# Neural Query-Biased Abstractive Summarization Using Copying Mechanism

Tatsuya Ishigaki[1(✉)], Hen-Hsen Huang[3], Hiroya Takamura[1,4], Hsin-Hsi Chen[2], and Manabu Okumura[1]

[1] Tokyo Institute of Technology, Tokyo, Japan
ishigaki@lr.pi.titech.ac.jp, {takamura,oku}@pi.titech.ac.jp
[2] National Taiwan University, Taipei, Taiwan
hhchen@ntu.edu.tw
[3] National Chengchi University, Taipei, Taiwan
hhhuang@nccu.edu.tw
[4] AIST, Tokyo, Japan

**Abstract.** This paper deals with the query-biased summarization task. Conventional non-neural network-based approaches have achieved better performance by primarily including the words overlapping between the source and the query in the summary. However, recurrent neural network (RNN)-based approaches do not explicitly model this phenomenon. Therefore, we model an RNN-based query-biased summarizer to primarily include the overlapping words in the summary, using a copying mechanism. Experimental results, in terms of both automatic evaluation with ROUGE and manual evaluation, show that the strategy to include the overlapping words also works well for neural query-biased summarizers.

**Keywords:** Abstractive summarization · Query-biased summarization

## 1 Introduction

A query-biased summarizer takes a query in addition to a source document as an input, and outputs a summary with respect to the query, as in Table 1. The generated summaries are intended to be used, for example, for snippets as the results of search engines. Query-biased summarization has been studied for decades [3,4,16,18]. Conventional approaches are mostly extractive, and often use the overlapping words as cues to calculate the salience score of a sentence [14,16,18].

On the other hand, recurrent neural network (RNN)-based approaches have enabled summarizers to generate fluent abstractive summaries [1,7,13], but do not explicitly model the strategy to primarily include the overlapping words. In this paper, therefore, we incorporate this strategy into RNN-based summarizers using copying mechanisms.


J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12036, pp. 174–181, 2020.
https://doi.org/10.1007/978-3-030-45442-5_22

**Table 1.** Example of a source document, a query, the gold summary. The words overlapping between the source and query are shown in **bold**.

| | |
|---|---|
| **Source:** | **Vigilanteism** simply causes more problems and will not fix the original problem. one should restore law and order rather than implementing disorder |
| **Query:** | Will **vigilanteism** restore law and order? |
| **Gold Summary:** | **Vigilanteism** merely instigates chaos |

A copying mechanism is a network to primarily include the words in the source document in the summary [5,6,17]. To achieve this, the copying mechanism increases the probability of including the words in the source document. A copying mechanism can be seen as an extension of the *pointer-network* [19], which only copies words in the input and does not output words other than in the input. Gu et al. [5], Gulcehre et al. [6], and Miao et al. [12] extended the pointer-network to copying mechanisms by using a function to balance *copying* and *generation*. See et al. [17] and Chen and Lapata [2] applied the copying mechanism to single-document summarization tasks without a query. We came up with an idea of using copying mechanisms to include the overlapping words in the summary. However, the copying mechanisms were originally designed for the settings without the query information, and it is not necessarily clear how we can integrate the mechanisms into a query-biased summarizer.

Encoder-decoders for the query-biased setting have been proposed. Hasselqvist et al. [7] proposed an architecture being able to copy the words in the source document, while our copying mechanisms copy the overlapping words and their surroundings explicitly. Nema et al. [13] presented a dataset extracted from Debatepedia. They proposed a method to gain the diversity of the summary, while we focus on copying mechanisms.

We propose three copying mechanisms designed for query-biased summarizers: copying from the source, copying the overlapping words, and copying the overlapping words and their surroundings. We empirically show that the models copying the overlapping words perform better. These results support the fact that the strategy to include the overlapping words, which was shown useful for conventional query-biased summarizers, also works well for neural network-based query-biased summarizers.

## 2 Base Model

We first explain a base query-biased neural abstractive summarizer proposed by Nema et al. [13], into which we integrate our copying mechanisms in the next section.

