Neural Latent Extractive Document Summarization

Xingxing Zhang[†], **Mirella Lapata**[‡], **Furu Wei**[†] and **Ming Zhou**[†]
[†]Microsoft Research Asia, Beijing, China
[‡]Institute for Language, Cognition and Computation,
School of Informatics, University of Edinburgh, UK
{xizhang,fuwei,mingzhou}@microsoft.com,mlap@inf.ed.ac.uk

Abstract

Extractive summarization models require sentence-level labels, which are usually created heuristically (e.g., with rule-based methods) given that most summarization datasets only have document-summary pairs. Since these labels might be suboptimal, we propose a latent variable extractive model where sentences are viewed as latent variables and sentences with activated variables are used to infer gold summaries. During training the loss comes *directly* from gold summaries. Experiments on the CNN/Dailymail dataset show that our model improves over a strong extractive baseline trained on heuristically approximated labels and also performs competitively to several recent models.

1 Introduction

Document summarization aims to automatically rewrite a document into a shorter version while retaining its most important content. Of the many summarization paradigms that have been identified over the years (see Mani 2001 and Nenkova and McKeown 2011 for comprehensive overviews), two have consistently attracted attention: *extractive* approaches generate summaries by copying parts of the source document (usually whole sentences), while *abstractive* methods may generate new words or phrases which are not in the document.

A great deal of previous work has focused on extractive summarization which is usually modeled as a sentence ranking or binary classification problem (i.e., sentences which are top ranked or predicted as `True` are selected as summaries). Early attempts mostly leverage human-engineered features (Filatova and Hatzivassiloglou, 2004) coupled with binary classifiers (Kupiec et al., 1995), hidden Markov models (Conroy and O'leary, 2001), graph based methods (Mihalcea, 2005), and integer linear programming (Woodsend and Lapata, 2010).

The successful application of neural network models to a variety of NLP tasks and the availability of large scale summarization datasets (Hermann et al., 2015; Nallapati et al., 2016) has provided strong impetus to develop data-driven approaches which take advantage of continuous-space representations. Cheng and Lapata (2016) propose a hierarchical long short-term memory network (LSTM; Hochreiter and Schmidhuber 1997) to learn context dependent sentence representations for a document and then use yet another LSTM decoder to predict a binary label for each sentence. Nallapati et al. (2017) adopt a similar approach, they differ in their neural architecture for sentence encoding and the features used during label prediction, while Narayan et al. (2018) equip the same architecture with a training algorithm based on reinforcement learning. Abstractive models (Nallapati et al., 2016; See et al., 2017; Paulus et al., 2017) are based on sequence-to-sequence learning (Sutskever et al., 2014; Bahdanau et al., 2015), however, most of them underperform or are on par with the baseline of simply selecting the leading sentences in the document as summaries (but see Paulus et al. 2017 and Celikyilmaz et al. 2018 for exceptions).

Although seemingly more successful than their abstractive counterparts, extractive models require sentence-level labels, which are not included in most summarization datasets (only document and gold summary pairs are available). Sentence labels are usually obtained by rule-based methods (Cheng and Lapata, 2016) or by maximizing the ROUGE score (Lin, 2004) between a subset of sentences and the human written summaries (Nallapati et al., 2017). These methods do not fully exploit the human summaries, they only create `True`/`False` labels which might be suboptimal.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 779–784
Brussels, Belgium, October 31 - November 4, 2018.

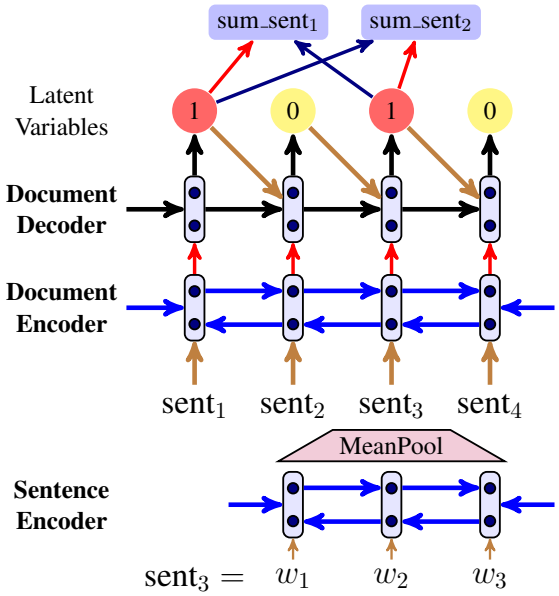

Figure 1: Latent variable extractive summarization model. sent_i is a sentence in a document and sum_sent_i is a sentence in a gold summary of the document.

