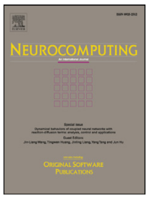
Neurocomputing 284 (2018) 52–62

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



Deep reinforcement learning for extractive document summarization

Kaichun Yao[a], Libo Zhang[b,*], Tiejian Luo[a], Yanjun Wu[b]

[a] University of Chinese Academy of Science, Beijing 101408, China
[b] Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

A R T I C L E   I N F O

Article history:
Received 11 July 2017
Revised 14 December 2017
Accepted 1 January 2018

Communicated by Dr. Tie-Yan Liu

MSC:
00-01
99-00

Keywords:
DQN
Extractive summarization
Hierarchical architecture
Rouge metric

A B S T R A C T

We present a novel extractive document summarization approach based on a Deep Q-Network (DQN), which can model salience and redundancy of sentences in the Q-value approximation and learn a policy that maximize the Rouge score with respect to gold summaries. We design two hierarchical network architectures to not only generate informative features from the document to represent the states of DQN, but also create a list of potential actions from sentences in the document for the DQN. At training time, our model is directly trained on reference summaries generated by human, eliminating the need for sentence-level extractive labels. For testing, we evaluate this model on the CNN/Daily corpus, the DUC 2002 dataset and the DUC 2004 dataset using Rouge metric. Our experiments show that our approach achieves performance which is better than or comparable to state-of-the-art models on these corpora without any access to linguistic annotation. This is the first time DQN has been applied to extractive summarization tasks.

1. Introduction

In the era of the information explosion, it is impossible to obtain all the information only through the human brain, especially the overwhelming news reports, massive text messages, etc. How to help people effectively have access to critical and salient information becomes a challenge. Therefore, it is necessary to develop an automatic summarization system to effectively extract the key information of the text and to guarantee the minimum loss of the information for the summary. At present, many efforts have been made to the following summarization approaches: abstractive summarization and extractive summarization. Abstractive methods aim to concisely paraphrase the information content in the documents to produce a condensed summary while extractive summarization techniques target at selecting salient words, sentences or passages from source documents.

Recently, the emergence of generative models based on neural network for text [1] makes abstractive approaches increasingly popular [2,3]. Especially, Nallapati et al. [4] modified an existing corpus that was used for the task of passage-based question answering [5] and then produced a multi-sentence summaries based corpus — the CNN/Daily Mail corpus which is available to train deep learning models for abstractive or extractive summarization.

In spite of the fact that the current deep learning approaches for abstractive summarization have achieved good performance on short input and output sequences, and abstractive approaches are consistent with the pattern of human beings summarization generation, these approaches have a poor performance for longer documents and summaries. Hence, extractive approaches are still appealing since they are less complicated, and produce grammatically and semantically correct summaries, so we do not have to worry about the genuine linguistic quality. A series of traditional techniques for extractive approaches have been proposed, such as greedy approaches [6], hidden Markov models [7], graph based approaches [8,9], constraint optimization based approaches [10], and integer linear programming [11], etc.

Neural network based approaches for extractive summarization have gained huge popularity with the development of deep learning techniques. For instance, Kageback et al. [12] used the recursive autoencoder [13] to summarize documents, achieving the best result on the Opinosis dataset [14]. Yin and Pei [15] used Convolutional Neural Network (CNN) to project sentences into dense distributed representations and then selected sentences by minimizing the cost based on their prestige and diverseness for the task of multi-document extractive summarization. In a recent work, Cheng and Lapata [16] proposed an attentional encoder-decoder of classifier model for extractive single-document summarization which is trained on the CNN/Daily Mail corpus. Nallapati et al. [17] also treated extractive summarization as binary-classification

* Corresponding author.
E-mail address: libo@iscas.ac.cn (L. Zhang).

https://doi.org/10.1016/j.neucom.2018.01.020
0925-2312/© 2018 Elsevier B.V. All rights reserved.

task and proposed a Recurrent Neural Network (RNN) based sequence model which selects sentences according to their content, salience and novelty. In addition to using supervised learning approaches in their works for extractive summarization, Ryang and Abekawa [18] constructed an extractive summarization system within the framework of reinforcement learning, and Rioux et al. [19] did the similar work with reinforcement learning methods.

Inspired by extractive summarization approaches based on traditional reinforcement learning and deep neural network, we develop a deep reinforcement learning framework for extractive summarization tasks by employing a Deep Q-Network(DQN). More specifically, we deploy a CNN-RNN hierarchical network architecture to not only generate informative features from text to represent the states of DQN, but also create a list of potential actions from the text for the DQN. At the same time, our DQN learns to make decision on which action (*e.g.* sentence) will be selected from action spaces (*e.g.* sentences in the document) according to their information content, salience and redundancy in the Q-value function approximation. In addition, we also deploy a RNN-RNN hierarchical network to compare the performance for CNN and RNN in encoding sentences. The difference of two hierarchical networks is that they encode each word in sentence to a fixed dimensional sentence representation with CNN or RNN in the bottom layer. After that, these sentence representations will be the inputs of RNN in the top layer to generate the hidden state representations of sentences and document content representation. In this paper, our work concentrates only on sentential extractive summarization of single document using deep reinforcement learning. We train our DQN model on the CNN/Daily Mail corpus used by Nallapati et al. [20] and Cheng et al. [16]. Unlike their supervised classifier models creating labels manually by converting abstractive summaries to extractive labels, our extractive summarization approach is based on deep reinforcement learning which allows us to directly optimize our summarizing tasks by using Rouge [21] metrics, so it eliminates the need of extractive labels and does not incur additional annotation costs.

