BoWLer: A neural approach to extractive text summarization

by

Pranav Ashok Dhakras, Manish Shrivastava

in

32nd Pacific Asia Conference on Language, Information and Computation (PACLIC 32)
(*PACLIC-2018*)

The Hong Kong Polytechnic University, Hong Kong SA

Report No: IIIT/TR/2018/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
December 2018

BoWLer: A neural approach to extractive text summarization

Pranav Dhakras **Manish Shrivastava**
Language Technologies Research Center, Kohli Center On Intelligent Systems,
International Institute of Information Technology, Hyderabad, India
pranav.dhakras@research.iiit.ac.in, m.shrivastava@iiit.ac.in

Abstract

While extractive summarization is a well studied problem, it is far from solved. In recent years a large number of interesting and complex models have been used to achieve significant improvements in performance. This can easily be attributed to Deep Learning models and dense vector representations but the performance gain comes with the cost of computational and representational complexity.

In this work, we present a simple, yet effective approach for extractive summarization of news articles. In line with many recent works in this area we propose an encoder-decoder architecture with a simple bag of word encoder for sentences followed by an attention based decoder for relevant sentence selection. Our model is trained end-to-end and its performance is comparable to the state-of-the-art models while being simpler both in terms of the number of parameters (significantly lesser) as well as the representational complexity.

1 Introduction

Single document summarization (SDS) aims to summarize a single text document and present the user with the most important aspects of the same. Further, SDS can be performed in two ways, *abstractive* and *extractive* where the former method is aimed at producing denser summaries which appear to be written by a human while the latter picks out key sentences, phrases and words from the original document and stitches them together to form a coherent summary. Although abstractive summarization naturally appears to be more desired of the two, the current abstractive summarization systems are riddled with problems including limited vocabulary, lack of fluency and most importantly the inability to consistently reproduce facts in the same light as in the original document. Extractive summarization on the other hand tends to be less condensed and may not always produce perfectly coherent summaries but produces fluent sentences and presents facts as is due to its property of picking out sentences from the source document.

In recent years, several neural summarization systems have been proposed. Notably, (Cheng and Lapata, 2016; Nallapati et al., 2017; Singh et al., 2017; Narayan et al., 2018) present neural extractive summarization systems based on encoder-decoder architecture. (Nallapati et al., 2016), (See et al., 2017) and (Paulus et al., 2017) all make modifications to sequence-to-sequence models and show promising results for abstractive summarization.

We present, **B**ag **O**f **W**ord embeddings **L**earn**ER**(BOWLER), a simple neural network based approach for extractive summarization. Our approach is completely data driven and trained end-to-end. We conduct experiments on the non-anonymized version of CNN-DailyMail dataset that was originally developed for reading comprehension tasks by Hermann et al. (2015) and later modified to suit extractive and abstractive summarization. The model focuses on the simplicity of representation and moves away from complex compositional representations for input sentences. The intuition is that the semantic content needed for a good extractive summary need not come from a very complex compositional architecture.

Our experiments show that our approach, using a simple Bag of Word Embeddings(BoWE) encoder, achieves near current state-of-the-art performance for extractive summarization while using fewer components, fewer parameters and reducing the training complexity.

The organization of the paper is as follows. Section 2 gives an overview of related work. Section 3 describes the approach we have taken while Section 4 explains the experimental setup and implementation details. Section 5 presents the results and comparison with other systems and Section 6 compares our model's complexity to other recent approaches. Section 7 presents our observations regarding the shortcomings in the model and the scope for further improvements.

2 Related Work

In the recent years, neural networks have shown promising results for several NLP tasks and the advancements in the field of word embeddings have further increased the interest of researchers in applying neural network techniques for summarization. Several recent works propose modifications to sequence-to-sequence (Sutskever et al., 2014) model with attention mechanism to tackle the task of abstractive summarization.

Specifically, Nallapati et al. (2016) make use of hierarchical LSTMs and large vocabulary trick (Jean et al., 2014) in their approach. See et al. (2017) introduce the concepts of coverage —to avoid redundancy and pointer networks (Vinyals et al., 2015) —to handle OOVs in abstractive summarization. Paulus et al. (2017) incorporate intra-attention that attends over the input and continuously generated output separately as well as implement a new training method boosted by reinforcement learning (RL).