In this paper we propose a latent variable extractive model and view labels of sentences in a document as binary latent variables (i.e., zeros and ones). Instead of maximizing the likelihood of "gold" standard labels, the latent model directly maximizes the likelihood of human summaries given selected sentences. Experiments on the CNN/Dailymail dataset (Hermann et al., 2015) show that our latent extractive model improves upon a strong extractive baseline trained on rule-based labels and also performs competitively to several recent models.

2 Model

We first introduce the neural extractive summarization model upon which our latent model is based on. We then describe a sentence compression model which is used in our latent model and finally move on to present the latent model itself.

2.1 Neural Extractive Summarization

In extractive summarization, a subset of sentences in a document is selected as its summary. We model this problem as an instance of sequence labeling. Specifically, a document is viewed as a sequence of sentences and the model is expected to predict a `True` or `False` label for each sentence, where `True` indicates that the sentence should be included in the summary. It is assumed that during training sentences and their labels in each document are given (methods for obtaining these labels are discussed in Section 3).

As shown in the lower part of Figure 1, our extractive model has three parts: a *sentence encoder* to convert each sentence into a vector, a *document encoder* to learn sentence representations given surrounding sentences as context, and a *document decoder* to predict sentence labels based on representations learned by the document encoder. Let $\mathcal{D} = (S_1, S_2, \ldots, S_{|\mathcal{D}|})$ denote a document and $S_i = (w_1^i, w_2^i, \ldots, w_{|S_i|}^i)$ a sentence in $\mathcal{D}$ (where w_j^i is a word in S_i). Let $Y = (y_1, \ldots, y_{|\mathcal{D}|})$ denote sentence labels. The sentence encoder first transforms S_i into a list of hidden states $(\mathbf{h}_1^i, \mathbf{h}_2^i, \ldots, \mathbf{h}_{|S_i|}^i)$ using a Bidirectional Long Short-Term Memory Network (Bi-LSTM; Hochreiter and Schmidhuber 1997; Schuster and Paliwal 1997). Then, the sentence encoder yields $\mathbf{v}_i$, the representation of S_i, by averaging these hidden states (also see Figure 1):

$$\mathbf{v}_i = \frac{1}{|S_i|} \sum_j \mathbf{h}_j^i \tag{1}$$

In analogy to the sentence encoder, the document encoder is another Bi-LSTM but applies on the sentence level. After running the Bi-LSTM on a sequence of sentence representations $(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{|\mathcal{D}|})$, we obtain context dependent sentence representations $(\mathbf{h}_1^E, \mathbf{h}_2^E, \ldots, \mathbf{h}_{|\mathcal{D}|}^E)$.

The document decoder is also an LSTM which predicts sentence labels. At each time step, it takes the context dependent sentence representation of S_i produced by the document encoder as well as the prediction in the previous time step:

$$\mathbf{h}_i^D = \text{LSTM}(\mathbf{h}_{i-1}^D, \begin{bmatrix} \mathbf{W}_e\ e(y_{i-1}) \\ \mathbf{h}_i^E \end{bmatrix}) \tag{2}$$

where $\mathbf{W}_e \in \mathbb{R}^{d \times 2}$ is the label embedding matrix (d is the hidden dimension for the document decoder LSTM) and y_{i-1} is the prediction at time step $i-1$; the predicted label distribution for y_i is:

$$p(y_i | y_{1:i-1}, \mathbf{h}_{i-1}^D) = \text{softmax}(\mathbf{W}_o\, \mathbf{h}_i^D) \tag{3}$$

where $\mathbf{W}_o \in \mathbb{R}^{2 \times d}$.

The model described above is usually trained by minimizing the negative log-likelihood of sentence labels in training documents; it is almost identical to Cheng and Lapata (2016) except that

we use a word-level long short-term memory network coupled with mean pooling to learn sentence representations, while they use convolutional neural network coupled with max pooling (Kim et al., 2016).

2.2 Sentence Compression

We train a sentence compression model to map a sentence selected by the extractive model to a sentence in the summary. The model can be used to evaluate the quality of a selected sentence with respect to the summary (i.e., the degree to which it is similar) or rewrite an extracted sentence according to the style of the summary.

