADG) to maintain coherence between parent and child summaries as well as within each summary. Events, such as deverbal noun references, event and entity continuations discourse markers, coreferences, among others, allow characterization of sentence relationships. Identifying and embedding these relationships within a directed graph structure allows the G-Flow system to measure the coherence of potential summaries by summing over the weights of all edges corresponding to sentence pairs in the summary.

Marking these as directed edges allows not only the creation of more coherent summaries but also the implementation of graph-related algorithms and networks. While the G-Flow system only makes use of the graph for coherence reasons, there remains potential for applying graph representations within other multi-document summarization systems as a form of discourse input.

Proposed Improvements to ADG

To implement a derivative of the ADG for direct salience prediction, there remains a need to encode macro-level sentence features in the discourse graph. While G-Flow's ADG provides many improvements from baseline graph representations, it suffers many disadvantages that diminish its ability to provide more accurate salience prediction when given to the neural network.

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).

	PDG	ADG	Cosine Similarity
Number of nodes	265	265	265
Number of edges	1023	1050	884
Average edge weight	0.075	0.295	0.359
Average node degree	0.171	5.136	2.260
ρ of degree and salience	0.136	0.113	0.093

Table 1. Characteristics of the three graph representations, averaged over the clusters (i.e. graphs) in DUC 2004. Note that max edge weight in all three representations is 1.0 due to rescaling for consistency. The degree of each node is calculated as the sum of edge weights.

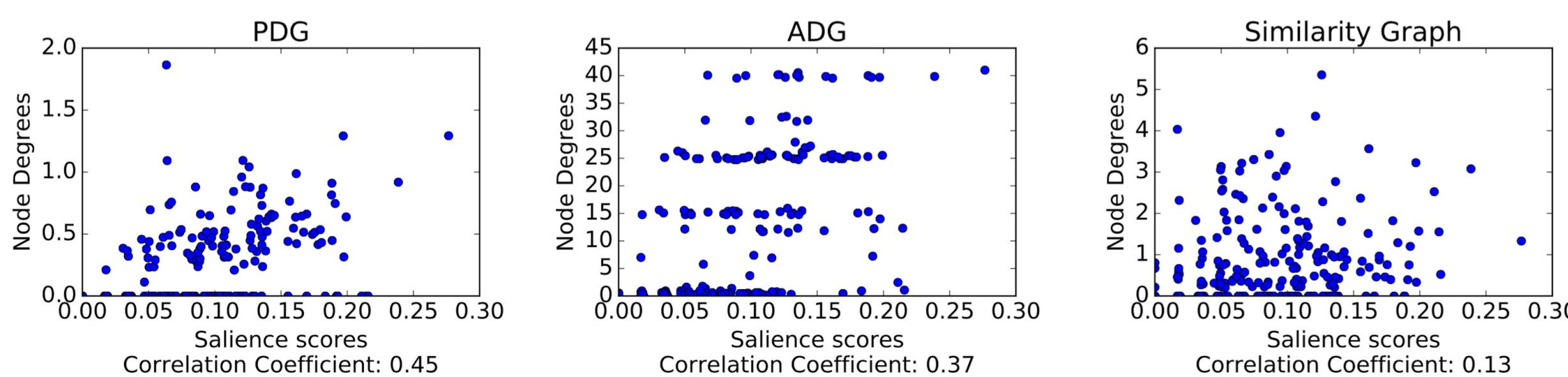


Figure 1. Visualization of the relationship between salience score and node degree for the three graph representation methods. Cluster d30011t from DUC 2004 is chosen as an example.

Method to Construct PDG

To advance the ADG's performance in providing predictors 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 $(u, v) \in E$, the weight is

$$w_{PDG}(u,v) = \frac{w_{ADG}(u,v)s(v)}{\sum_{u' \in V} w_{ADG}(u',v)s(u')}$$

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 edges 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