While the above mentioned systems focus on abstractive summarization our work is better related to the following extractive summarization systems.

Cheng and Lapata (2016) model extractive summarization as a sequence labeling problem with an encoder-decoder architecture. They use a CNN-LSTM encoder to transform word embeddings into a document representation with hidden layers representing the sentences. A further LSTM layer along with soft attention on sentences does the task of labeling the important sentences and selecting the most relevant sentences.

Singh et al. (2017) build over this approach by adding a two-hop Memory Network (Weston et al., 2014) in their model so as to enhance the document representation.

Nallapati et al. (2017) use a hierarchical encoder comprising of word and sentence level RNNs as well as incorporate an abstractive co-training mechanism to determine which sentences from a document should be included in the extractive summary while also taking into consideration features like sentence position, content, and saliency.

Recently, Narayan et al. (2018) conceptualize extractive summarization as a summary ranking task. Their model resembles that of Cheng and Lapata (2016) but additionally incorporates a training algorithm using reinforcement learning to rank multiple system generated summaries and selecting the best one.

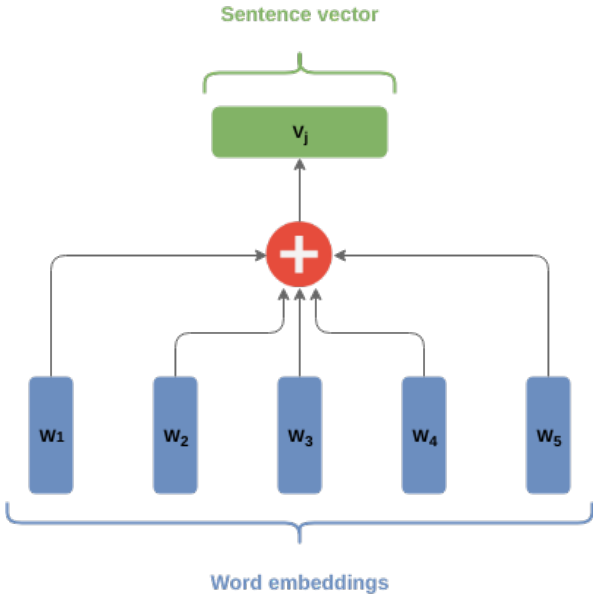
3 Approach

Similar to several recent works, we conceptualize the task of extractive summarization as that of sequence labeling wherein given a document D with sentences (s_1, s_2, ..., s_n) we want to predict labels (y_1, y_2, ..., y_n) such that $y_i \in \{0, 1\}$ (where 0 means do not select while 1 means select the sentence as candidate to be included in the summary). Further, we generate an extractive summary S by selecting k ($< n$) top scored sentences from document D such that their labels y_j are 1.

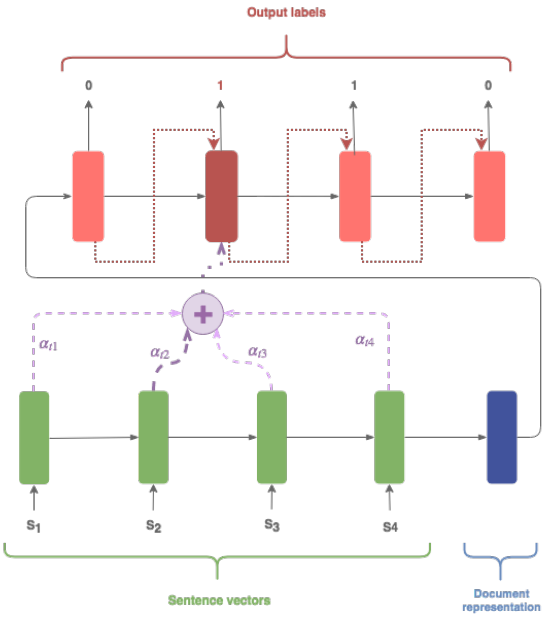
Cheng and Lapata (2016; Singh et al. (2017; Narayan et al. (2018) base their models on encoder-decoder architecture and employ hierarchical document encoders with CNNs to encode word embeddings into a sentence vectors and a LSTM based sentence to document encoder.