**Encoders:** The base model has two bi-directional Long Short-term Memory (LSTM) [8]-based encoders; one is for the query $\mathbf{q} = \{q_1, ..., q_{|\mathbf{q}|}\}$ and another is for the source document $\mathbf{d} = \{w_1, ..., w_{|\mathbf{d}|}\}$. In each encoder, the outputs of the forward and the backward LSTM are concatenated into a vector. We refer

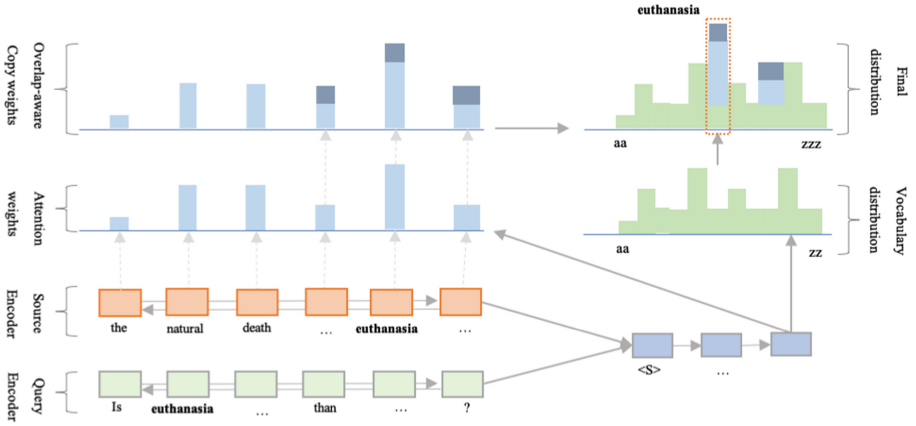


**Fig. 1.** Overview of a query-biased summarizer with a copying mechanism.

to the generated vector for the $i$-th word in the query as $h_i^q$, and the $j$-th word in the source document as $h_j^d$.

**Decoder with Query- and Source Document- Attentions:** The decoder outputs the summary. The final state $h_{|\mathbf{d}|}^d$ is used to initialize the first state of the LSTM in the decoder. In each time step $t$, the decoder calculates the attention weights $a_{t,i}^q = v_q \cdot \tanh(W_q s_t + U_q h_i^q)$ for every word in the query. $W_q, U_q \in \mathbb{R}^{l \times l}$ are weight matrices, and $v_q \in \mathbb{R}^l$ is a $l-$dimensional weight vector, where each element is automatically learned from the training data. $s_t = LSTM_d(s_{t-1}, [y_{t-1}; d_{t-1}])$ is the output of the LSTM in the decoder. Here, $y_{t-1}$ is the embedding vector of the previously generated word and $d_t$ is a document representation explained later. The weights are converted into probabilities $\alpha_{t,i}^q = \frac{\exp(a_{t,i}^q)}{\sum_{i=1}^{|\mathbf{q}|} \exp(a_{t,i}^q)}$. We now obtain a query vector: $q_t = \sum_{i=1}^{|\mathbf{q}|} \alpha_{t,i}^q h_i^q$.

The source document attention mechanism further calculates the attention weights $a_{t,j}^d$ for every word in the source document and converts them into probabilities $\alpha_{t,j}^d$:

$$a_{t,j}^d = v_d \cdot \tanh(W_d s_t + U_d h_j^d + Z q_t),$$
$$\alpha_{t,j}^d = \frac{\exp(a_{t,j}^d)}{\sum_{j=1}^{|\mathbf{w}|} \exp(a_{t,j}^d)}. \quad (1)$$

$W_d, U_d, Z \in \mathbb{R}^{l \times l}$ and $v_d \in \mathbb{R}^l$ are learnable parameters. Note that Eq. (1) contains $q_t$, which means that the weights are calculated by considering the query. We then take the weighted average to obtain a document vector: $d_t = \sum_{j=1}^{|\mathbf{w}|} \alpha_{t,j}^d h_j^d$.

Finally, the score of generating the word $n$ in the pre-defined dictionary $N$ is calculated as $a_{t,gen}(n) = \delta_n \cdot W_o(W_{dec} s_t + V_{dec} d_t)$. The scores are converted into a probability distribution:

$$p_{t,gen}(n) = \frac{\exp(a_{t,gen}(n))}{\sum_{m=1}^{|N|} \exp(a_{t,gen}(n_m))}, \tag{2}$$

where $W_o \in \mathbb{R}^{l \times N}$, $W_{dec} \in \mathbb{R}^{l \times l}$ and $V_{dec} \in \mathbb{R}^{l \times l}$ are learnable parameter matrices. $\delta_n \in \{0,1\}^{|N|}$ is a one-hot vector where the element corresponding to the word $n$ is 1, or 0 otherwise. Thus, the dot product of $\delta_n$ and $W_o(W_{dec}s_t + V_{dec}d_t)$ calculates the score of generating the word $n$. Equation (2) converts the score into a probability distribution.

