Graph-based Neural Multi-Document Summarization

Michihiro Yasunaga[1] **Rui Zhang**[1] **Kshitijh Meelu**[1]
Ayush Pareek[2] **Krishnan Srinivasan**[1] **Dragomir Radev**[1]
[1]Department of Computer Science, Yale University
[2]The LNM Institute of Information Technology
{michihiro.yasunaga,r.zhang,kshitijh.meelu}@yale.edu
{ayush.original}@gmail.com
{krishnan.srinivasan,dragomir.radev}@yale.edu

Abstract

We propose a neural multi-document summarization (MDS) system that incorporates sentence relation graphs. We employ a Graph Convolutional Network (GCN) on the relation graphs, with sentence embeddings obtained from Recurrent Neural Networks as input node features. Through multiple layer-wise propagation, the GCN generates high-level hidden sentence features for salience estimation. We then use a greedy heuristic to extract salient sentences while avoiding redundancy. In our experiments on DUC 2004, we consider three types of sentence relation graphs and demonstrate the advantage of combining sentence relations in graphs with the representation power of deep neural networks. Our model improves upon traditional graph-based extractive approaches and the vanilla GRU sequence model with no graph, and it achieves competitive results against other state-of-the-art multi-document summarization systems.

1 Introduction

Document summarization aims to produce fluent and coherent summaries covering salient information in the documents. Many previous summarization systems employ an extractive approach by identifying and concatenating the most salient text units (often whole sentences) in the document.

Traditional extractive summarizers produce the summary in two steps: sentence ranking and sentence selection. First, they utilize human-engineered features such as sentence position and length (Radev et al., 2004a), word frequency and importance (Nenkova et al., 2006; Hong and Nenkova, 2014), among others, to rank sentence salience. Then, they select summary-worthy sentences using a range of algorithms, such as graph centrality (Erkan and Radev, 2004), constraint optimization via Integer Linear Programming (McDonald, 2007; Gillick and Favre, 2009; Li et al., 2013), or Support Vector Regression (Li et al., 2007) algorithms. Optionally, sentence reordering (Lapata, 2003; Barzilay et al., 2001) can follow to improve coherence of the summary.

Recently, thanks to their strong representation power, neural approaches have become popular in text summarization, especially in sentence compression (Rush et al., 2015) and single-document summarization (Cheng and Lapata, 2016). Despite their popularity, neural networks still have issues when dealing with multi-document summarization (MDS). In previous neural multi-document summarizers (Cao et al., 2015, 2017), all the sentences in the same document cluster are processed independently. Hence, the relationships between sentences and thus the relationships between different documents are ignored. However, Christensen et al. (2013) demonstrates the importance of considering discourse relations among sentences in multi-document summarization.

This work proposes a multi-document summarization system that exploits the representational power of deep neural networks and the sentence relation information encoded in graph representations of document clusters. Specifically, we apply Graph Convolutional Networks (Kipf and Welling, 2017) on sentence relation graphs. First, we discuss three different techniques to produce sentence relation graphs, where nodes represent sentences in a cluster and edges capture the connections between sentences. Given a relation graph, our summarization model apples a Graph Convolutional Network (GCN), which takes in sentence embeddings from Recurrent Neural Networks as input node features. Through multiple layer-wise prop-

Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), pages 452–462,
Vancouver, Canada, August 3 - August 4, 2017.

agation, the GCN generates high-level hidden features for the sentences. We then obtain sentence salience estimations through a regression on top, and extract salient sentences in a greedy manner while avoiding redundancy.

We evaluate our model on the DUC 2004 multi-document summarization (MDS) task. Our model shows a clear advantage over traditional graph-based extractive summarizers, as well as a baseline GRU model that does not use any graph, and achieves competitive results with other state-of-the-art MDS systems. This work provides a new gateway to incorporating graph-based techniques into neural summarization.

2 Related Work

2.1 Graph-based MDS

Graph-based MDS models have traditionally employed surface level (Erkan and Radev, 2004; Mihalcea and Tarau, 2005; Wan and Yang, 2006) or deep level (Pardo et al., 2006; Antiqueira et al., 2009) approaches based on topological features and the number of nodes (Albert and Barabási, 2002). Efforts have been made to improve decision making of these systems by using discourse relationships between sentences (Radev, 2000; Radev et al., 2001). Erkan and Radev (2004) introduce LexRank to compute sentence importance based on the eigenvector centrality in the connectivity graph of inter-sentence cosine similarity. Mei et al. (2010) propose DivRank to balance the prestige and diversity of the top ranked vertices in information networks and achieve improved results on MDS. Christensen et al. (2013) build multi-document graphs to identify pairwise ordering constraints over the sentences by accounting for discourse relationships between sentences (Mann and Thompson, 1988). 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.