While our approach is inspired by these previous works, we show that using a simple sentence encoder based on Bag of Word Embeddings(BoWE) can produce comparable results to the approaches mentioned above which use more complex sentence encoders using CNNs and Memory Networks and thus reducing the model complexity as well as model size.

Our model follows an encoder-decoder approach



(a) Model architecture depicting Bag of Word Embeddings sentence encoder.



(b) Model architecture depicting document encoder and attention based summary generator.

Figure 1: Architecture of model.

having three main components, sentence encoder, document encoder and summary generator which we describe in the subsequent subsections.

3.1 Sentence Encoder

The core component of our model is a simple Bag of Word Embeddings(BoWE) sentence encoder which encodes sentences into continuous representations, that is, we consider a sentence vector as the mean of its constituent tokens' word embeddings. The word embeddings used are pre-trained and out-of-vocabulary words are replaced by *"unknown"* token while computing a sentence vector. Formally, given a document D containing sentences $(s_1, s_2, ..., s_n)$ where each sentence contains words $(w_1, w_2, ..., w_{il_i})$ and has lengths $(l_1, l_2, ..., l_n)$, we compute the corresponding sentence vectors $(v_1, v_2, ..., v_n)$ as follows,

$$v_i = \frac{1}{l_i} \sum_{j=1}^{l_i} w_{ij} \tag{1}$$

where, w_{ij} is the pre-trained word embedding for j^{th} token in the sentence s_i.

The word embedding w_x of a token t_x is given as follows,

$$w_x = \begin{cases} E_{t_x}, & \text{if } t_x \in V \\ E_{<unk>}, & \text{otherwise} \end{cases} \tag{2}$$

where, V is a vocabulary of limited size, E is the embedding matrix of word embeddings and $< unk > \in \{V, < unk >\}$ is a special *unknown* token dedicated for out-of-vocabulary(*OOVs*) tokens. Figure 1 shows the sentence encoder as described above.

3.2 Document Encoder

The document encoder composes a sequence of sentences to obtain a document representation. Given a document D consisting of a sequence of sentences $(s_1, s_2, ..., s_n)$ we first encode each of them to a continuous vector space $(v_1, v_2, ..., v_n)$ using the above mentioned sentence encoder. We process this sequence of vectors using a bidirectional recurrent neural network with Gated Recurrent Unit (Cho et al., 2014) (BiGRU) cells to avoid the vanishing gradient problem when training long documents. The

GRU equations are formulated as shown below,

$$e_t^f = GRU(v_t^{\rightarrow}, e_{t-1}) \tag{3}$$

$$e_t^b = rev(GRU(rev(v_t^{\rightarrow}), e_{t-1})) \tag{4}$$

where e_t is the real-valued hidden-state vector at time step t, v_t is the corresponding sentence input vector and $rev()$ is the reverse function. Consequently, for a document D with m sentences, the bidirectional document encoder's final hidden state, which is the document representation e, is given by,

$$e = \left[e_m^f, e_m^b\right] \tag{5}$$

where e_m^f and e_m^b are the final hidden states of the forward and backward sentence-level GRU-RNNs respectively, [] represents vector concatenation.

3.3 Summary Generator

Our summary generator component is two phased, classifying sentences as worthy of being included in the summary followed by selecting certain sentences from this pool of candidates to generate the final summary. The following subsections describe these two phases.

3.3.1 Sentence classification

Our sentence selector sequentially labels each sentence in a document with '1' (possible candidate for the summary) or '0' (otherwise). It is implemented with an attention based GRU-RNN and performs the binary classification using a softmax layer. This architecture resembles the attention based decoder devised by (Luong et al., 2015) with the only difference being the size of output vocabulary which in our case is limited to 2, namely, '0' and '1'.

The working of the sentence selector component is explained by the following equations,