**Objective Function:** All learnable matrices are tuned to minimize the negative likelihood for the reference summaries $y$ in the training data $D$: $-\frac{1}{|D|}\sum_D log\ p(x|y)$.

## 3 Copying Mechanisms for Query-Biased Summarizers

We discuss the copying mechanisms for query-biased summarizers. Figure 1 shows the overview of a query-biased summarizer with a copying mechanism. In the following subsections, we present three mechanisms; `SOURCE`, `OVERLAP`and `OVERLAP-WIND`.

`SOURCE`: We explain `SOURCE`, which copies the words from the source document. The strategy is a straightforward extension of the existing copying mechanisms [5,6,17]. The neural query-biased summarizer proposed by Hasselqvist et al. [7] also adopted this strategy, but they did not report its impact. We further extend this mechanism in the following subsections. In this strategy, the output layer calculates the probability distribution over the set $N \cup M = \{n_1, ..., n_{|N|+|M|}\}$, where $N$ is the set of words in the pre-defined dictionary and $M$ is the set of words in the source document. Thus, $p_{t,gen}$ in Eq. (2) is modified to consider the extended vocabulary as follows:

$$p'_{t,gen}(n) = \begin{cases} p_{t,gen}(n) & (n \in N), \\ 0 & (n \notin N). \end{cases} \tag{3}$$

In the copying mechanism, we consider two different probabilities for the word $n$ in the vocabulary; the generation probability $p'_{t,gen}$ and the copying probability $p_{t,copy}$. The switching probability $sw_t = \sigma(z_d \cdot d_t + z_s \cdot s_t + z_y \cdot y_{t-1})$ balances those probabilities as: $p_t(n) = sw_t p'_{t,gen}(n) + (1 - sw_t)p_{t,copy}(n)$. Here, $z_d, z_s \in \mathbb{R}^l$ and $z_y \in \mathbb{R}^{l_{emb}}$. $\sigma$ represents a sigmoid function. $l_{emb}$ is the dimension size of a word embedding. $p_{t,copy}$ is calculated as follows:

$$p_{t,copy}(n) = \begin{cases} \alpha^d_{t,idx\_s(n)} & (n \in M), \\ 0 & (n \notin M). \end{cases} \tag{4}$$

$idx\_s(n)$ is a function to return the position of the word $n$ in the source document. The attention weight $\alpha^d_{t,idx\_s(n)}$ provided by the source document attention module is used as the score for outputting the word $n$ in the source document.

`OVERLAP`: We propose `OVERLAP`, the model to copy the overlapping words. This model calculates the probability distribution over the set $N \cup M = \{n_1, ..., n_{|N|+|M|}\}$ in the same way as in `SOURCE`. This model increases the scores for the overlapping words as follows:

$$a^o_{t,n} = \begin{cases} (1+\lambda_d) a^d_{t,idx\_s(n)} & (n \in Q \cap M), \\ a^d_{t,idx\_s(n)} & (n \in M \setminus (Q \cap M)), \\ 0 & (n \notin M). \end{cases} \tag{5}$$

The scores are converted into a probability distribution: $p_{t,copy}(n) = \frac{\mathtt{exp}(a^o_{t,idx(n)})}{\sum_{j=1}^{|N|+|M|} \mathtt{exp}(a^o_{t,j})}$.

In the equations above, $Q$ refers to the set of content words[1] in the query. Thus, $Q \cap M$ represents the overlapping words. $\lambda_d \in \mathbb{R} > 0$ is a hyperparameter that controls the importance of the overlapping words. By using $\lambda_d$, this model can assign a relatively high probability for the overlapping words.Thus, the overlapping words are more likely to be included in the summary. $\lambda_d$ is tuned on validation data.