We evaluate our model automatically on the following datasets: the benchmark DUC 2002 document summarization corpus and the CNN/Daily Mail news highlights corpus. Evaluations are performed using ROUGE for ROUGE-1, ROUGE-2 and ROUGE-L values of ROUGE metrics for extractive summarization. We also evaluate our approach using the DUC 2004 dataset for comparison with the results in other traditional reinforcement learning methods such as Ryang and Abekawa [18], and Rioux et al. [19], which is a multi-document extractive summarization corpus. The main contributions of this paper are as follows: (i) We apply DQN in extractive summarization tasks for the first time. (ii) We design two hierarchical network architectures (*i.e.* CNN-RNN and RNN-RNN) for DQN to capture local and global features of the document. (iii) In the Q-value approximation, we model salience and redundancy of sentences in the document to help DQN make better policy. Experimental results show that our summarizer achieves performance which is better than or comparable to state-of-the-art approaches employing hand engineered features and sophisticated linguistic constraints.

2. Background

2.1. Definition of extractive summarization

Approaches of extractive summarization are used to generate a summary by extracting some sentences or words from the original document, and extractive methods are able to yield naturally grammatical summaries so that we don't worry about the genuine linguistic quality.

In this paper, the extractive summarization tasks are defined as follows. Given a document D consisting of a sequence of sentences $\{x_1, x_2, ..., x_m\}$, and each sentence is composed of n words $\{w_1, w_2, ..., w_n\}$ where $|D| = m,\ x_i$ represents a single sentence in the document and w_k represents a word in the sentence. The objective of extractive summarization is to create a summary *Sum* from the source document *D* by selecting a subset of *j* sentences (where $j < m$), and *Sum* is a subset of the document: $Sum \subset D$.

Next, a score function score(*Sum*) and a length function $L(Sum)$ are defined, where the subset Sum is the summary generated from the given document and $L(Sum)$ indicates the length of the summary *Sum*. The score function is defined in Section 3.4, which usually takes the tradeoff between relevance and redundancy into consideration. In extractive summarization tasks, the length of summary is arbitrary, which can be based on the word or sentence, and we assume the summary length is limited to *K*. The extractive summarization problem is defined as follows:

$$\begin{aligned} & Sum^* = argmax\, score(Sum) \\ & s.t. \quad L(Sum) \le K\, and\, Sum \subset D \end{aligned} \tag{1}$$

The aim of the summarization problem is to find the optimal summary Sum^* that maximizes this score function for its provided document *D*.

2.2. Reinforcement learning and deep Q-network

A standard reinforcement learning setup consists of an agent interacting with its environment [22,23] in discrete iteration steps. Given a set of internal states $s = s_1, ..., s_l$ and a set of predefined actions $A = a_1, ..., a_K$, at each iteration step *i*, the agent receives a state s_i and takes an action a_i by following certain policies. Then a new state s_{i+1}, and a scalar reward r_i will be received from environment. The objective of the agent is to maximize the cumulative reward in the process of a sequences of actions. Each such action constitutes a transition tuple $(s_i, a_i, r_i, s_{i+1},)$ of a Markov Decision Process (MDP). Generally speaking, the environment may be partially observed, and a sequence of state transition tuples can be used to formulate the environment.

Reinforcement learning methods are widely used in controling system [24,25]. Among these technologies, Q-Learning [26] is a model-free technique which is used to approximate optimal action-value function $Q(s, a)$ that measures the action's expected long-term reward for the agent. The Q-value function depends on all possible state-action pairs which are visited and updated by the policy. The main challenge for Q-learning is how to approximate the action-value function $Q(s, a)$. Typically, a parameterized function $Q(s, a; \theta)$ is used to approximate $Q(s, a)$, where the parameter θ is often learned from feature representations of the states and actions of the environment [27,28]. Promisingly, with the popularity of deep learning techniques, deep learning has shown the powerful advantage for effectively generating informative feature representations for a variety of complex problems. Mnih et al. proposed the Deep Q-Network (DQN) [29], which approximates the Q-value function with a non-linear deep convolutional neural network and automatically creates informative features to represent the internal states of the reinforcement learning.

In DQN, the agent learns a policy by interacting with environment at each iteration step *i*, and targets at maximizing its long term reward. Beginning with a random Q-value function, the agent takes actions according to the current policy, obtains rewards from environment and continuously updates its Q-values at each iteration step. The iterative updates are derived from the Bellman equation [28], shown as follows:

$$Q_{i+1}(s, a) = E[r + \lambda max_{a'} Q_i(s', a'|s, a)] \tag{2}$$

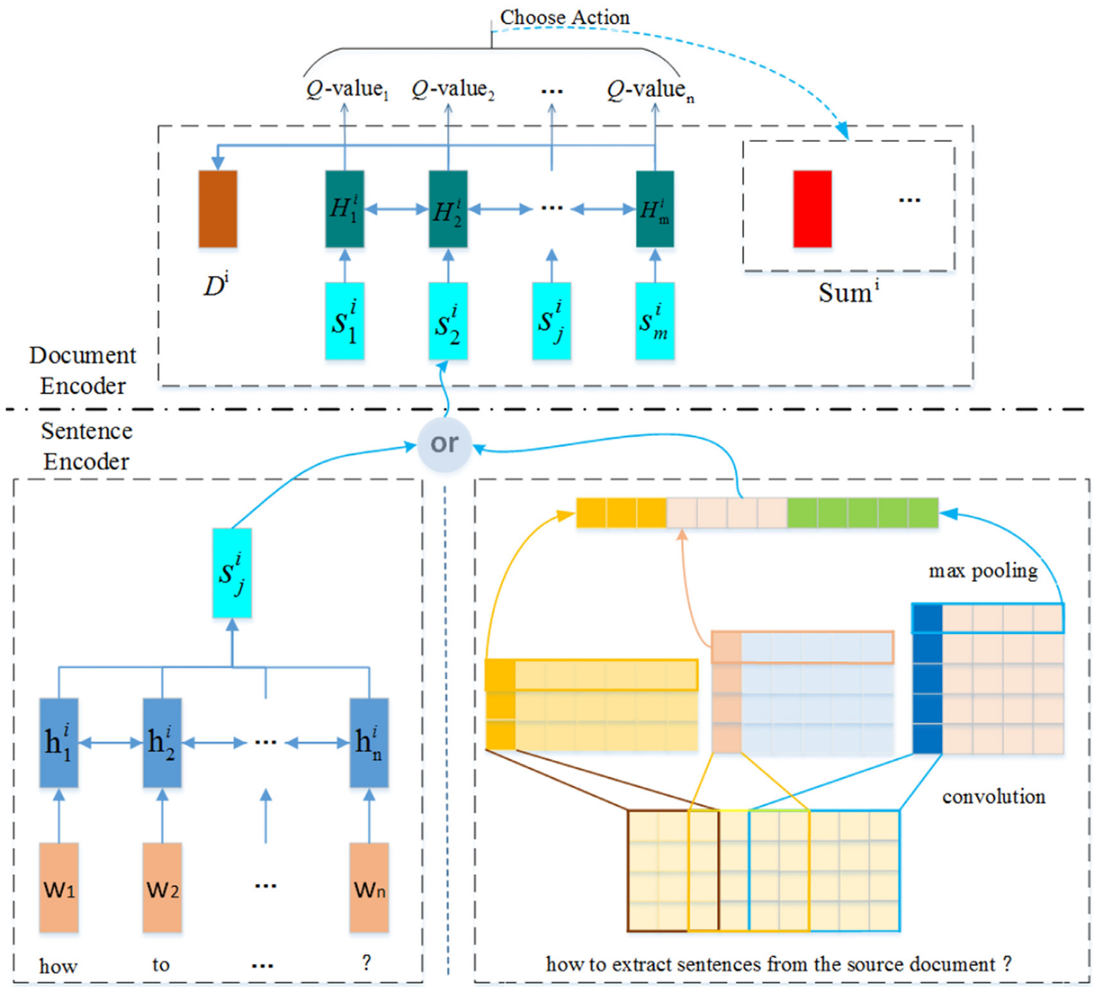


Fig. 1. The two different hierarchical network architectures of DQN for extractive summarization (RNN-RNN architecture and CNN-RNN architecture). The left-bottom in the graph is RNN operating at word-level, the right-bottom in the graph is CNN operating at word-level, and the top in the graph is RNN running over sentences. Double-pointed arrows indicate a bi-directional RNN. The word *or* in the middle represents RNN or CNN as our sentence encoder. The character *i* indicates each iteration step for the DQN, D^i on the left-top denotes the document content representation at iteration step *i*, and Sum^i on the right-top denotes the summary representation at iteration step *i*.

where r is reward and E denotes the expectation which often compute over all transition tuples that involve the agent taking action a in state s. λ denotes a discounted factor for future rewards and $\lambda \in [0, 1]$. Setting $\lambda = 0$ means that the current policy is short-sighted and it only pays attention to the reward of the current action. On the contrary, a larger value for λ denotes more confidence on the future. A good policy should balance the current reward and the future reward, and maximize the cumulative reward.

3. DQN for extractive summarization

In this work, we develop a deep reinforcement learning framework for extractive summarization tasks by employing a DQN. The DQN model is schematically illustrated in Fig. 1. In detail, we deploy two different architectures based on CNN-RNN hierarchical network or RNN-RNN hierarchical network to not only generate informative features from the document to represent the states of DQN, but also create a list of potential actions from sentences in the document for the DQN. From the hierarchy, CNN or RNN in the bottom layer is a sentence encoder operating at word level within each sentence, which encodes words to a fixed dimension sentence feature vector. RNN operating at word level is depicted in left bottom in Fig. 1 and CNN operating at word level is depicted in right-bottom in Fig. 1. RNN in the top layer is a document encoder running over sentence level, which encodes sentence feature vectors to a document feature vector, depicted in top in Fig. 1. In practice, sentence encoder and document encoder are a dynamic encoding process in our DQN, which continually generates the representations of words, sentences, document content and summary at each iteration of the DQN.

3.1. *Sentence encoder*

3.1.1. *RNN structure*

Recurrent neural network has achieved promising performances in text sequence processing tasks, especially handling a variable-length sequence [30]. The RNNs with the gating units have been shown to outperform vanilla RNN on many tasks [31]. Therefore, we use a gated recurrent unit (GRU) based Recurrent Neural Network (RNN) [31] as the basic building block of our sentence encoder. RNN based sentence encoder is a bi-directional recurrent network with GRU and we denote it as Bi-GRU-RNN. A GRU consist of the update gate, the reset gate and the activation and it can adaptively capture dependencies of different time scales, as shown by the following equations:

$$u_t = \sigma(W_{ux}x_t + W_{uh}h_{t-1} + b_u) \tag{3}$$

$$r_t = \sigma(W_{rx}x_t + W_{rh}h_{t-1} + b_r) \tag{4}$$

$$h'_t = tanh(W_{hx}x_t + W_{hh}(r_t \odot h_{t-1}) + b_h) \quad (5)$$

$$h_t = (1 - u_t) \odot h'_t + u_t \odot h_{t-1} \quad (6)$$

where $\sigma(\cdot)$ and $tanh(\cdot)$ indicate the element-wise sigmoid and hyperbolic tangent functions, x_t, u_t, r_t and h_t are referred to as the input vector, the update gate, the reset gate and the hidden-state output vector at time step t, and $\odot$ is an element-wise multiplication operator. The W's and b's are the parameters of the Bi-GRU-RNN.