2.2 Summarization Using Neural Networks

Neural networks have recently been popular for text summarization (Kågebäck et al., 2014; Rush et al., 2015; Yin and Pei, 2015; Cao et al., 2016; Wang and Ling, 2016; Cheng and Lapata, 2016; Nallapati et al., 2016, 2017; See et al., 2017). For example, Rush et al. (2015) introduce a neural attention feed-forward network-based model for sentence compression. Wang and Ling (2016) employ encoder-decoder RNNs to effectively produce short abstractive summaries for opinions. Cao et al. (2016) develop a query-focused summarization system called AttSum which deals with saliency ranking and relevance ranking using query-attention-weighted CNNs.

Very recently, thanks to the large scale news article datasets (Hermann et al., 2015), Cheng and Lapata (2016) train an extractive summarization system with attention-based encoder-decoder RNNs to sequentially label summary-worth sentences in single documents. See et al. (2017), adopting an abstractive approach, augment the standard attention-based encoder-decoder RNNs with the ability to copy words from the source text via pointing and to keep track of what has been summarized. These models (Cheng and Lapata, 2016; See et al., 2017) achieve state-of-the-art performance on the DUC 2002 single-document summarization task. However, scaling up these RNN sequence-to-sequence approaches to the multi-document summarization task has not been successful, 1) due to the lack of large multi-document summarization datasets needed to train the computationally expensive sequence-to-sequence model, and 2) because of the inadequacy of RNNs to capture the complex discourse relations across multiple documents. Our multi-document summarization model resolves these issues 1) by breaking down the summarization task into salience estimation and sentence selection that do not require an expensive decoder architecture, and 2) by utilizing sentence relation graphs.

3 Method

Given a document cluster, our method extracts sentences as a summary in two steps: sentence salience estimation and sentence selection. Figure 1 illustrates our architecture for sentence salience estimation. Given a document cluster, we first build a sentence relation graph, where interacting sentence nodes are connected by edges. For each sentence, we apply an RNN with Gated Recurrent Units (GRU^{sent}) (Cho et al., 2014; Chung et al., 2014) and extract the last hidden state as the sentence embedding. We then apply Graph Convolutional Networks (Kipf and Welling, 2017) on the sentence relation graph with the sentence embeddings as the input node features, to produce final sentence embeddings that reflect the graph representation. Thereafter, a second level GRU (GRU^{doc}) produces the entire cluster embedding

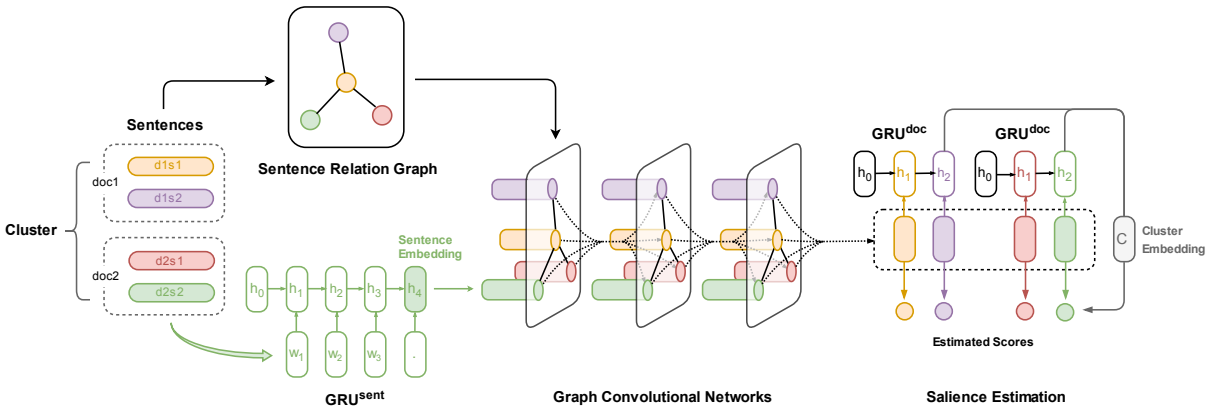

Figure 1: Illustration of our architecture for sentence salience estimation. In this example, there are two documents in the cluster and each document has two sentences. Sentences are processed by the GRU^{sent} to get input sentence embeddings. The GCN takes the input sentence embeddings and the sentence relation graph, and outputs high-level hidden features for individual sentences. GRU^{doc} produces the cluster embedding from the output sentence embeddings. The salience is estimated from the output sentence embeddings and the cluster embedding. w_i: the word embedding for i-th word. h_i: the hidden state of GRU at i-th step.

by sequentially connecting the final sentence embeddings. We estimate the salience of each sentence from the final sentence embeddings and the cluster embedding. Finally, based on the estimated salience scores, we select sentences in a greedy way until reaching the length limit.

