Automatic Text Summarization Using Reinforcement Learning with Embedding Features

Gyoung Ho Lee, Kong Joo Lee
Information Retrieval & Knowledge Engineering Lab.,
Chungnam National University,
Daejeon, Korea
gyholee@gmail.com, kjoolee@cnu.ac.kr

Abstract

An automatic text summarization system can automatically generate a short and brief summary that contains a main concept of an original document. In this work, we explore the advantages of simple embedding features in Reinforcement leaning approach to automatic text summarization tasks. In addition, we propose a novel deep learning network for estimating Q-values used in Reinforcement learning. We evaluate our model by using ROUGE scores with DUC 2001, 2002, Wikipedia, ACL-ARC data. Evaluation results show that our model is competitive with the previous models.

1 Introduction

In this work, we present extractive text summarization for a single document based on Reinforcement leaning (RL) method. One of the advantages of the extractive approach is that a summary consists of linguistically correct sentences as long as a source document has a certain level of linguistic quality.

One of the most well-known solutions of extractive text summarization is to use maximal marginal relevance (MMR) (Goldstein et al., 2000). However, MMR cannot take into account for the quality of a whole summary because of its greediness (Ryang and Abekawa, 2012). Another solution is to use optimization techniques such as integer linear programming (ILP) to infer the scores of sentences with consideration of the quality of a whole summary (McDonald, 2007). However, these methods have a very large time complexity so they are not applicable for text summarization tasks.

A Reinforcement Learning method would be an alternative approach to optimize a score function in extractive text summarization task. Reinforcement learning is to learn how an agent in an environment behaves in order to receive optimal rewards in a given current state (Sutton, 1998).

The system learns the optimal policy that can choose a next action with the most reward value in a given state. That is to say, the system can evaluate the quality of a partial summary and determine the sentence to insert in the summary to get the most reward. It can produce a summary by inserting a sentence one by one with considering the quality of the hypothetical summary. In this work, we propose an extractive text summarization model for a single document based on a RL method.

A few researchers have proposed the RL approaches in automatic text summarization (Goldstein et al., 2000; Rioux et al. 2014; Henß et al. 2015). Previous studies mainly exploited hand-crafted complex features in RL-based automatic text summarization. However, choosing important features for a task, re-implementing the features for a new domain, and re-generating new features for a new application are very difficult and time-consuming jobs. The recent mainstream of NLP applications with Deep Learning is to reduce the burden of hand-crafted features. Embedding is one of the simplest deep learning techniques to build features that represent words, sentences, or documents. It has already shown state-of-the art performance in many NLP applications.

The main contributions of our work are as follows: first, we explore the advantages of simple embedding features in RL approach to automatic text summarization tasks. Content embeddings vector and position embeddings vector are only

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 193–197,
Taipei, Taiwan, November 27 – December 1, 2017

features that our system adopts. Second, we propose a novel deep learning network for estimating Q-values used in RL. This network is devised to consider the relevance of a candidate sentence for an entire document as well as the naturalness of a generated summary.

We evaluate our model by using ROUGE scores (Lin, 2004) and show that the performance of our model is comparable to that of the previous studies which rely on as many features as possible.

2 Related Work

As far as we know, Ryang and Abekawa (2012) have so far been the first ones who applied RL to the text summarization. The authors regard the extractive summarization task as a search problem. In their work, a state is a subset of sentences and actions are transitions from one state to the next state. They only consider the final score of the whole summary as reward and use TD(λ) as RL framework. Rioux et al. (2014) extended this approach by using TD. They employed ROUGE as part of their reward function and used bi-grams instead of tf ∗ idf as features. Henß and Mieskes (2015) introduced Q-learning to text summarization. They suggest RL-based features that describe a sentence in the context of the previously selected sentences and how adding this sentence changes hypothetical summary. Our work extends previous work using DQN based algorithm and embedding features.