This layer consists of bi-directional GRU-RNN, and operates at word level, as presented in the left-bottom in Fig. 1. The forward RNN generates hidden state representations at each word position in sequence according to the current word embedding input and the previous hidden state. At the same time, the backward RNN also computes hidden state representations for each word in reversed sequence. That is to say, the pair of forward and backward RNNs is considered as a bidirectional RNN. Then, we obtain the sentential representations from the average pooling of the concatenated hidden states of the bi-directional RNN running at word-level, which used as inputs to document encoder (a Bi-GRU-RNN structure as well). The representation vector for each sentence in the document is as shown below:

$$S = \frac{1}{N_s} \sum_{j=1}^{N_s} [h_j^f, h_j^b] \quad (7)$$

where h_j^f and h_j^b are the hidden states corresponding to the jth word of the forward and backward word-level RNNs respectively, N_s is the number of words in a sentence and [ ] represents vector concatenation.

3.1.2. CNN structure

We also explore CNN structure as our sentence encoder. That is, we generate sentential representation vectors using a single-layer convolutional neural network with a max pooling operation [32–34]. Likewise, the CNN runs over words in each sentence, resulting to sentence-level representations which are then used as inputs to document encoder (a Bi-GRU-RNN structure as well).

There are two reasons for obtaining sentential representations with a convolutional neural network. At first, CNN has shown success in natural language processing tasks such as text sentiment analysis [35], character-aware neural language models [34] and so on. What's more, compared with RNN, CNN can be trained more effectively and process text faster due to no long-term dependencies.

Let V denote the vocabulary of words in the document, d denote the dimensionality of word embeddings, and $Q \in R^{(d \times |V|)}$ be the matrix word embeddings. Suppose word $w \in V$ and a sequence of n words (w_1, w_2, w_n) constitutes a sentence in the document, which can be represented by a dense column matrix $W \in R^{(n \times d)}$. We execute a narrow convolution operation between sentence representation matrix W and a filter (or kernel) $K \in R^{(c \times d)}$ of width c as follows:

$$f_j^k = tanh(W_{j:j+c-1} \odot K + b) \quad (8)$$

where $\odot$ denotes an element-wise multiplication operator. f_j^k denotes the jth element of the kth feature map f^k and b denotes the bias. We perform max-pooling operation over time to obtain a single feature (the kth feature) representing the sentence corresponding to the kernel K:

$$y^k = max_j f_j^k \quad (9)$$

In order to obtain different feature maps of a sentence to represent sentence vectors, we use multiple feature maps to compute a list of features that match the dimensionality of a sentence under each kernel width. In addition, we apply multiple kernels with different widths to obtain a set of different sentence vectors. Finally, we concatenate these feature maps to obtain the final sentence representation. The CNN structure is schematically illustrated in the right-bottom of Fig. 1. In the example, the sentence embeddings have twelve dimensions, so twelve feature maps are used under each kernel width. The orange feature maps have width three, the pink feature maps have width four and the blue feature maps have width five. The sentence embeddings obtained under each kernel width are concatenated to get the final sentence representation.

In practice, we obtain a new set of features by running y^k through one layer of a highway network [36], which is regarded as the real inputs to next Bi-GRU-RNN layer. One layer of a highway network describes as follows:

$$S = t \odot g(W_H y + b_H) + (1 - t) \odot y \quad (10)$$

where g is a nonlinearity, $t = \sigma(W_T y + b_T)$ is called the transform gate, and $(1 - t)$ is called the carry gate. In order to ensure the same dimensions for y and S, W_T and W_H are square matrices.

3.2. Document encoder

Document encoder also consists of a bi-directional GRU-RNN, which has same structure with sentence encoder based on Bi-GRU-RNN, but it runs at the sentence-level and accepts the outputs of sentence encoder as its inputs. Its graphical representation is presented in top in Fig. 1. Firstly, document encoder encodes sentential representation vectors obtained from the bottom layer to the corresponding hidden states. Next, we concatenate the hidden states of forward and backward RNNs, which perform a non-linear transformation to obtain the representation of the entire document, as shown blow.

$$D = tanh\left(W_d \frac{1}{N_d} \sum_{j=1}^{N_d} [H_j^f, H_j^b] + b_d \right) \quad (11)$$

where H_j^f and H_j^b denote the hidden states corresponding to the jth sentence of Bi-GRU-RNN in top layer respectively, N_d is the number of sentences in the document, b_d is the bias and '[ ]' represents vector concatenation.

Document encoder is the import component in whole model, which not only generates hidden states for each sentences in the document, but also composes a sequence of sentences into a document representation. At the same time, the representation of the current summary depends on the hidden state of the sentence. The hidden states of document encoder may be treated as a list of partial representations, which concentrates mostly on the corresponding input sentence given the previous and following context and captures local sentential information. What's more, these representations altogether constitute the document representation, which is used to capture global sentential information.

3.3. Training the DQN

In this section, we describe how to train our DQN model in detail. The algorithm for training deep Q-networks is depicted in Algorithm 1. The hierarchical network consisting of sentence encoder and document encoder not only learns the representations of the internal states for the DQN, but also generates a set of action representations for sentences in document.