3.1 Graph Representation of Clusters

To best evaluate the architecture, we consider three graph representation methods to model sentence relationships within clusters. First, as prior methods in representing document clusters often adhere to the standard of cosine similarity (Erkan and Radev, 2004), our initial baseline approach naturally used this representation. Specifically, we add an edge between two sentences if the tf-idf cosine similarity measure between them, using the bag-of-words model, is above a threshold of 0.2.

Secondly, the G-Flow system (Christensen et al., 2013) utilizes discourse relationships between sentences to create its graph representations, known as Approximate Discourse Graph (ADG). The ADG constructs edges between sentences by counting discourse relation indicators such as deverbal noun references, event and entity continuations, discourse markers, and co-referent mentions. These features allow characterization of sentence relationships, rather than simply their similarity.

While G-Flow's ADG provides many improvements from baseline graph representations, it suffers several disadvantages that diminish its ability

Personalization Features
• Position in Document
• From 1^{st} 3 Sentences?
• No. of Proper Nouns
• > 20 Tokens in Sentence?
• Sentence Length
• No. of Co-referent Verb Mentions
• No. of Co-referent Common Noun Mentions
• No. of Co-referent Proper Noun Mentions

Table 1: List of features that were input to the regression function in obtaining sentence personalization scores.

to aid salience prediction when given to the neural network. Specifically, the ADG lacks much diversity in its assigned edge weights. Because the 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, importance — 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). 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 $s(v)$, which can be viewed as weighting of sentences, is calculated for every sentence v to account for surface features in each sentence. These features, listed in Table 1, are used as input for linear regression, as per Christensen et al. (2013). The regression is applied to each sentence to obtain the personalization score, $s(v)$. 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

$$w_{PDG}(u, v) = \frac{w_{ADG}(u, v)s(v)}{\sum_{u' \in V} w_{ADG}(u', v)s(u')} \quad (1)$$

The inclusion of the sentence personalization scores allows the PDG to account for macro-level features in each sentence, augmenting information for salience estimation. To provide more clarity, we include a figure of the PDG in later sections.

Although it may be possible to incorporate the sentence personalization features later into the salience estimation network, we chose to encode them in the PDG to improve the edge weight distribution of sentence relation graphs and to make our salience estimation architecture methodically consistent. Additionally, in order to maintain consistency between graph representations, following 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.

3.2 Graph Convolutional Networks

We apply Graph Convolutional Networks (GCN) from Kipf and Welling (2017) on top of the sentence relation graph. In this subsection, we explain in detail the formulation of GCN, and how GCN produces the final sentence embeddings.

The goal of GCN is to learn a function $f(X, A)$ that takes as input:

- $A \in \mathbb{R}^{N \times N}$, the adjacency matrix of graph $\mathcal{G}$, where N is the number of nodes in $\mathcal{G}$.
- $X \in \mathbb{R}^{N \times D}$, the input node feature matrix, where D is the dimension of input node feature vectors.

and outputs high-level hidden features for each node, $Z \in \mathbb{R}^{N \times F}$, that encapsulate the graph structure. F is the dimension of output feature vectors. The function $f(X, A)$ takes a form of layer-wise propagation based on neural networks. We compute the activation matrix in the $(l+1)^{th}$ layer as $H^{(l+1)}$, starting from $H^0 = X$. The output of L-layer GCN is $Z = f(X, A) = H^{(L)}$.