`OVERLAP-WINDOW`: We finally explain `OVERLAP-WIND`, the model that copies the overlapping words and their surrounding words. We assume that the surroundings of overlapping words might also be important. This model calculates the scores for the overlapping words and their surroundings as follows:

$$a^{o\_w}_{t,idx\_s(n)} = \begin{cases} (1+\lambda_d) a^d_{t,idx\_s(n)} & (n \in M_{L_d}), \\ a^d_{t,idx\_s(n)} & (n \in M \setminus M_{L_d}), \\ 0 & (n \notin M), \end{cases} \tag{6}$$

where $M_{L_d}$ is the set that contains $L_d (\in \mathbb{N})$ words around the overlapping word in addition to the overlapping word itself. Then, the scores are converted into a probability distribution by using the `softmax` function: $p_{t,copy}(n) = \frac{\mathtt{exp}(a^{o\_w}_{t,idx\_s(n)})}{\sum_{j=1}^{|N|+|M|} \mathtt{exp}(a^{o\_w}_{t,j})}$.

## 4 Experiments

We used the publicly available dataset[2] provided by Nema et al. [13]. The data contains the tuples of a source document, a query and a summary, extracted from Debatepedia[3]. We used 80% of the data for training, and the remaining was equally split for parameter tuning and testing.

[1] We used the list of stop words defined in the nltk library for filtering to obtain content words.

[2] https://github.com/PrekshaNema25/DiverstiyBasedAttentionMechanism.

[3] http://www.debatepedia.org/en.

We used Adam [9] for the optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The initial learning rate was set to 0.0004. The word embeddings for both the query and the source document were initialized by GloVe [15] and further tuned during training the models. We selected the best-performing value from 200, 300 and 400 for the dimension size for LSTM by using the data for tuning. We used 32 for the batch size. We used all the vocabulary in the training data as the pre-defined dictionary.

We prepared three baselines without any copying mechanisms. The first, `ENC-DEC`, was a simple encoder-decoder based summarizer without the query encoder. This model uses Eq. (1) without $q_t$. The second, `ENC-DEC QUERY`, was the query-aware encoder-decoder explained in Sect. 2. The third, `DIVERSE`, was the state-of-the-art model proposed by Nema et al. [13]. We adopted full-length ROUGE [11] for the automatic evaluation metric. In addition, we conducted manual evaluation by human judges. 55 randomly selected document/query pairs and their summaries generated by `DIVERSE`, `SOURCE`, and `OVERLAP-WIND`were shown to crowdworkers on Amazon Mechanical Turk. We assigned 10 workers for each set of document/query/summaries and asked them to rank the summaries. We adopted readability and responsiveness as the manual evaluation criteria, following the evaluation metric in DUC2007[4]. The workers were allowed to give the same rank to multiple summaries.

## 5 Results

We show the ROUGE scores[5] and the averaged rankings from human judges in Table 2.

**Table 2.** The full-length ROUGE-1, ROUGE-2, ROUGE-L (higher is better) and the averaged rankings (lower is better) from human judges. The best performing model is in bold.

| | ROUGE-1 | ROUGE-2 | ROUGE-L | Readability | Responsiveness |
|---|---|---|---|---|---|
| `Reference` | – | – | – | 1.50 | 1.55 |
| `ENC-DEC` | 13.73 | 2.06 | 12.84 | – | – |
| `ENC-DEC QUERY` | 29.28 | 10.24 | 28.21 | – | – |
| `DIVERSE` [13] | 41.02 | 26.44 | 40.78 | 3.36 | 3.39 |
| `SOURCE` | 43.32 | 29.12 | 42.96 | 1.99 | 1.93 |
| `OVERLAP` | 43.47 | 29.68 | 43.26 | – | – |
| `OVERLAP-WIND` ($L_d = 1$) | **44.41**$^\dagger$ | **30.48**$^\dagger$ | **44.20**$^\dagger$ | **1.83**$^\dagger$ | **1.85**$^\dagger$ |
| `OVERLAP-WIND` ($L_d = 2$) | 43.16 | 29.15 | 42.90 | – | – |
| `OVERLAP-WIND` ($L_d = 3$) | 44.03 | 29.78 | 43.77 | – | – |

[4] https://duc.nist.gov/duc2007/tasks.html.
[5] The option for ROUGE is -a -n 2 -s.