At each iteration step i, we encode each sentence in the document consisting of n words $\{w_1, w_2, w_n\}$ using sentence encoder, whose inputs is the corresponding word embedding vectors. This

Algorithm 1 Deep Q-network training for extractive summarization.

```
1: Initialize replay memory $R$ to capacity $N$
2: Initialize action-value function $Q(s, a; \theta)$ with random weights $\theta$
3: Initialize target action-value function $\hat{Q}(s, a; \theta)$ with weights $\theta = \hat{\theta}$
4: randomize given training set with sequence pairs $< X, Y >$
5: for $epsiode = 1, M$ do
6:     for each sequence pair source text $x \in X$ and gold summary $y \in Y$ do
7:         initialize the state $s^1$, that is $Sum^1 = \varnothing$
8:         for iteration $i = 1, L$ do
9:             feed $x$ into sentence encoder and document encoder, and obtain sentence representations $H^i_j$ and document representation $D^i$
10:            if $random() < \epsilon$ then
11:                select a random action $a^i$
12:            else
13:                compute $Q(s, a; \theta)$ for all actions using equation (10); select $a^i = argmax_a Q(s, a; \theta)$
14:            end if
15:            add action $a^i$ to the current summary to obtain new state $s^{i+1}$ ($Sum^{i+1}$), and compute the similarity of current summary and gold summary $y$ , resulting reward score $r^i$
16:            store transition ($s^i$; $a^i$; $r^i$; $s^{i+1}$) in $R$
17:            sample a mini-batch of transitions ($s^i$; $a^i$; $r^i$; $s^{i+1}$) from $R$
18:            set $q^i = \begin{cases} r^i & \text{if } s^{i+1} \text{ is state of termination,} \\ r^i + \lambda max_{a'} \hat{Q}(s^{i+1}, a'; \hat{\theta}) & \text{otherwise.} \end{cases}$
19:            perform gradient descent step on $(q^i - Q(s^i, a^i; \theta))^2$ with respect to network parameters $\theta$
20:            every $C$ steps reset $\hat{\theta} = \theta$
21:        end for
22:    end for
23: end for
```

encoding process running at word level results in a fixed dimensional vector representation S^i_j (*i.e.*, S^i_1, S^i_2, , S^i_m in Fig. 1), and its formulation representation is Eq. (7) for RNN structure or Eq. (10) for CNN structure in Section 3.1. Sentence encoder generates the vector representation of each sentence in the document which is used as the inputs to document encoder.

Next, we feed sentence vector S^i_j into document encoder, generating the hidden layer vector representation H^i_j for each sentence (*i.e.*, H^i_1, H^i_2, H^i_m in Fig. 1). In addition, we obtain the representation of the extracted summary Sum^i (*i.e.*, Sum^i in Fig. 1) and the representation of document content D^i (*i.e.*, D^i in Fig. 1) for the entire sentence. We utilize Eq. (11) in Section 3.2 to compute the document content representation at each iteration step *i*, and the summary representation will be given later.

The DQN learns to approximate the Q-value function based on the current state. Generally speaking, there is no practical meaning for the traditional Q-value approximation in specific tasks such as video games [37,38]. How to approximate the Q-value function in meaningful way? In our extractive summarization approach, we consider the information content, the salience and the redundancy between the action (sentence vector representation) and the current state (the extracted summary) in the Q-value function *Q*(*s*, *a*) approximation. The Q-value function *Q*(*s*, *a*) is estimated by the following formulation:

$$Q^i_j(s, a) = W_C H^i_j + {H^i_j}^T W_s D^i - {H^i_j}^T W_r tanh(Sum^i) + W_p P_j + b_q \quad (12)$$

where H^i_j denotes the hidden state representation of the *j*th sentence given by document encoder at iteration step *i; D^i* is the entire document representation at iteration step *i; P_j* is the positional embedding of the sentence in the document; W_C, W_s and W_p are parameter vectors in order to model the information content richness, the salience of the sentence and positional importance of sentences respectively; the term $H^T_j W_r \tanh(Sum^i)$ captures the redundancy of the sentence with respect to the current summary and Sum^i is the dynamic representation of the summary at current iteration step *i*. Given a set Ω representing the current chosen the number of the sentence in the document by the DQN, Sum^i is shown as follows:

$$Sum^i = \sum_j \frac{Q^i_j(s, a)}{\sum_{k=1}^m Q^i_k(s, a)} H^i_j \qquad s.t. \quad j \subset \Omega \quad (13)$$

The initial state of *Sum* should be set to ∅. That is, $Sum^0 = \varnothing$. The summary representation depends on the sentence hidden states and the Q-value of the corresponding actions (sentences) taken by the DQN. The Q-value for each action in Ω is normalized to the respective weight at each iteration step, so it is dynamic process to generate the summary representation dynamic process.

The DQN considers the vector representation Sum^i as its internal state. Also, $\{H^i_1, H^i_2, ..., H^i_j, ..., H^i_m\}$ are treated as the potential action representations which directly correspond to the number of every sentence {1, 2, ..., *j*, ..., *m*} in the document. From these sequence of sentence numbers (action spaces), the DQN learns to predict what actions should be taken in order to accumulate larger long-term reward.

The DQN will take the action with the max Q-value as its output. The action took by the DQN is considered as selecting the number of the sentence in document at iteration step *i*. Then the current state of the DQN will be modified accordingly. That is, the summary will be modified by adding a new sentence into it. Especially, at different iteration step, the document content and the representation of the sentences will be updated as well due to the modification of the parameter of the DQN. This process results in a modified representation of sentence, summary and document, namely H^{i+1}, Sum^{i+1} and D^{i+1}. Next, the score between the gold summary and the current extracted summary is evaluated by a ROUGE metric, which then a reward r^i is assigned to the action of selecting the corresponding sentence. Thus, a transition tuple for the DQN contains $[Sum^i, H^i_j, r^i, Sum^{i+1}]$.