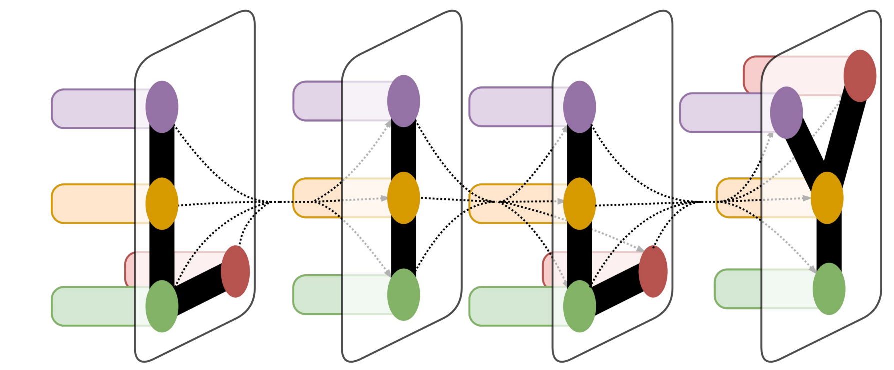
We would like to thank the LILY Lab, as well as all coauthors that were instrumental to this work: Rui Zhang, Ayush Parek, and Professor Dragomir Radev.

Yale

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 slide.

GRU^{sent}: we obtain initial sentence embeddings by applying RNN with GRU on the words in each sentence.

Graph Convolutional Networks (GCN):
We then apply GCN (Kipf and Welling, 2017) on the sentence relation graph, with the sentence embeddings as the input node features, $H^{(0)}$.
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:

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

where $H^{(l)}$ is the input node features, D is the normalization factor for the adjacency matrix A, $W^{(l)}$ is the parameter to learn in this layer, and $H^{(l+1)}$ 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_i in a cluster, we calculate the salience estimate of s_i as follows:

$$f(s_i) = \mathbf{v}^T \tanh(\mathbf{W}_1\mathbf{C} + \mathbf{W}_2\mathbf{s}_i)$$

where **C** is the cluster embedding, $\mathbf{s}_i$ is the final sentence embedding **v**, $\mathbf{W}_1$, $\mathbf{W}_2$ are the parameters to learn.

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:

$$\mathcal{L} = -\sum_{C}\sum_{s_i \in C} R(s_i)\log(\text{salience}(s_i))$$

where $R(s_i)$ is the actual ROUGE score and salience(s_i) is the estimate. Both are 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 (100 words). The sentence is redundant if the tf-idf cosine similarity between 460 the sentence and the current summary is above 0.5.