To introduce the formulation, consider a simple form of layer-wise propagation:

$$H^{(l+1)} = \sigma\left(AH^{(l)}W^{(l)}\right) \quad (2)$$

where σ is an activation function such as $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$. $W^{(l)}$ is the parameter to learn in the l^{th} layer. Eq 2 has two limitations. First, multiplying by A means that for each node, we sum up the feature vectors of all neighboring nodes but not the node itself. We fix this by adding self-loops in the graph. Second, since A is not normalized, multiplying by A will change the scale of feature vectors. To overcome this, we apply a symmetric normalization by using $D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ where D is the node degree matrix. These two renormalization tricks result in the following propagation rule:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \quad (3)$$

where $\tilde{A} = A + I_N$ is the adjacency matrix of the graph $\mathcal{G}$ with added self-loops (I_N is the identity matrix). $\tilde{D}$ is the degree matrix with $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. Kipf and Welling (2017) also provide a theoretical justification of Eq 3 as a first-order approximation of spectral graph convolution (Hammond et al., 2011; Defferrard et al., 2016).

As an example, if we have a two-layer GCN, we first calculate $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ in a preprocessing step, and then produce

$$Z = f(X, A) = \sigma\left(\hat{A}\,\sigma\left(\hat{A}XW^{(0)}\right)W^{(1)}\right)$$

3.3 Sentence Embeddings

As the input node features X of GCN, we use sentence embeddings calculated by GRU^{sent}.

Given a document cluster C with N sentences $(s_1, s_2, ..., s_N)$ in total, for each sentence s_i of L words $(w_1, w_2, ..., w_L)$, GRU^{sent} recurrently updates hidden states at each time step t:

$$\mathbf{h}_t^{sent} = \mathrm{GRU}^{sent}(\mathbf{h}_{t-1}^{sent}, \mathbf{w}_t) \quad (4)$$

where $\mathbf{w}_t$ is the word embedding for w_t, $\mathbf{h}_t^{sent}$ is the hidden state of GRU^{sent}. $\mathbf{h}_0$ is initialized as a zero vector, and the input sentence embedding $\mathbf{x}_i$ is the last hidden state:

$$\mathbf{x}_i = \mathbf{h}_L^{sent} \quad (5)$$

All sentence embeddings from the given document cluster are grouped as the node feature matrix X:

$$X = \begin{bmatrix} — & \mathbf{x}_1^T & — \\ — & \mathbf{x}_2^T & — \\ & \vdots & \\ — & \mathbf{x}_N^T & — \end{bmatrix} \quad (6)$$

X is fed into GCN subsequently to obtain the final sentence embeddings $\mathbf{s}_i$ that incorporate the graph representation of sentence relationships:

$$Z = f(X, A) = \begin{bmatrix} — & \mathbf{s}_1^T & — \\ — & \mathbf{s}_2^T & — \\ & \vdots & \\ — & \mathbf{s}_N^T & — \end{bmatrix} \quad (7)$$

3.4 Cluster Embedding

Additionally, in order to have a global view of the entire document cluster, we apply a second-level RNN, GRU^{doc}, to encode the entire document cluster. Given a document cluster C with M documents $(d_1, d_2, ..., d_M)$, for document d_i with $|d_i|$ sentences, GRU^{doc} first builds the document embedding $\mathbf{d}_i$ on top of sentence embeddings:

$$\mathbf{h}_t^{doc} = \text{GRU}^{doc}(\mathbf{h}_{t-1}^{doc}, \mathbf{s}_t) \quad (8)$$

$$\mathbf{d}_i = \mathbf{h}_{|d_i|}^{doc} \quad (9)$$

where $\mathbf{s}_t$ is the sentence embedding in the document d_i. In Eq 9, we extract the last hidden state as the document embedding for d_i. In Eq 10, we average over document embeddings to produce the cluster embedding $\mathbf{C}$:

$$\mathbf{C} = \frac{1}{M} \sum_{i=1}^{M} \mathbf{d}_i \quad (10)$$

All the GRUs we used are forward. We also experimented with backward GRUs and bi-directional GRUs, but neither of them meaningfully improved upon forward GRUs.

3.5 Salience Estimation

For the sentence s_i in the cluster C, we calculate the salience of s_i as the following, similarly to the attention mechanism in neural machine translation (Bahdanau et al., 2015):

$$f(s_i) = \mathbf{v}^T \tanh(\mathbf{W}_1 \mathbf{C} + \mathbf{W}_2 \mathbf{s}_i) \quad (11)$$

$$\text{salience}(s_i) = \frac{f(s_i)}{\sum_{s_j \in C} f(s_j)} \quad (12)$$

where $\mathbf{v}, \mathbf{W}_1, \mathbf{W}_2$ are learnable parameters. In Eq 11, we first calculate the score $f(s_i)$ by considering the sentence embedding itself, $\mathbf{s}_i$, and the cluster embedding $\mathbf{C}$ for the global context of the multi-document. The score is then normalized as $\text{salience}(s_i)$ via softmax in Eq 12.