The training of the DQN is treated as a process of minimizing a sequence of loss functions, and the weight parameters θ of Q-network is updated at each iteration *i*. In practice, we use two network to improve the stability of algorithm [37]. The original Q-network Q with parameters θ approximates the Q-value, and a target Q-network $\hat{Q}$ with parameters $\hat{\theta}$ will be used to generate the targets q^i in the Q-learning update. The loss function $L^i(\theta^i)$ is given as follows:

$$L^i(\theta^i) = E_{s,a}[(q^i - Q^i(s, a; \theta^i))^2] \quad (14)$$

where $q^i = E_{s,a}[r^i + \lambda max_{a'} Q^i(s^{i+1}, a'; \hat{\theta}) | s, a]$ is the target Q-value or reward, which is generated by the target Q-network with parameters $\hat{\theta}$ from next state s^{i+1}. Namely, the DQN is trained to predict its expected future reward. The parameters θ^i of loss function $L^i(\theta^i)$ is updated with the following gradient:

$$\nabla_{\theta^i} L^i(\theta^i) = E_{s,a}[2(q^i - Q^i(s, a; \theta^i)) \nabla_{\theta^i} Q^i(s, a; \theta^i)] \quad (15)$$

We clone the original Q-network *Q* to the target Q-network $\hat{Q}$ after every *C* update steps and use $\hat{Q}$ to generate the Q-learning targets q^i for the following *C* update steps. In the process of training, a experience replay [39] is used to store the agents experience (transition tuple $[Sum^i, H^i_j, r^i, Sum^{i+1}]$) at every iteration step. Besides, the agent chooses the action with the maximal Q-value in order to maximize its expected future rewards or selects a random action with probability ϵ [40] by an ϵ-greedy policy to ensure adequate exploration of the state space. The full algorithm is presented in Algorithm 1.

3.4. Rouge score for DQN reward

The DQN selects sentences (actions) from the document should be the ones that maximize the Rouge score with respect to ground truth summaries. Our reward function calculate the similarity between the gold summary $\{y_1, ..., y_j\}$ and the current extracted summary $\{x_1, ..., x_j\}$, which is based on the n-gram concurrence score metric and the longest-common-subsequence recall metric contained within ROUGE [21], in terms of ROUGE-1, ROUGE-2, ROUGE-3 and ROUGE-L in our work, shown as follows:

$$reward(Sum) = \begin{cases} -1 & \text{if length}(Sum) > K, \\ score(Sum) & \text{otherwise.} \end{cases} \tag{16}$$

We measure the score difference between the current iteration and the previous iteration, hence, $score(Sum)$ is formulated as:

$$score(Sum) = \begin{cases} score(S_c) - score(S_p) & \text{if score}(S_n)\text{-score}(S_p) \\ & >threhold, \\ -1 & \text{otherwise.} \end{cases} \tag{17}$$

where S_c and S_p represent the summary extracted in the current iteration step and in the previous iteration step, and *threhold* is a constant, which means the sentence extracted in the current iteration for current summary must be informative and effective compared to previous summary. The score function is defined as follows:

$$score(S_{n/p}) = 1 \times Rouge\_1 + 5 \times Rouge\_2 + 2 \times Rouge\_3 + 2 \times Rouge\_L \tag{18}$$

The different weights are assigned to the score of $Rouge\_1$, $Rouge\_2$, $Rouge\_3$, and $Rouge\_L$ respectively. The larger weight means we pay more attentions to the corresponding Rouge metric.

4. Related works

The traditional approach to resolve extractive summarization problem such as TextRank [41] is by ranking sentences importance according to their salience and novelty. Some researchers regard extractive document summarization as a binary classification problem. For example, Shen et al. [42] have proposed a modeling approach that extracts sentences from the source document using Conditional Random Fields to binary-classify sentence sequences.

Especially in recent years, the deep learning method is applied to the extractive document summarization task, but most datasets for single-document summarization tasks such as DUC corpora are too small to train deep learning models. Nallapati et al. [4] and Cheng et al. [16] have reconstructed a new corpus based on news stories from CNN and Daily Mail, which is made up of around 280,000 documents and human generated summaries. On this datasets, Cheng and Lapata [16] have constructed a classifier framework based on an encoder–decoder model where the encoder learns the representation of sentences and documents while the decoder classifies each sentence using an attention mechanism. Nallapati et al. [17] have proposed an approach named SummaRuNNer, a sequence classifier based on a hierarchical recurrent network structure that selects sentences from the source document according to their information content, salience and redundancy.

Our approach is very different from theirs that we use deep reinforcement learning framework (a Deep Q-Network) to select sentences that eliminate the need for extractive labels. The classifier based approach is a supervised learning and it requires sentence-level labels to train the model for extractive summarization. Svore et al. [43] convert the abstractive summaries to extractive labels via a simple greedy approach in their model. Further, an integer linear programming (ILP) based approach recently proposed by Cao et al. [44] solve this problem optimally.

It is different from the classifier based approaches that Ryang et al. [18] and Rioux et al. [19] have proposed an automatic extractive summarization framework using reinforcement learning approach. The former proposed an approach named ASRL which is used to solve multi-document summarization tasks. ASRL uses $TD(\lambda)$ algorithm to learn and then execute a policy for a summarizing tasks. Another reinforcement learning method named REAPER proposed by the latter is very similar to ASRL. In their experiments, the generated summary is evaluated using ROUGE-1, ROUGE-2, and ROUGE-L, and the results indicate that ASRL and REAPER based on reinforcement learning approach outperform greedy and integer linear programming (ILP) techniques on DUC 2004 dataset.

Unlike traditional reinforcement learning, we use a Deep Q-Network (DQN) to learn the representation of sentences and documents. At the earliest, the DQN has been applied in playing Atari games [37,38,45] successfully. Trained with a variant of Q-learning [46], the DQN uses a deep neural network to generate informative features to represent the internal states and approximate policy function for taking action. In addition to playing video games, other researchers have employed DQN to learn control policies in natural language tasks such as playing text-based games [47–49]. In our work, we employ two different hierarchical networks to not only generalize informative features from text to represent the states of DQN, but also create a list of potential actions from the text for the DQN, and the actions taken by the DQN constitute our summary.