$$h_t = GRU(emb(o_{t-1}), h_{t-1}) \tag{6}$$

$$\alpha_{(t,i)} = \frac{exp(h_t e_j)}{\sum_j exp(h_t e_j)} \tag{7}$$

$$c_t = \sum\nolimits_i \alpha_{(t,i)} e_i \tag{8}$$

$$x_t = [c_t, h_t] \tag{9}$$

$$h_t^{'} = tanh(W_p x_t + b_p) \tag{10}$$

$$p(y_t|s_t, D, \theta) = softmax(h_t^{'}) \tag{11}$$

$$o_t = \text{argmax}(y_t) \tag{12}$$

where, for a document D at a given time step t, o_{t-1} is the predicted label for the previous sentence, h_t is the sentence selector's hidden state, $(e_1, e_2, ..., e_{l_d})$ are the hidden states of the document encoder Bi-GRU, α_{ti} are the attention weights, c_t is the mixed context vector, [] is the concatenation operation, $h_t^{'}$ is the projected hidden state, θ are the learned model parameters, $p(y_t|s_t, D, \theta)$ are the probability scores for labels $0, 1$ as estimated by the model and $o_t \in \{0, 1\}$ is predicted label for the sentence. $emb()$ is the target side embedding matrix which converts sentence labels in real-valued continuous vectors.

In other words, at a given time t , the sentence selector reads sentence s_t from document D and predicts how worthy is the sentence for inclusion in the summary S, conditioned on the document representation (obtained from the document encoder) and the label predictions made for the previous sentences in the document.

3.3.2 Sentence selection

The sentence classification module does not by itself differentiate between the importance of two sentences labeled as '1'. In a document D comprised of n sentences we evaluate the importance of the sentences by assigning them scores $(\gamma_1, \gamma_2, ..., \gamma_n)$ is given as,

$$\gamma_t = \begin{cases} p(o_t = 1|s_t, D, \theta), & \text{if } o_t = 1 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where $p(o_t|s_t, D, \theta)$ is the confidence score of a sentence s_t being selected as a possible candidate sentence for the extractive summary and θ are the model parameters. This enables the sentence selector to identify most relevant sentences from the document.

Since we may wish to limit the length of the generated summary we apply two selection strategies. In the first strategy, we select the first k from the sentences labeled as '1' in the order of their appearance in the document. We call this approach BOWLER-First. In the second strategy, we rank the sentences (labeled as '1') and choose only the top k sentences (sorted according on γ_t) from the document. We call the second strategy BOWLER-Top

For both strategies, we insert the selected sentences into the summary one by one in the same order that they appear in the original document so as to maintain the flow and coherence of the predicted summary.

4 Experimental setup

We conduct our experiments on a modified version CNN-DailyMail dataset which was made available by Hermann (2015). We use the non-anonymized version of CNN-DailyMail dataset as released by (Cheng and Lapata, 2016)[1] Table 1 presents some key statistics about the dataset.

As a first preprocessing step we trim all documents to a length of 50 sentences, well above the average of 26 sentences per document. The version of dataset that we used has extractive labels (0, 1, 2) for all sentences in a document where 0 means sentence should not be included in the extractive summary, 1 means sentence should be included and 2 means that the sentence may be included. During training phase, we treat label 2 as we treat label 1, believing that these sentences also contain information relevant enough to be included in summary.

To evaluate the system, we report ROUGE-F scores on 3 sentence system summaries as compared against the abstractive summaries available in the dataset. We choose to use 3 sentence summaries because the average number of sentences in abstractive summaries is between 3 to 4.

	Train	Dev	Test
# of samples	277551	13367	11439
# of sentences in document	27.23	25.71	26
# of sentences in summary	3.74	4.11	3.88

Table 1: Statistics of CNN-DailyMail dataset

4.1 Implementation details

We use 100 dimensional GloVe (Pennington et al., 2014)[2] and we use full vocabulary of 400K uncased tokens which is significantly larger than 150K limit as set by (Nallapati et al., 2017). Thus, our sentence representation is 100 dimensional. We set the hidden sizes of document encoder and sentence selector to be 100 and 200 respectively. The document encoder and sentence selector components use BiGRU-RNN and GRU-RNN respectively with 2 stacked layers each. We train our model for 20 epochs using Adam (Kingma and Ba, 2014) optimizer with a batch size of 64 and a learning rate α of 0.001 and clip the gradients in the network to have a maximum norm of 5. All model parameters were randomly initialized over a uniform distribution within $[-0.08, 0.08]$. We use dropout regularization with probability of 0.3 on all GRU layers. Additional, to help boost the training in the initial epochs we make use of teacher forcing algorithm (1989) with a probability of 0.5.