3.6 Training

The model parameters include the parameters in GRU^{sent} and GRU^{doc}, the weights in GCN layers, and the parameters for salience estimation $(\mathbf{v}, \mathbf{W}_1, \mathbf{W}_2)$. Parameters in GRU^{sent} and GRU^{doc} are not shared. The model is trained end-to-end to minimize the following cross-entropy loss between the salience prediction and the normalized ROUGE score of each sentence:

$$\mathcal{L} = -\sum_{C} \sum_{s_i \in C} R(s_i) \log(\text{salience}(s_i)) \quad (13)$$

$R(s_i)$ is calculated by $R(s_i) = \text{softmax}(\alpha\, r(s_i))$, where $r(s_i)$ is the average of ROUGE-1 and ROUGE-2 Recall scores of sentence s_i by measuring with the ground-truth human-written summaries. α is a constant rescaling factor to make the distribution sharper. The value of α is determined from the validation data set. $\alpha r(s_i)$ is then normalized across the cluster via softmax, similarly to Eq 12.

3.7 Sentence Selection

Given the salience score estimation, we apply a simple greedy procedure to select sentences. Sentences with higher salience scores have higher priorities. First, we sort sentences in descending order of the salience scores. Then, we select one sentence from the top of the list and append to the summary if the sentence is of reasonable length (8-55 words, as in (Erkan and Radev, 2004)) and is not redundant. The sentence is redundant if the tf-idf cosine similarity between the sentence and the current summary is above 0.5 (Hong and Nenkova, 2014). We select sentences this way until we reach the length limit.

4 Experiments

In this section, we evaluate our model on benchmark MDS data sets, and compare with other state-of-the-art systems. We aim to show that our model, by combining sentence relations in graphs with the representation power of deep neural networks, can improve upon other traditional graph-based extractive approaches and the vanilla GRU model which does not use any graph. In addition,

	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: Statistics for DUC Multi-Document Summarization Data Sets.

we further study the effect of graph and different graph representations on the summarization performance and investigate the correlation of graph structure and sentence salience estimation.

4.1 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 2 shows the statistics of the data sets. We use DUC 2001, 2002, 2003 and 2004 containing 30, 59, 30 and 50 clusters of nearly 10 documents each respectively. 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, with stemming and stop words not removed as suggested by Owczarzak et al. (2012).

4.2 Experimental Setup

We conduct four experiments on our model: three using each of the three types of graphs discussed earlier, and one without using any graph. In the experiments with graphs, for each document cluster, we tokenize all the documents into sentences and generate a graph representation of their relations by the three methods: Cosine Similarity Graph, Approximate Discourse Graph (ADG) from G-Flow, and our Personalized Discourse Graph (PDG). Note that for the Cosine Similarity Graph, we compute the tf-idf cosine similarity for every pair of sentences using the bag-of-word model and add an edge for similarity above 0.2. The weight of the edge is the value of similarity. 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.

We use 300-dimensional pre-trained word2vec embeddings (Mikolov et al., 2013) as input to GRU^{sent} in Eq 4. The word embeddings are fine-tuned during training. We use three GCN hidden layers ($L = 3$). The hidden states in GRU^{sent}, GCN hidden layers, and GRU^{doc} are all 300-dimensional vectors ($D = F = 300$).

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

The rescaling factor α in the objective function (Eq 13) is chosen as 40 from $\{10, 20, 30, 40, 50, 100\}$ based on the validation performance. The objective function is optimized using Adam (Kingma and Ba, 2015) stochastic gradient descent with a learning rate of 0.001 and a batch size of 1. We use gradient clipping with a maximum gradient norm of 1.0. The model is validated every 10 iterations, and the training is stopped early if the validation performance does not improve for 10 consecutive steps. We trained using a single Tesla K80 GPU. For all the experiments, the training took approximately 30 minutes until a stop.

4.3 Results

Table 3 summarizes our results. First we take our simple GRU model as the baseline of the RNN-based regression approach. As seen from the table, the addition of Cosine Similarity Graph on top of the GRU clearly boosts the performance. Furthermore, the addition of ADG from G-Flow gives a slighly better performance. Our Personalized Discourse Graph (PDG) enhances the R-1 score by more than 1.50. The improvement indicates that the combination of graphs and GCNs processes sentence relations across documents better than the vanilla RNN sequence models.