**ROUGE Scores:** ENC-DEC, without query information, achieved very low performance. Adding the query encoder (ENC-DEC QUERY) improved the score. Furthermore, copying the words in the source document (SOURCE) achieved better scores than those of the best-performing model without a copying mechanism (DIVERSE). Thus, integrating the copying mechanism improved the performance even in the query-biased setting. Among our models, OVERLAP and OVERLAP-WIND ($L_d = 1$) achieved the better performances than SOURCE. The dagger (†) indicates that the differences between the scores of SOURCE, OVERLAP and those of our best-performing model (OVERLAP-WIND($L_d = 1$)) are statistically significant with the paired bootstrap resampling test used in Koehn et al. [10] ($p < 0.05$). This supports our assumption that the strategy to copy the overlapping words is shown effective even for RNN-based summarizers.

**Rankings by Human Judges:** Our best model OVERLAP-WIND ($L_d = 1$) is ranked higher than the state-of-the-art DIVERSE and SOURCE. The differences between OVERLAP-WIND ($L_d = 1$) and SOURCE are statistically significant ($p < 0.05$) with the paired bootstrap resampling test [10]. The results of manual evaluation also support our assumption.

## 6 Conclusion

We proposed the copying mechanisms designed for query-biased summarizers to primarily include the words overlapping between the source document and the query. Our experimental results showed that the mechanisms to primarily include the overlapping words between the source document and the query achieved the better performances in terms of both ROUGE and rankings by human judges. The results suggested that the strategy to include the overlapping words, which has been shown useful for conventional non-neural summarizers, also works well for RNN-based summarizers.

## References

1. Baumel, T., Eyal, M., Elhadad, M.: Query focused abstractive summarization: incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. arXiv preprint arXiv:1801.07704 (2018)
2. Cheng, J., Lapata, M.: Neural summarization by extracting sentences and words. In: Proceedings of The 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Berlin, Germany, pp. 484–49 (2016)
3. Dang, H.T.: Overview of DUC 2005. In: Proceedings of 2005 Document Understanding Conferences, DUC 2005, pp. 1–12. Citeseer (2005)
4. Daumé III, H., Marcu, D.: Bayesian query-focused summarization. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2006, pp. 305–312 (2006)
5. Gu, J., Lu, Z., Li, H., Li, V.O.: Incorporating copying mechanism in sequence-to-sequence learning. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016. pp. 1631–1640 (2016)

6. Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., Bengio, Y.: Pointing the unknown words. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, vol. 1, pp. 140–149 (2016)
7. Hasselqvist, J., Helmertz, N., Kågebäck, M.: Query-based abstractive summarization using neural networks. arXiv preprint arXiv:1712.06100 (2017)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
9. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of 3rd International Conference on Learning Representations, ICLR 2015 (2015)
10. Koehn, P.: Statistical significance tests for machine translation evaluation. In: Proceedings of the 2014 Conference on Empirical Methods on Natural Language Processing, EMNLP 2014, pp. 388–395 (2004)
11. Lin, C.Y.: Rouge: a package for automatic evaluation of summaries. In: Proceedings of ACL2004 Workshop, pp. 74–81 (2004)
12. Miao, Y., Blunsom, P.: Language as a latent variable: discrete generative models for sentence compression. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, pp. 319–328 (2016)
13. Nema, P., Khapra, M.M., Laha, A., Ravindran, B.: Diversity driven attention model for query-based abstractive summarization. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2017, pp. 1063–1072, July 2017
14. Otterbacher, J., Erkan, G., Radev, D.R.: Biased LexRank: passage retrieval using random walks with question-based priors. Inf. Process. Manag. **45**(1), 42–54 (2009)
15. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, pp. 1532–1543 (2014)
16. Schilder, F., Kondadadi, R.: FastSum: fast and accurate query-based multi-document summarization. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, ACL 2008, pp. 205–208 (2008)
17. See, A., Liu, P.J., Manning, C.D.: Get to the point: summarization with pointer-generator networks. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, vol. 1, pp. 1073–1083 (2017)
18. Tombros, A., Sanderson, M.: Advantages of query biased summaries in information retrieval. In: Proceedings of 21st Annual ACM/SIGIR International Conference on Research and Development in Information Retrieval, SIGIR 1998, pp. 2–10. ACM (1998)
19. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: Proceedings of Twenty-Ninth Conference on Neural Information Processing Systems, NIPS 2015, pp. 2692–2700 (2015)