5 Systems comparison

Table 2 shows the ROUGE-F scores, on full length summaries on the entire CNN-DailyMail test set. As mentioned in Section 3.3, we present two approaches, BOWLER-Top and BOWLER-First, that vary only in their sentence selection strategies (that is, the underlying learned parameters are remain the same). We also implement Lead-3 baseline and compare our systems against this baseline as well as several recent approaches (both abstractive and extractive).

We observe that our best model (BOWLER-First) outperforms or gives comparable results to other extractive models. It is worth noting that our models use a Bag of Word Embeddings(BoWE) sentence encoder which ignores the word order but still produces better results than the system proposed by Cheng and Lapata (2016) which also poses extractive summarization as a sequence labeling task. This is highlighted by the fact that Cheng and Lapata (2016) use a CNN based sentence encoder while we use a BoWE sentence encoder. Similarly, BOWLER-First outperforms SummaRunner (Nallapati et al., 2017) which makes use of LSTMs to encode both sentences as well as documents. REFRESH (Narayan et al., 2018), the current state-of-the-art extractive summarizer, uses an architecture which resembles that of Cheng and Lapata (2016) and augments it with the use of Reinforce-

[1]The the data is publicly available *here*.

[2]GloVe vectors are available *here*.

System	Type	CNN			DailyMail			CNN-DailyMail		
		R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Lead-3	ext	30.2	11.6	26.2	**41.4**	18.1	37.6	40.4	17.5	36.6
SummaRunner (2017)$^+$	ext	—	—	—	—	—	—	39.6	16.2	35.3
(Cheng and Lapata, 2016)*	ext	28.4	10	25	36.2	15.2	32.9	35.5	14.7	32.2
REFRESH (2018)	ext	**30.4**	**11.7**	**26.9**	41.0	**18.8**	**37.7**	40.0	**18.2**	36.6
BOWLER-Top (ours)	ext	29.8	11.4	26.3	39.0	17.0	35.2	38.2	16.6	34.4
BOWLER-First (ours)	ext	30.2	11.6	26.8	41.3	18.0	37.4	40.3	17.5	36.5
(Paulus et al., 2017)	abs	—	—	—	—	—	—	**41.2**	15.8	**39.1**
POINT-GEN-COV (2017)	abs	—	—	—	—	—	—	39.5	17.3	36.4
(Nallapati et al., 2016)$^+$	abs	—	—	—	—	—	—	35.46	13.30	32.65

Table 2: Quantitative analysis using ROUGE-F scores on full length summaries for entire CNN-DailyMail dataset. $^+$ use anonymized version of dataset and thus not strictly comparable. * numbers as reported by Narayan et al. (2018) on full length summaries.

System	CNN-DailyMail		
	R-1	R-2	R-L
CHEAT (k=2)	**47.8**	**26.2**	**43.5**
CHEAT (k=3)	44.2	24.2	40.6
BOWLER-Top (ours)	38.2	16.6	34.4
BOWLER-First (ours)	40.3	17.5	36.5

Table 3: Comparison between our model and CHEAT—an estimated upper limit for extractive summarization.

ment Learning to rank summaries and optimizes for ROUGE score. BOWLER-First produces comparable results to theirs.

It seems that the semantic content needed for good quality summary can be captured solely on the basis of good word representations without the need of a complex compositional model. However, several experiments involving various model architectures and word embeddings would be required to really establish this hypothesis. Thus, we leave it as an interesting avenue for future works.

We also compare our extractive system against recent abstractive summarization systems and observe that our model outperforms models of Nallapati et al. (2016) and See et al. (2017) but does not quite match the model proposed by Paulus et al. (2017).

Additionally, we wanted to estimate the best possible extractive summaries for the CNN-DailyMail dataset. For this purpose we created a system called CHEAT which has access to both the document and the gold abstractive summary. Given this information, CHEAT computes the ROUGE scores for each sentences in the document against the ground truth and selects the top k sentences according to their respective ROUGE scores.[3] Liu et al. (2018) also include a similar cheating system in their analysis. Table 3 shows a comparison between our model and CHEAT indicating that there is scope for significant improvements in extractive summarization.

	# of parameters
BOWLER	867005
SummaRunner (2017)	1571201
Cheng and Lapata (2016)	25727003

Table 4: Comparison of number of network parameters.