3 Model for Automatic text summarization

In this work, we apply the Deep Q-Networks (DQN)-based model (Volodymyr et al. 2015) to automatic text summarization tasks. In the case of text summarization, the state denotes a summary which can still be incomplete and the action denotes the addition of a sentence to this summary. For using RL method in text summarization, there are two parameters that should be predefined. One is a length limitation of a summary. The other parameter is a reward value for a partial summary. In this work, we use the same length limitation and reward function used in Henß et al., (2015). The reward function is defined as:

$$\boldsymbol{Reward}(\text{state}_t, \text{state}_{t+1}) = \boldsymbol{Score}(\text{state}_{t+1}, \text{Hd}) - \boldsymbol{Score}(\text{State}_t, \text{Hd}) \quad (1)$$

In the above equation, *Score* measures the quality of the partial summary (state) by comparing it with the corresponding human reference summary **Hd**. We use the ROUGE-2 score for the measurement.

3.1 Q-Network

Q-learning models the value $Q(s_t; a_t)$ of performing an action a_t in the current state s_t. We use a deep neural network model with a regression function to compute the Q-values.

The model can calculate a Q-value based on a partial summary (current state) and a candidate sentence (action). The output Q-value indicates the expectation value that the agent can get when it selects the candidate sentence as a part of the summary. Figure 1 shows the Q-network model we propose in this work.

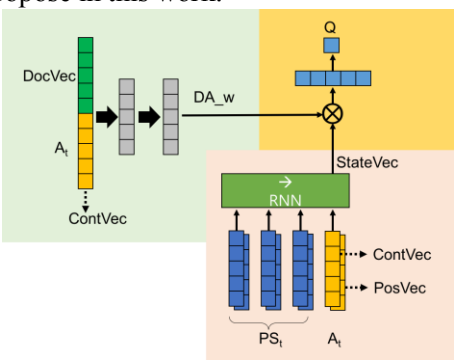

Figure 1: Architecture of our Q-network

The Q-network model consists of three modules shown in Figure 1. First module (upper left in the figure) is to represent that a semantic relationship between the input document and the candidate sentence (A_t). The input document is provided as DocVec in which the whole meaning of the document is embedded. The candidate sentence is represented as a sentence embedding vector ContVec. The vector DocVec and ContVec are the inputs to this module. The output of the module is vector DA_w, which is calculated as：

$$DA\_w = sigmoid(W_{da_{out}}, \text{hidden}_{\text{da}}) \quad (2)$$

$$hidden_{da} = \tanh(w_{da_{hidden}}, Input_{da}) \quad (3)$$

$$input_{da} = Concat(DocVec, ContVec\ of\ A_t) \quad (4)$$

For simplicity, the biases are all omitted in the equations. The active function of the output layer is sigmoid function so that each element in DA_w has a value between 0~1.

The second module (bottom right in the figure) is to represent the relationship between the partial summary and the candidate sentence. The module is constructed as a RNN (Recurrent Neural Network), in which the candidate sentence appears in the given partial summary(PS) as a history. Each sentence is represented as the two vectors which contain the content information(ContVec) and the position information(PosVec), respectively. Through the RNN, the partial summary and the candidate sentence are transformed into the final state vector (StateVec). We implement the RNN which uses GRU unit and tanh activation function, and outputs 50-dimensional state vector.

The third module (upper center in the figure) takes the DA_w and StateVec as the input, and calculates the Q-value as the output. It combines the two vectors into element-wise dots and converts them into Q-value through a linear regression function. The output Q-value is the expected value that can be obtained when a new summary is generated by inserting a candidate sentence into a partial summary.

3.2 Reinforcement Learning in Text Summarization

The agent of the text summarization in this work has a sentence *Selector* and three memories. The role of the agent is to generate a summary for an input document by using the sentence *Selector*. The agent has the following three memories. 1) D memory contains information about sentences in an input document. 2) PS memory contains information about sentences in a partial summary generated so far. 3) C memory contains information about the sentences that have not yet been selected in the summary.

After the *Selector* calculates Q-values for each sentence in C memory based on information of D and PS memory, it moves the sentence with the largest Q-value from C Memory to PS Memory. RL method enables the *Selector* to select an appropriate sentence for generating the best summary.