For our compression model we adopt a standard attention-based sequence-to-sequence architecture (Bahdanau et al., 2015; Rush et al., 2015). The training set for this model is generated from the same summarization dataset used to train the exractive model. Let $\mathcal{D} = (S_1, S_2, \ldots, S_{|\mathcal{D}|})$ denote a document and $\mathcal{H} = (H_1, H_2, \ldots, H_{|\mathcal{H}|})$ its summary. We view each sentence H_i in the summary as a target sentence and assume that its corresponding source is a sentence in $\mathcal{D}$ most similar to it. We measure the similarity between source sentences and candidate targets using ROUGE, i.e., $S_j = \text{argmax}_{S_j} \text{ROUGE}(S_j, H_i)$ and $\langle S_j, H_i \rangle$ is a training instance for the compression model. The probability of a sentence $\hat{H}_i$ being the compression of $\hat{S}_j$ (i.e., $p_{s2s}(\hat{H}_i|\hat{S}_j)$) can be estimated with a trained compression model.

2.3 Latent Extractive Summarization

Training the extractive model described in Section 2.1 requires sentence-level labels which are obtained heuristically (Cheng and Lapata, 2016; Nallapati et al., 2017). Our latent variable model views sentences in a document as binary variables (i.e., zeros and ones) and uses sentences with activated latent variables (i.e., ones) to infer gold summaries. The latent variables are predicted with an extractive model and the loss during training comes from gold summaries *directly*.

Let $\mathcal{D} = (S_1, S_2, \ldots, S_{|\mathcal{D}|})$ denote a document and $\mathcal{H} = (H_1, H_2, \ldots, H_{|\mathcal{H}|})$ its human summary (H_k is a sentence in $\mathcal{H}$). We assume that there is a latent variable $z_i \in \{0, 1\}$ for each sentence S_i indicating whether S_i should be selected, and $z_i = 1$ entails it should. We use the extractive model from Section 2.1 to produce probability distributions for latent variables (see Equation (3)) and obtain them by sampling $z_i \sim p(z_i|z_{1:i-1}, \mathbf{h}^D_{i-1})$ (see Figure 1). $\mathcal{C} = \{S_i|z_i = 1\}$, the set of sentences whose latent variables equal to one, are our current extractive summaries. Without loss of generality, we denote $\mathcal{C} = (C_1, \ldots, C_{|\mathcal{C}|})$. Then, we estimate how likely it is to infer the human summary $\mathcal{H}$ from $\mathcal{C}$. We estimate the likelihood of summary sentence H_l given document sentence C_k with the compression model introduced in Section 2.2 and calculate the normalized[1] probability s_{kl}:

$$s_{kl} = \exp\left(\frac{1}{|H_l|} \log p_{s2s}(H_l|C_k)\right) \quad (4)$$

The score R_p measures the extent to which $\mathcal{H}$ can be inferred from $\mathcal{C}$:

$$R_p(\mathcal{C}, \mathcal{H}) = \frac{1}{|\mathcal{C}|} \sum_{k=1}^{|\mathcal{C}|} \max_{l=1}^{|\mathcal{H}|} s_{kl} \quad (5)$$

For simplicity, we assume one document sentence can only find one summary sentence to explain it. Therefore, for all H_l, we only retain the most evident s_{kl}. $R_p(\mathcal{C}, \mathcal{H})$ can be viewed as the "precision" of document sentences with regard to summary sentences. Analogously, we also define R_r, which indicates the extent to which $\mathcal{H}$ can be covered by $\mathcal{C}$:

$$R_r(\mathcal{C}, \mathcal{H}) = \frac{1}{|\mathcal{H}|} \sum_{l=1}^{|\mathcal{H}|} \max_{k=1}^{|\mathcal{C}|} s_{kl} \quad (6)$$

$R_r(\mathcal{C}, \mathcal{H})$ can be viewed as the "recall" of document sentences with regard to summary sentences. The final score $R(\mathcal{C}, \mathcal{H})$ is the weighted sum of the two:

$$R(\mathcal{C}, \mathcal{H}) = \alpha\, R_p(\mathcal{C}, \mathcal{H}) + (1 - \alpha)\, R_r(\mathcal{C}, \mathcal{H}) \quad (7)$$

Our use of the terms "precision" and "recall" is reminiscent of relevance and coverage in other summarization work (Carbonell and Goldstein, 1998; Lin and Bilmes, 2010; See et al., 2017).