To gain a global view of our performance, we also compare our result with other baseline multi-document summarizers and the state-of-the-

	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 4: 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.

art systems related to our regression method. We compute ROUGE scores from the actual output summary of each system. We run the G-Flow code released by Christensen et al. (2013) to get the output summary of the G-Flow system. The output summary of other systems are compiled in Hong et al. (2014). To ensure fair comparison, we use ROUGE-1.5.5 with the same parameters as in Hong et al. (2014) across all methods: -n 2 -m -l 100 -x -c 95 -r 1000 -f A -p 0.5 -t 0.

From Table 3, we observe that 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. The major difference is that RegSum performs regression on word level and estimates the salience of each word through a rich set of word features, such as frequency, grammar, context, and hand-crafted dictionaries. RegSum then computes sentence salience based on the word scores. On the other hand, our model simply works on sentence level, spotlighting sentence relations encoded as a graph. Incorporating more word-level features into our discourse graphs may be an interesting future direction to explore.

4.4 Discussion

As shown in Table 3, our graph-based models outperform the vanilla GRU model, which has no graph. Additionally, for the three graphs we consider, PDG improves R-1 score by 0.82 over ADG, and ADG outperforms the Cosine Similarity Graph by 0.08 on the R-1 score. While the Cosine Similarity Graph encodes general word-level connections between sentences, discourse graphs, especially our personalized version, specialize in representing the narrative and logical relations between sentences. Therefore, we hypothesize that the PDG provides a more informative guide to estimating the importance of each sentence. In an attempt to better understand the results and validate the effect of sentence relation graphs (especially of the PDG), we have conducted the analysis that follows.

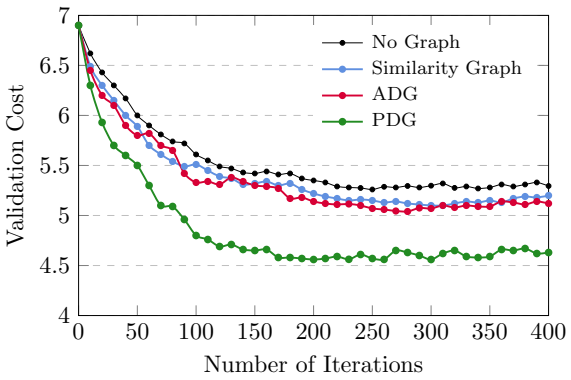

Figure 2: Visualization of the learning curves for the four experiments. The vertical axis displays the validation costs in the interval 4.0 - 7.0.

	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 5: 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.

Training Statistics. We compare the learning curves of the four different settings: GRU without any graph, GRU+GCN with the Cosine Similarity Graph, GRU+GCN with ADG, and GRU+GCN with PDG (see Table 4 & Figure 2). 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

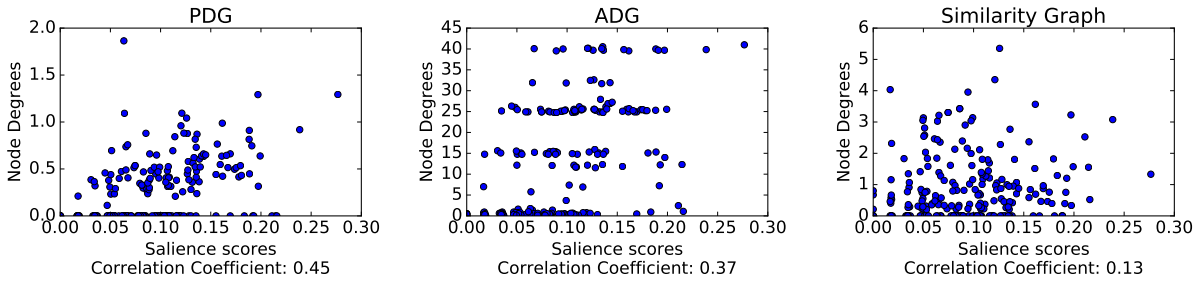

Figure 3: Visualization of the relationship between salience score and node degree for the three graph representation methods. Cluster d30011t from DUC 2004 is chosen as an example.

than the models with graphs. For the three graph methods, ADG converges faster and has better validation performance than the Cosine Similarity Graph. 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.

Graph Statistics. We also analyze the characteristics of the three graph representation methods on DUC 2004 document clusters. Table 5 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 3. 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 our 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.