Our Q-Learning algorithm is similar to DQN. Please refer the Volodymyr et al. (2015) for a more detailed explanation about the algorithm. The difference between DQN and our model is table 1 and equation (5)

$$a_t = argmax_{c \in C_t} Q(D, PS_t, c; \theta) \quad (5)$$

Initial state	Next state
$D = \{s_1, s_2, \ldots, s_n\}$, $C_0 = \{s_1, s_2, \ldots, s_n\}$ $PS_0 = \{s_0\}$ $S_0 = \{D, C_0, PS_0\}$	$C_{t+1} \leftarrow C_t - \{a_t\}$ $PS_{t+1} \leftarrow PS_t + \{a_t\}$ $S_{t+1} \leftarrow \{D, C_{t+1}, PS_{t+1}\}$

Table 1. Definition of State0 and Next state in our Q-Learning algorithm

4 Experiments

4.1 Experimental data

In order to evaluate our method, we use DUC 2001, DUC 2002 datasets, Anthology Reference Corpus (ACL-ARC) (Bird, 2008), and WIKI data which have been used in the previous studies Henß et al, (2015).

The numbers of training data and testing data are shown in Table 2.

	DUC 2001	DUC 2002	ACL-ARC	WIKI
#-TR	299	-	5500	1936
#-TE	306	562	613	900

Table 2: Number of each data
#-TR=The number of documents in training data
#-TE=The number of document in testing data

4.2 Features

In this work, a sentence is the unit of summarization. A sentence is represented by both a content vector and a position vector. A content vector(ContVec) represents the meaning of a sentence, and a position vector (PosVec) indicates the position of a sentence in a document. ContVec is estimated by the average of word embeddings in a sentence. We use pre-trained 50-dimensional word embeddings vector from GloVe (Pennington et al., 2014). Due to the lack of training data, word embeddings vector are not updated during the training.

The position of a sentence in a document is useful information for summarization. For example, important sentences are likely to appear in front of a newspaper article. Therefore, we adopt positional information as a main feature.

The positional information has three views.

1) An absolute position of the sentence (PosSent) within a paragraph

2) An absolute position of the paragraph (PosPara) in which the sentence belongs within a section
3) An absolute position of the section (PosSect) in which the sentence belongs in a document.

Each position is encoded as a 50-dimensional vector (PosSentVec, PosParaVec, PosSectVec).
The vector PosVec denotes a sentence position and is calculated as:

$$PosVec = Element\text{-}wise\ sum\ of\ (PosSectVec, PosParaVec, PosSentVec) \quad (6)$$

In Figure 1, each sentence in RNN is represented as the element-wise sum of ContVec and PosVec to take into account the meaning of the sentence as well as its position. For calculating DA_w in Figure 1, a candidate sentence A_t is represented as a ContVec only. DocVec is estimated as the average of embeddings vector of words that occur in the document.

4.3 Experimental Results

Figure 2 shows the training progress on WIKI's validation data. We use ROUGE-2 scores for evaluation. The y-axis of the graph is the ROUGE-2 score for the validation data and the x-axis is the number of validation steps. One single evaluation is performed after every 200 times of mini-batch training.

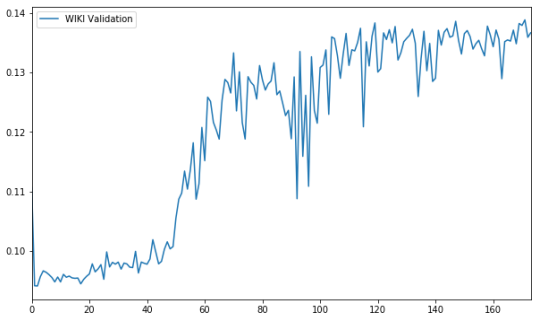

Figure 2: Training progress on WIKI's training and validation data

In Figure 2, the ROUGE-2 scores rarely change until the first 50 steps. We infer that the agent tries various combinations of sentences during this period. However, the ROUGE-2 score surges rapidly between 50 and 60 steps. At this step, the agent comes to know which sentence to choose to generate the best summary. After 120 steps, the model reaches a stable status.

Table 3 shows the first two sentences of the summary results for the Wikipedia article 'Banded sugar ant'. The number '#/#' in Table 3 indicates the positional information. In the third row of table 3, 1/7 means that it was extracted from the first section of the seven sections of the source text. This section has 3 paragraphs, and the first paragraph out of 3 paragraphs, which has 4 sentences. 1/3 and 1/4 mean that the system summary sentence is extracted from this position.