We train the model by minimizing the negative expected $R(\mathcal{C}, \mathcal{H})$:

$$\mathcal{L}(\theta) = -\mathbb{E}_{(z_1,\ldots,z_{|\mathcal{D}|}) \sim p(\cdot|\mathcal{D})}[R(\mathcal{C}, \mathcal{H})] \quad (8)$$

where $p(\cdot|\mathcal{D})$ is the distribution produced by the neural extractive model (see Equation (3)). Unfortunately, computing the expectation term is prohibitive, since the possible latent variable combinations are exponential. In practice, we approximate this expectation with a single sample from

[1] We also experimented with unnormalized probabilities (i.e., excluding the exp in Equation (4)), however we obtained inferior results.

the distribution of $p(\cdot|\mathcal{D})$. We use the REINFORCE algorithm (Williams, 1992) to approximate the gradient of $\mathcal{L}(\theta)$:

$$\nabla\mathcal{L}(\theta) \approx \sum_{i=1}^{|\mathcal{D}|} \nabla \log p(z_i|z_{1:i-1}, \mathbf{h}_{i-1}^{D})[R(\mathcal{C},\mathcal{H}) - b_i]$$

Note that the model described above can be viewed as a reinforcement learning model, where $R(\mathcal{C},\mathcal{H})$ is the reward. To reduce the variance of gradients, we also introduce a baseline linear regression[2] model b_i (Ranzato et al., 2016) to estimate the expected value of $R(\mathcal{C},\mathcal{H})$. To avoid random label sequences during sampling, we use a pre-trained extractive model to initialize our latent model.

3 Experiments

Dataset and Evaluation We conducted experiments on the CNN/Dailymail dataset (Hermann et al., 2015; See et al., 2017). We followed the same pre-processing steps as in See et al. (2017). The resulting dataset contains 287,226 document-summary pairs for training, 13,368 for validation and 11,490 for test. To create sentence level labels, we used a strategy similar to Nallapati et al. (2017). We label the subset of sentences in a document that maximizes ROUGE (against the human summary) as `True` and all other sentences as `False`. Using the method described in Section 2.2, we created a compression dataset with 1,045,492 sentence pairs for training, 53,434 for validation and 43,382 for testing. We evaluated our models using full length F1 ROUGE (Lin, 2004) and the official `ROUGE-1.5.5.pl` script. We report ROUGE-1, ROUGE-2, and ROUGE-L.

Implementation We trained our extractive model on an Nvidia K80 GPU card with a batch size of 32. Model parameters were uniformly initialized to $[-\frac{1}{\sqrt{c}}, \frac{1}{\sqrt{c}}]$ (c is the number of columns in a weight matrix). We used Adam (Kingma and Ba, 2014) to optimize our models with a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. We trained our extractive model for 10 epochs and selected the model with the highest ROUGE on the validation set. We rescaled the gradient when its norm exceeded 5 (Pascanu et al., 2013) and regularized all LSTMs with a dropout rate of 0.3 (Srivastava et al., 2014; Zaremba et al., 2014). We also applied word dropout (Iyyer et al., 2015) at rate 0.2. We set the hidden unit size $d = 300$ for both word-level and sentence-level LSTMs and all LSTMs had one layer. We used 300 dimensional pre-trained FastText vectors (Joulin et al., 2017) to initialize our word embeddings. The latent model was initialized from the extractive model (thus both models have the same size) and we set the weight in Equation (7) to $\alpha = 0.5$. The latent model was trained with SGD, with learning rate 0.01 for 5 epochs. During inference, for both extractive and latent models, we rank sentences with $p(y_i = \texttt{True}|y_{1:i-1}, \mathcal{D})$ and select the top three as summary (see also Equation (3)).

[2]The linear regression model b_t is trained by minimizing the mean squared error between the prediction of b_t and $R(\mathcal{C},\mathcal{H})$.

Model	R-1	R-2	R-L
LEAD3	40.34	17.70	36.57
LEAD3 (Nallapati et al., 2017)	39.20	15.70	35.50
abstract	35.46	13.30	32.65
pointer+coverage	39.53	17.28	36.38
abstract-RL	**41.16**	15.75	**39.08**
abstract-ML+RL	39.87	15.82	36.90
SummaRuNNer	39.60	16.20	35.30
EXTRACT-CNN	40.11	17.52	36.39
REFRESH (Narayan et al., 2018)	40.00	18.20	36.60
EXTRACT	40.62	18.45	37.14
LATENT	41.05	**18.77**	37.54
LATENT+COMPRESS	36.69	15.43	34.33

Table 1: Results of different models on the CNN/Dailymail test set using full-length F1 ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L (R-L).