5. Experiments and results

5.1. Corpora

The CNN/Daily Mail corpus is used in our experiments for training and testing our DQN model. It is originally created for the task of passage-based question answering [5] and reconstructed for the task of extractive summarization [16,17] and abstractive summarization [4]. Our extractive approach will make a fair comparison with the two former [16,17] only on the Daily Mail corpus, and to compare with the latter [4], we evaluate it on the joint CNN/Daily Mail dataset. There are 196,557 training documents, 12,147 validation documents and 10,396 test documents in the Daily Mail corpus. In total, we have 286,722 training sets, 13,362 validation sets and 11,480 test sets for the whole CNN/Daily Mail corpus. There are about 800 words per document in training set, and each of document is about 28 sentences on an average while the gold summaries contain an average of 3–4 sentences.

Besides, we also evaluate our model on the DUC 2002 single-document summarization dataset consisting of 567 documents and on the DUC 2004 dataset consisting of 50 document clusters, each containing 10 documents as an additional out-of-domain test set.

5.2. Evaluation

In our experiments, the quality of summaries extracted by our approach is automatically evaluated by using the Rouge metric [21], which computes the number of overlapping units such as n-gram, word sequences, and word pairs between the computer-generated summary and the ground truth summary. To compare with other extractive summarization approaches such as Cheng et al. [16] and Nallapati et al. [17], 75 bytes and 275 bytes of limited length Rouge recall are used on the Daily Mail corpus.

Likewise, the full-length Rouge F1 metric is used on the whole CNN/Daily Mail corpus in order to compare with the work of Nallapati et al. [4]. According to the official guidelines of DUC 2002 corpus, we use the limited length Rouge recall metric at 75 words. To compare with other approaches of reinforcement learning [18,19], we also test our approach on the DUC 2004 dataset. We compute the score of Rouge-1, Rouge-2 and Rouge-L using the matches of unigrams, bigrams and longest common subsequences respectively, which assess informativeness and fluency between our extracted summaries and reference summaries.

5.3. Baselines

The summary generated by our DQN model is compared with other various summarization methods on different datasets. On the Daily Mail corpus, the joint CNN/Daily Mail corpus and the DUC 2002 corpus, Lead-3 model is used as the standard baseline, which simply selects the leading three sentences from the document as the extracted summary. In addition, we also compare with the extractive and abstractive approaches based on deep learning on these datasets. For example, Cheng et al. [16] have proposed a deep learning based supervised extractive model which achieves high performance. Nallapati et al. [4] have proposed an abstractive method using attentional encoder-decoder recurrent neural retworks which achieves state-of-the-art performance on abstractive summarization tasks. Nallapati et al also proposed two extractive approaches — Deep Select [50] and SummaRuNNer [17]. The former is based on a deep learning selector architecture and the latter is a deep learning classifier model which achieves state-of-the-art performance on two different corpora. On the Daily Mail and DUC 2002 datasets, LReg model, a feature-rich logistic classifier, is also used as a baseline. On DUC 2002 corpus, several approaches such as Integer Linear Programming based ILP proposed by Woodsend and Lapata [11], graph based TGRAPH proposed by Parveen, et al. [51] and URANK proposed by Wan [52] are reported as baselines, which achieve high performance on this corpus. On DUC 2004 corpus, ILP approach is used as a baseline once again. At the same time, ASRL and REAPER, two traditional approach based on reinforcement learning proposed by Ryang et al. [18] and Rioux et al. [19] respectively, are also used as baselines on this dataset.

5.4. Experimental settings

We present our experimental settings for evaluating the performance of our DQN model in this section. The word vectors inputted as sentence encoder were initialized with 100-dimensional Glove [53] embeddings pre-trained on the joint CNN/Daily Mail corpus. We trained our model using the RMSProp[1] algorithm with mini-batches of size 64 at training time. The vocabulary size is limited to 6K. In order to speed up computation, the maximum number of sentences per document is restricted to 50, and the maximum sentence length to 50 words. That is to say, each sentence per document was padded to the maximum length with an additional mask variable. We fixed the hidden state size at 200 for Bi-GRU-RNN structure of sentence encoder. but for CNN structure, we used a list of kernel sizes {1, 2, 3, 4, 5, 6, 7} and corresponding feature maps {40, 50, 50, 60, 60, 70, 70}, so the size of sentence and document embeddings is 400 and 800, respectively. The depth of each GRU module was 1. We initialized all parameters for CNN or RNN randomly with a uniform distribution within [−0.1, 0.1].

For the DQN training, we set the epsilon ϵ to 0.9, which decreased 0.001 automatically after each 1000 iterations. In other words, most of actions were random at the beginning of the DQN training, and then became more and more greedy as the training process carry on. We used a prioritized experience replay [54] memory with a capacity of 200,000 to save the transition in each iteration. The discount factor λ for reward was set to 0.95. Also, we set the difference threshold between the summary extracted in previous iteration and the summary extracted in current iteration to 0.1. We trained our DQN model based on CNN-RNN structure and RNN-RNN structure, so we respectively denote them as $DQN_{cnn-rnn}$ and $DQN_{rnn-rnn}$ in the results. All our experiments were run on a NVIDIA GTX 1080 GPU with 8GB memory and our DQN model with CNN-RNN structure and RNN-RNN structure took about 10 days and 15 days of training time until convergence, respectively.

5.5. Results

5.5.1. Daily mail Corpus

In Table 1 we report the results of our DQN model on the Daily Mail corpus using Rouge recall at 75 bytes of summary length and at 275 bytes of summary length, respectively. The performance of DQN model is compared with state-of-the-art model - SummaRuNNer [17] and other baselines. $DQN_{cnn-rnn}$ represents our DQN model based on CNN-RNN hierarchical architecture and $DQN_{rnn-rnn}$ denotes our DQN model based on RNN-RNN hierarchical architecture.