6 Model complexity

Several of the approaches discussed in Section 2 pose extractive summarization as a sequence labeling task and describe architectures largely based on CNN-LSTM (or hierarchical LSTM) encoder and LSTM decoder with some additional components or novel training methods. While our approach is inspired by these previous works, it is simpler in two ways, firstly, it removes the complex word to sen-

[3]We tried three values (2, 3, 4) for k and found k=2 to work best on the test data.

System	CNN-DailyMail								
	ROUGE-1			ROUGE-2			ROUGE-L		
	R	P	F-1	R	P	F-1	R	P	F-1
Lead-3	54.3	34.0	**40.4**	23.7	14.7	**17.5**	49.2	30.8	**36.6**
BOWLER-Top (ours)	**57.4**	30.5	38.2	**25.0**	13.2	16.6	**51.5**	27.3	34.4
BOWLER-First (ours)	54.4	33.8	40.3	23.7	14.6	**17.5**	49.3	30.7	36.5
POINT-GEN-COV (2017)	43.0	**39.1**	39.5	18.8	**17.1**	17.2	39.6	**36.0**	36.4

Table 5: Variations between models on the basis of precision, recall and F-1 scores.

tence encoder components and secondly, it has fewer network parameters.[4] Table 4 shows this comparison. We can observe that our model takes half as many parameters as SummaRunner (2017), mainly because our model removes the word level LSTM used by Nallapati et al. (2017). Similarly, (Cheng and Lapata, 2016)'s model has almost 30 times as many parameters as ours largely due to their use of CNN based sentence encoder. Although we were unable to run the REFRESH system, we can estimate the number of parameters in the system to be in same range of Cheng and Lapata (2016) since REFRESH is an extension over Cheng and Lapata (2016). Clearly, our model is significantly smaller as well as less complex as compared to other relevant works on extractive summarization and yet performs on par with all these systems.

7 Analysis & Observations

Since See et al. (2017) have made the outputs of their system publicly available, we also compare our system with theirs on ROUGE-Recall, ROUGE-Precision and ROUGE-F scores as demonstrated by Table 5. We observe that extractive approaches such as Lead-3 baseline and BOWLER outperform abstractive counterparts like POINT-GEN-COV (2017) on ROUGE-Recall by a large margin but suffer due to their verbosity when compared on ROUGE-Precision scores. From Table 2 we observe that there is a significant difference in the performance of our two approaches which differ in only their sentence selection strategies(model parameters remain the same). We investigate the difference between two approaches by analyzing their respective sentence distributions as seen in Figures 2 and 3.

[4]This excludes input word embedding parameters since both models use pre-trained embeddings.

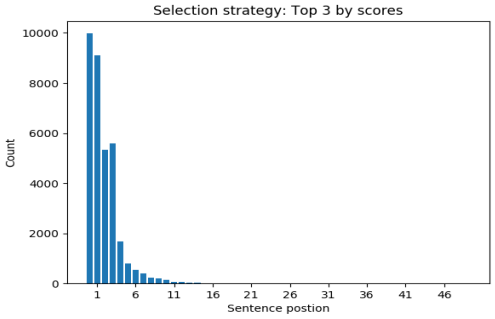


Figure 2: Position-wise sentence distribution for BOWLER-Top

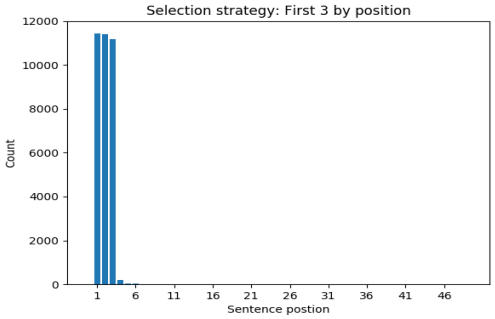


Figure 3: Position-wise sentence distribution for BOWLER-First

For BOWLER-First, we observe that the selection strategy selects the first 3 sentences for a very large number of documents. At the first glance, it appears suspiciously similar to Lead-3 approach but our model is backed by the position-wise confidence scores seen in Figure 4. From analysis of BOWLER-Top, we observe that the frequency of a sentence being selected for summary roughly decreases as the sentence position is increased . This is supported by Figure 4 which shows the average confidence scores

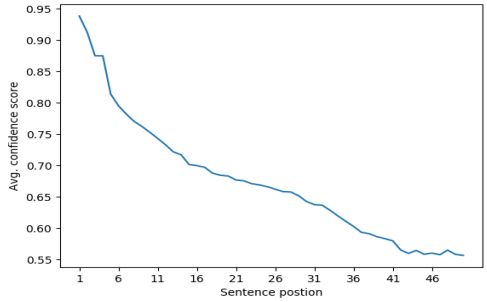


Figure 4: Analysis of position-wise confidence scores as assigned to sentences by BOWLER-Top.