Comparison Systems We compared our model against LEAD3, which selects the first three leading sentences in a document as the summary and a variety of abstractive and extractive models. Abstractive models include a sequence-to-sequence architecture (Nallapati et al., 2016); *abstract*), its pointer generator variant (See et al. 2017; *pointer+coverage*), and two reinforcement learning-based models (Paulus et al. 2017; *abstract-RL* and *abstract-ML+RL*). We also compared our approach against an extractive model based on hierarchical recurrent neural networks (Nallapati et al. 2017; *SummaRuNNer*), the model described in Section 2.1 (EXTRACT) which encodes sentences using LSTMs, a variant which employs CNNs instead (Cheng and Lapata 2016; EXTRACT-CNN), as well as a similar system based on reinforcement learning (Narayan et al. 2018; REFRESH).

Results As shown in Table 1, EXTRACT, our extractive model outperforms LEAD3 by a wide margin. EXTRACT also outperforms previously published extractive models (i.e., *SummaRuNNer*, EXTRACT-CNN, and REFRESH). However, note that *SummaRuNNer* generates anonymized summaries (Nallapati et al., 2017) while our models generate non-anonymized ones, and therefore the results of EXTRACT and *SummaRuNNer* are not strictly comparable (also note that LEAD3 results are different in Table 1). Nevertheless, EXTRACT exceeds LEAD3 by $+0.75$ ROUGE-2 points and $+0.57$ in terms of ROUGE-L, while *SummaRuNNer* exceeds LEAD3 by $+0.50$ ROUGE-2 points and is worse by -0.20 points in terms of ROUGE-L. We thus conclude that EXTRACT is better when evaluated with ROUGE-2 and ROUGE-L. EXTRACT outperforms all abstractive models except for *abstract-RL*. ROUGE-2 is lower for *abstract-RL* which is more competitive when evaluated against ROUGE-1 and ROUGE-L.

Our latent variable model (LATENT; Section 2.3) outperforms EXTRACT, despite being a strong baseline, which indicates that training with a loss directly based on gold summaries is useful. Differences among LEAD3, EXTRACT, and LATENT are all significant with a 0.95 confidence interval (estimated with the ROUGE script). Interestingly, when applying the compression model from Section 2.2 to the output of our latent model (LATENT+COMPRESS), performance drops considerably. This may be because the compression model is a sentence level model and it removes phrases that are important for creating the document-level summaries.

4 Conclusions

We proposed a latent variable extractive summarization model which leverages human summaries directly with the help of a sentence compression model. Experimental results show that the proposed model can indeed improve over a strong extractive model while application of the compression model to the output of our extractive system leads to inferior output. In the future, we plan to explore ways to train compression models tailored to our summarization task.

Acknowledgments

We thank the EMNLP reviewers for their valuable feedback and Qingyu Zhou for preprocessing the CNN/Dailymail dataset. We gratefully acknowledge the financial support of the European Research Council (award number 681760; Lapata).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *In Proceedings of the 3rd International Conference on Learning Representations*, San Diego, California.

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336. ACM.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1662–1675, New Orleans, Louisiana.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany.

John M Conroy and Dianne P O'leary. 2001. Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407. ACM.

Elena Filatova and Vasileios Hatzivassiloglou. 2004. Event-based extractive summarization. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 104–111, Barcelona, Spain.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701. Curran Associates, Inc.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1681–1691, Beijing, China.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *In Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 2741–2749, Phoenix, Arizona.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain.

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920, Los Angeles, California.

Inderjeet Mani. 2001. *Automatic Summarization*. John Benjamins Pub Co.

Rada Mihalcea. 2005. Language independent extractive summarization. In *Proceedings of the ACL Interactive poster and demonstration sessions*, pages 49–52, Ann Arbor, Michigan.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *In Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 3075–3091, San Francisco, California.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759, New Orleans, Louisiana.

Ani Nenkova and Kathleen McKeown. 2011. Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2–3):103–233.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1310–1318, Atlanta, Georgia.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

MarcAurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proceedings of the 4th International Conference on Learning Representations*, San Juan, Puerto Rico.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112. urran Associates, Inc.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Kristian Woodsend and Mirella Lapata. 2010. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574, Uppsala, Sweden.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.