Document Title: Banded sugar ant		
System Summary		
1	1/7 1/3 1/4	The banded sugar ant was first described by German entomologist Wilhelm Ferdinand Erichson, who named it "Formica consobrina" in 1842.
2	3/7 1/2 2/7	It occurs along the north-east coast of Queensland, from Charters Towers in the north to Brisbane in the south.
Human Summary		
1		The banded sugar ant ("Camponotus consobrinus"), also known as the sugar ant, is a species of ant endemic to Australia.
2		A member of the genus "Camponotus" in the subfamily Formicinae, it was described by German entomologist Wilhelm Ferdinand Erichson in 1842.

Table 3: Example of system summary

Table 4 shows the comparison experiments in terms of ROUGE-2.

	ACL-ARC	WIKI	DUC 2001	DUC 2002
TextRank	0.0844	0.1256	0.1866	0.2240
L2R	0.1052	0.1276	0.1934	0.2181
Regression	0.0883	0.1261	**0.1942**	0.2187
RL-full	0.1102	0.1321	0.1993	**0.2252**
Our work	**0.1158**	**0.1398**	0.1832	0.2163

Table 4: Rouge-2 of our work and previous studies

Our method outperforms the previous studies (TextRank(Mihalcea and Tarau, 2005), L2R(Henß et al, 2015), Regression(Henß et al, 2015), RL-full (Henß et al, 2015)) on ACL-ARC and WIKI data but achieves lower performance than the previous studies on DUCs. However, the differences of ROUGE scores between the models are relatively small. These experimental results show that our model is competitive with the previous models. Also, the embedding features have proven to be useful in RL method.

5 Conclusion

In this work, we propose an extractive text summarization model for a single document based on RL method. We use only embedding features that convey meaning and position of a sentence. We also extend the previous study by introducing DQN-based algorithms to train Q-network efficiently and effectively. Evaluation results show that our model is promising for text summarization even though it uses only simple features. Finally, we would like to apply our method to more complex summarization tasks such as multi-document or query focus summarization.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning(NRF-2015R1C1A2A01051685)

References

Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning Seattle, Washington, USA, October 2013, pages 256–265. Association of Computational Linguistics.

R. McDonald. 2007. A study of global inference algorithms in multi-document summarization. Advances in Information Retrieval, pages 557–564.

Cody Rioux, Sadid A. Hasan, and Yllias Chali. 2014. Fear the reaper: A system for automatic multidocument summarization with reinforcement learning. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), October 25-29, 2014, Doha, Qatar., pages 681–690.

HENß, Stefan; MIESKES, Margot; GUREVYCH, Iryna. 2015. A Reinforcement Learning Approach for Adaptive Single-and Multi-Document Summarization. In: GSCL.. pages. 3-12.

MNIH, Volodymyr, et al. 2015. Human-level control through deep reinforcement learning. Nature, 518.7540: 529-533.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Workshop on Text Summarization Branches Out at ACL 2004, Barcelona, Spain, 25–26 July, 2006, pages 74–81.

Steven Bird, Robert Dale, Bonnie Dorr, Bryan Gibson, Mark Joseph, Min-Yen Kan, Dongwon Lee, Brett Powley, Dragomir Radev, and Yee Fan Tan. 2008. The ACL anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC), Marrakech, Morocco, 26 May – 1 June 2008.

Rada Mihalcea and Paul Tarau. 2005. A language independent algorithm for single and multiple document summarization. In Proceedings of the 2nd International Joint Conference on Natural Language Processing, Jeju Island, South Korea, 11–13 October 2005, pages 19–24.

Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In EMNLP. Vol. 14, pages. 1532-1543.

Sutton, R. S., & Barto, A. G. 1998. Reinforcement learning: An introduction(Vol. 1, No. 1). Cambridge: MIT press.

Goldstein, J., Mittal, V., Carbonell, J., & Kantrowitz, M. (2000, April). Multi-document summarization by sentence extraction. In Proceedings of the 2000 NAACL-ANLPWorkshop on Automatic summarization-Volume 4 (pp. 40-48). Association for Computational Linguistics.