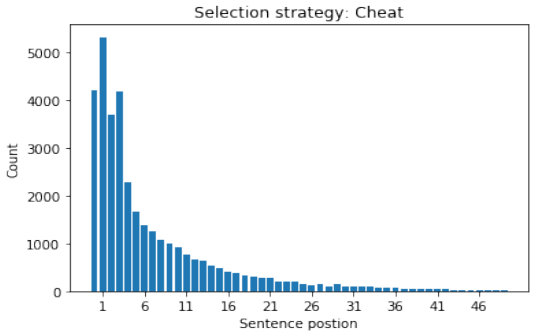


Figure 5: Position-wise sentence distribution for CHEAT—an estimated upper limit for extractive summarization

for each sentence position across all documents.

In Table 5 the ROUGE-R scores for BOWLER-Top are consistently higher than that of other systems indicating that the sentences selected by BOWLER-Top are indeed more informative. However, the ROUGE-P for BOWLER-Top, which is the least among all, affects its overall performance negatively. The lower ROUGE-P score is attributed to the significantly longer length of the summaries produced by BOWLER-Top (579 bytes) as compared to BOWLER-First (481 bytes).

From the higher ROUGE-R scores, we can infer that our extractive model is good at finding the most relevant sentences in a document whereas the lower ROUGE-P scores indicate that the next line of work would naturally be to increase the brevity of the outputs.

Figure 2 shows that the our model has a bias in selecting sentences from earlier sections of a document. To verify this we also present in Figure 5 the position-wise sentence distribution for CHEAT which by definition should be close to the true position-wise sentence distribution.

Comparing Figures 2 and 5 one can see that our model is indeed strongly biased towards the start of the document but it also shows that the reason for this bias is partially due to a similar, albeit, weaker bias in the true distribution. We identify that techniques to counter this bias would indeed help in increasing the performance of extractive summarization systems.

8 Conclusion

In this work, we present a simple and effective neural sequence labeling model for extractive summarization for news domain and show that its performance is close to current state-of-the-art extractive summarization system. Our simple BoWE based approach is comparable to the state of the art for extractive summarization suggesting that complex compositional representations may be an overkill for summarization in news domain. Further work could be done to explore this aspect as well as study how different types of word embeddings affect the the system performance.

We also establish a rough upper limit for extractive summarization techniques on CNN-DailyMail dataset and observe that there is scope for significant improvements to extractive summarization systems.

From our comparison against recent abstractive summarization systems we observe that current extractive systems outperform these abstractive system when trying to select the most relevant information in a document (Table 5). As a direction for future work we propose the use of hybrid extractive-abstractive models which first pick the relevant sentences and then use linguistic rules and techniques to drop the least significant words and/or phrases to generate more condensed summaries. It would also be worth exploring different ranking mechanisms for better selection of candidate sentences.

Acknowledgments

We would like to thank Pruthwik Mishra, Saumitra Yadav, Prathyusha Danda and Shweta Ghaisas for their valuable and timely inputs.

References

[Cheng and Lapata2016] Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*.

[Cho et al.2014] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

[Hermann et al.2015] 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.

[Jean et al.2014] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2014. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*.

[Kingma and Ba2014] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Liu et al.2018] Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *CoRR*, abs/1801.10198.

[Luong et al.2015] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025.

[Nallapati et al.2016] 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*.

[Nallapati et al.2017] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081.

[Narayan et al.2018] Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636*.

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

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[See et al.2017] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

[Singh et al.2017] Abhishek Kumar Singh, Manish Gupta, and Vasudeva Varma. 2017. Hybrid memnet for extractive summarization. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 2303–2306. ACM.

[Sutskever et al.2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.

[Vinyals et al.2015] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

[Weston et al.2014] Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *CoRR*, abs/1410.3916.

[Williams and Zipser1989] R. J. Williams and D. Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, June.