Node Degree and Salience. In Table 5, we also calculate the correlation coefficient ρ, per graph, between the degree of each sentence node and its salience score. We observe that all the graph representations show positive correlation between the node degree and the salience score. Moreover, the order of correlation strength is PDG > ADG > Cosine Similarity Graph. Though node degree is a simple measure of these graphs, this observation supports our hypothesis on the efficacy of sentence relation graphs, particularly of PDGs, to provide a guide to salience estimation. [1]

As a case study to illustrate our observation, we chose one cluster (d30011t) from DUC 2004. Figure 3 shows the scatter plots of the node degree and salience score of each sentence.

Visualization of the PDG. Finally, to demonstrate the functionality of the PDG and complement our discussion from Section 3.1, we visualize the PDG on cluster d30011t with the salience score on each node in Figure 4 (also see Figure 5 for the actual sentences).

From the visualization, it can be observed that the nodes representing salient sentences (such as (d_6, s_8), (d_6, s_7), and (d_2, s_4)) tend to have higher degrees in the PDG. We can also observe that the PDG represents edges which connect nodes of sentences from different documents, in contrast with the traditional sequence model.

From Figure 5, we note that the most salient sentence (d_6, s_8) actually describes much of the reference summary. As an example of discourse relation, (d_6, s_7) and (d_2, s_4), the two nodes connected to (d_6, s_8), provide the background for

[1] However, we shall add that simply selecting sentences of highest node degrees in PDGs did not itself produce good summaries, compared to our GCN model. Hence, we utilize the graph representations specifically as inputs to the GCN.

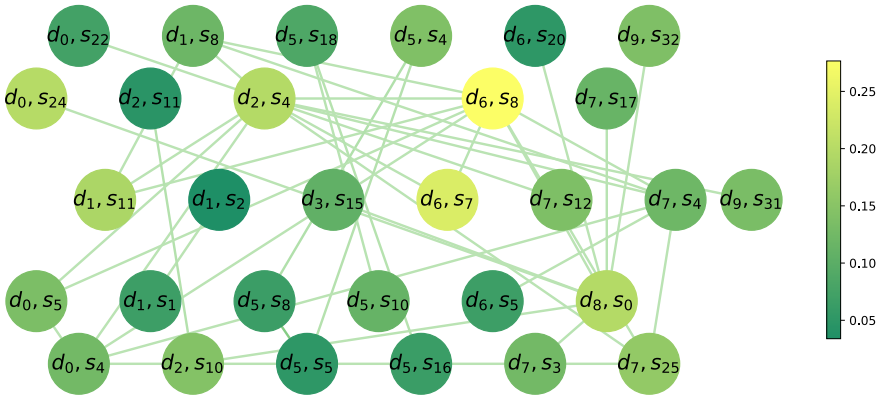

Figure 4: Visualization of the PDG on cluster d30011t. Each node is a sentence, with label (DocumentID, SentenceID). The node color represents the salience score (see the color bar). For simplicity, we only display edges of weight above 0.03. Best viewed in color.

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. (...)
Sent-label (6,8): Anwar was ... after two weeks of nationwide rallies at which he called for **government reform** and Mahathir's resignation, he was **arrested** ....
Sent-label (6,7): The two had differed over **economic policy** and Anwar has said Mahathir feared he was a threat to his 17-year rule.
Sent-label (2,4): Mahathir and Anwar had differed over **economic policy** and Anwar says Mahathir feared him as an alternative leader.
Sent-label (0,22): Before his arrest, Anwar designated his wife, Azizah Ismail, as the leader of his new ``reform'' movement.

Figure 5: Reference summary and illustrative sentences from cluster d30011t.

(d_6, s_8), even though they do not share many words in common with it. On the other hand, (d_0, s_{22}), which is only connected with (d_2, s_4), is not salient as it does not provide a central message for the summary.

5 Conclusion

In this paper, we presented 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, can capture sentence relations across documents and demonstrates improved salience prediction and summarization, achieving competitive performance with current state-of-the-art systems. Furthermore, through multiple analyses, we have validated the efficacy of sentence relation graphs, particularly of PDG, to help to learn the salience of sentences. This work shows the promise of the GCN models and of discourse graphs applied to processing multi-document inputs.

Acknowledgements

We would like to thank the members of the Sapphire Project (University of Michigan and IBM), as well as all the anonymous reviewers for their helpful suggestions on this work. This material is based in part upon work supported by IBM under contract 4915012629. Any opinions, findings, conclusions, or recommendations expressed herein are those of the authors and do not necessarily reflect the views of IBM.