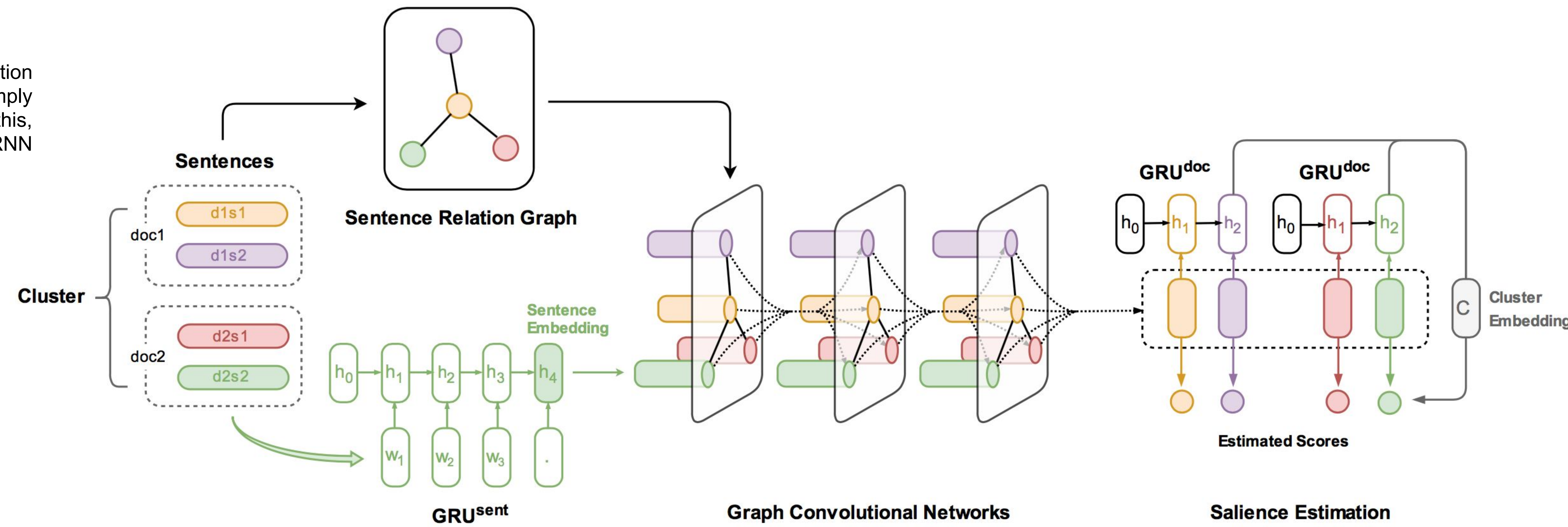


Figure 1. Illustration of our architecture for sentence salience estimation.

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 omit the GCN part and simply use the GRU sentence and cluster encoders.

3. Preprocessing

For each document cluster, we tokenize all the documents into sentences 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^{sent}
- All the hidden states are 300 dimensions
- 3 GCN layers

	DUC'01	DUC'02	DUC'03	DUC'04
# of Clusters	30	59	30	50
# of Documents	309	567	298	500
# of Sentences	24498	16090	7721	13270
Vocabulary Size	28188	22174	13248	18036
Summary Length	100 words	100 words	100 words	665 Bytes

Table 1: Statistics for DUC Multi-Document Summarization Data Sets.

	R-1	R-2
SVR (Li et al., 2007)	36.18	9.34
CLASSY11 (Conroy et al., 2011)	37.22	9.20
CLASSY04 (Conroy et al., 2004)	37.62	8.96
GreedyKL (Haghighi and Vanderwende, 2009)	37.98	8.53
TsSum (Conroy et al., 2006)	35.88	8.15
G-Flow (Christensen et al., 2013)	35.30	8.27
FreqSum (Nenkova et al., 2006)	35.30	8.11
Centroid (Radev et al., 2004b)	36.41	7.97
Cont. LexRank (Erkan and Radev, 2004)	35.95	7.47
RegSum (Hong and Nenkova, 2014)	**38.57**	**9.75**
GRU	$36.64_{\pm 0.11}$	8.47
GRU+GCN: Cosine Similarity Graph	$37.33_{\pm 0.23}$	8.78
GRU+GCN: ADG from G-Flow	$37.41_{\pm 0.32}$	8.97
GRU+GCN: Personalized Discourse Graph	$\mathbf{38.23}_{\pm \mathbf{0.22}}$	**9.48**

Table 2: ROUGE Recalls on DUC 2004. We show mean (and standard deviation for R-1) over 10 repeated trials for each of our experiments.

	PDG	ADG	Cosine Similarity	No Graph
Num of Iterations	200	280	310	250
Train Cost	4.286	5.460	5.458	5.310
Validation Cost	4.559	5.077	5.099	5.214

Table 3: Training statistics for the four experiments. The first row shows the number of iterations the model took to reach the best validation result before an early stop. The train cost and validation cost at that time step are shown in the second row and third row, respectively. All the values are the average over 10 repeated trials.

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.

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. The 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 CLASSY04, the best peer system in DUC 2004, and Support Vector Regression (SVR), a widely used regression-based summarizer.
We remain at a comparable level to RegSum, the state-of-the-art multi-document summarizer using regression. However, note that RegSum performs word level regression, while our model simply works on sentence level.

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).

We compare the training curves of the four different settings in Table 3: no graph, Cosine Similarity Graph, ADG, and PDG. Without a graph, the model converges faster and achieves lower training cost than the Cosine Similarity Graph and ADG. This is most likely due to the simplicity of the architecture, but it is also less generalizable, yielding a higher validation cost than the models with graphs.
For the three graph methods, PDG converges even faster than "No Graph" and achieves the lowest training cost and validation cost amongst all methods. This shows that the PDG has particularly strong representation power and generalizability.

Conclusion

Our neural multi-document summarization system exploits the graph representations of sentence relationships via GCN, and demonstrates improved summarization performance over a traditional RNN sequence model. Moreover, we have validated the efficacy of sentence relation graphs, particularly PDG, to aid the model to learn the salience of sentences. This work shows the promise of the GCN models and of discourse graphs applied to processing multi-documents.