References

Réka Albert and Albert-László Barabási. 2002. Statistical mechanics of complex networks. *Reviews of modern physics* 74(1):47.

Lucas Antiqueira, Osvaldo N Oliveira, Luciano da Fontoura Costa, and Maria das Graças Volpe Nunes. 2009. A complex network approach to text summarization. *Information Sciences* 179(5):584–599.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Ben-

gio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Regina Barzilay, Noemie Elhadad, and Kathleen R McKeown. 2001. Sentence ordering in multidocument summarization. In *Proceedings of the first international conference on Human language technology research*.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2017. Improving multi-document summarization via text classification. In *AAAI*.

Ziqiang Cao, Wenjie Li, Sujian Li, Furu Wei, and Yanran Li. 2016. Attsum: Joint learning of focusing and summarization with neural attention. In *COLING*.

Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. 2015. Ranking with recursive neural networks and its application to multi-document summarization. In *AAAI*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *ACL*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *NAACL*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *NIPS 2014 Deep Learning and Representation Learning Workshop* .

John M Conroy, Judith D Schlesinger, Jade Goldstein, and Dianne P Oleary. 2004. Left-brain/right-brain multi-document summarization. In *Proceedings of the Document Understanding Conference (DUC 2004)*.

John M Conroy, Judith D Schlesinger, Jeff Kubina, Peter A Rankel, and Dianne P O'Leary. 2011. Classy 2011 at tac: Guided and multi-lingual summaries and evaluation metrics. *TAC* 11:1–8.

John M Conroy, Judith D Schlesinger, and Dianne P O'Leary. 2006. Topic-focused multi-document summarization using an approximate oracle score. In *COLING/ACL*.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* 22:457–479.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*. Association for Computational Linguistics, pages 10–18.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *NAACL*.

David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. 2011. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis* 30(2):129–150.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*.

Kai Hong, John M Conroy, Benoit Favre, Alex Kulesza, Hui Lin, and Ani Nenkova. 2014. A repository of state of the art and competitive baseline summaries for generic news summarization. In *LREC*.

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *EACL*.

Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@ EACL*. Citeseer, pages 31–39.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *ACL*.

Chen Li, Xian Qian, and Yang Liu. 2013. Using supervised bigram-based ilp for extractive summarization. In *ACL*.

Sujian Li, You Ouyang, Wei Wang, and Bin Sun. 2007. Multi-document summarization using support vector regression. In *Proceedings of DUC*. Citeseer.

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse* 8(3):243–281.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *European Conference on Information Retrieval*.

Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. Divrank: the interplay of prestige and diversity in information networks. In *SIGKDD*.

Rada Mihalcea and Paul Tarau. 2005. A language independent algorithm for single and multiple document summarization. In *IJCNLP*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. *AAAI* .

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *CoNLL*.

Ani Nenkova, Lucy Vanderwende, and Kathleen McKeown. 2006. A compositional context sensitive multi-document summarizer: exploring the factors that influence summarization. In *SIGIR*.

Karolina Owczarzak, John M Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*. Association for Computational Linguistics, pages 1–9.

Thiago Pardo, Lucas Antiqueira, Maria Nunes, Osvaldo Oliveira, and Luciano da Fontoura Costa. 2006. Modeling and evaluating summaries using complex networks. *Computational Processing of the Portuguese Language* pages 1–10.

Dragomir R Radev. 2000. A common theory of information fusion from multiple text sources step one: cross-document structure. In *Proceedings of the 1st SIGdial workshop on Discourse and dialogue-Volume 10*. Association for Computational Linguistics, pages 74–83.

Dragomir R Radev, Timothy Allison, Sasha Blair-Goldensohn, John Blitzer, Arda Celebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, Danyu Liu, et al. 2004a. Mead-a platform for multidocument multilingual text summarization. In *LREC*.

Dragomir R Radev, Sasha Blair-Goldensohn, Zhu Zhang, and Revathi Sundara Raghavan. 2001. Newsinessence: A system for domain-independent, real-time news clustering and multi-document summarization. In *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, pages 1–4.

Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. 2004b. Centroid-based summarization of multiple documents. *Information Processing & Management* 40(6):919–938.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Xiaojun Wan and Jianwu Yang. 2006. Improved affinity graph based multi-document summarization. In *NAACL*.

Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. In *NAACL*.

Wenpeng Yin and Yulong Pei. 2015. Optimizing sentence modeling and selection for document summarization. In *IJCAI*.