Our DQN model performs worse than the state-of-the-art model — SummaRuNNer at 75 bytes of summary length and performs on par with it at 275 bytes of summary length. The potential reason may be that our approach models information content, salience and redundancy of sentences based on the current summary content, therefore our approach has an advantage over long summary. Although the results achieved by the DQN at 275 bytes are slightly worse than that of SummaRuNNer, SummaRuNNer is trained by supervised learning and it needs sentence-level binary labels for each document, while our DQN model optimizes directly Rouge metric eliminating the need for extractive labels. What's more, the experiments show the model based on CNN-RNN or RNN-RNN hierarchical architecture performs a little difference that indicates CNN is applicable to Natural Language Processing (NLP) tasks as well compared to RNN's superiority in NLP.

5.5.2. CNN/Daily Mail Corpus

We also evaluate our DQN model on the joint CNN/Daily Mail corpus in Table 2. The performance of some abstractive and extractive approaches are reported on this dataset. For example, the abstractive models include Nallapati et al'16 [4] and SummaRuNNer-abs [17] and the extractive models include Lead-3 and SummaRuNNer [17], which all perform good results on this corpus using full-length F1 as the metric. Our DQN model uses the same metric as them in order to have a fair comparison with their works. On this joint corpus, our approach performs on par with state-of-the-art model — SummaRuNNer as shown in Table 2. Table 2 also demonstrates the current performance gap between extractive and abstractive approaches to summarization. Apparently, extractive approaches have a superior performance over the abstractive ones since abstractive summarization is a much harder problem.

5.5.3. DUC 2002 Corpus

Table 3 shows the performance comparison of our DQN model with other extractive approaches on the DUC 2002 dataset using Rouge recall with summary length restricted to 75 words. We train the model on the Daily Mail Corpus but we evaluate it on this out-of-domain DUC 2002 corpus. The experimental results indicate our reinforcement learning approach is statistically on par with deep learning based models such as Cheng et al'16 [16], Deep-Selector [50] and SummaRuNNer [17]. However, our DQN model

[1] http://www.cs.toronto.edu/tijmen/csc321/slides/lecture_slides_lec6.pdf

Table 1
Performance of various models on the entire Daily Mail Test set using the limited length recall variants of Rouge with respect to the abstractive ground truth at 75 bytes and 275 bytes.

Approach	75bytes			275bytes		
	Rouge-1	Rouge-2	Rouge-L	Rouge-1	Rouge-2	Rouge-L
Lead-3	21.9	7.2	11.6	40.5	14.9	32.6
LReg	18.5	6.9	10.2	–	–	–
Cheng et al'16	22.7	8.5	12.5	42.2	17.3	34.8
Deep-Select	26.1	10.7	14.3	41.3	15.3	33.5
SummaRuNNer	**26.2**	**10.8**	**14.4**	**42.2**	**16.9**	**34.1**
$DQN_{cnn-rnn}$	23.6	9.8	13.4	41.7	16.4	33.5
$DQN_{rnn-rnn}$	23.8	10.0	13.5	41.9	16.5	33.8

Table 2
Performance comparison of abstractive and extractive models on the entire CNN Daily Mail test set using full-length F1 variants of Rouge.

Approach	Rouge-1	Rouge-2	Rouge-L
Lead-3	39.2	15.7	35.5
Nallapati et al'16	35.4	13.3	32.6
SummaRuNNer-abs	37.5	14.5	33.4
SummaRuNNer	**39.6**	**16.2**	35.3
$DQN_{cnn-rnn}$	39.3	15.8	35.2
$DQN_{rnn-rnn}$	39.4	16.1	**35.6**

Table 3
Experimental results with ROUGE-1, ROUGE-2 and ROUGE-L scores on the DUC 2002 set using the limited length recall variants of Rouge at 75 words.

Approach	Rouge-1	Rouge-2	Rouge-L
Lead-3	43.6	21.0	40.2
LReg	43.8	20.7	40.3
ILP	45.4	21.3	42.8
TGRAPH	48.1	**24.3**	–
URANK	**48.5**	21.5	–
Cheng et al'16	47.4	23.0	**43.5**
Deep-Selector	45.9	21.5	42.4
SummaRuNNer	46.6	23.1	43.0
$DQN_{cnn-rnn}$	45.9	22.3	42.5
$DQN_{rnn-rnn}$	46.4	22.7	42.9

Table 4
Experimental results with ROUGE-1,ROUGE-2 and ROUGE-L scores on DUC2004.

Approach	Rouge-1	Rouge-2	Rouge-L
ILP	34.7	7.5	31.2
ASRL	39.0	9.4	33.7
REAPER	40.3	11.3	36.5
$DQN_{cnn-rnn}$	42.9	13.8	38.8
$DQN_{rnn-rnn}$	**43.4**	**14.1**	**39.2**

preform worse than the state-of-the-art models such as graph-based TGRAPH [51] and URANK [52] algorithms on this corpus. Just like deep learning based supervised models such as SummaRuNNer [17] and Cheng et al. [16], our model usually performs better on the domain datasets which is trained on, but may suffer from domain adaptation issues when evaluated on a different dataset such as DUC 2002. In domain variation issues, graph based unsupervised methods may be more robust.

5.5.4. DUC 2004 Corpus

In order to compare with traditional reinforcement learning approaches — ASRL [18] and REAPER [19], we evaluate our DQN model on another out-of-domain DUC 2004 dataset as presented in Table 4. Our DQN model is trained on the Daily Mail corpus for single-document summarization task but DUC 2004 corpus is a multi-document summarization task, so we concatenate all sentences in a document cluster into a document. The results in Table 4 demonstrate our deep reinforcement learning approach significantly outperforms the traditional reinforcement learning. the potential reason may be that the former is superior to the latter in generalizing informative features from text.

6. Discussion

Compared with deep neural network based classifier models, our extractive summarization approach is based on deep reinforcement learning and it can be directly optimized using the Rouge metrics, eliminating the need for extractive sentence-level labels. Comparatively speaking, the Deep Q-Network is less stable and converge harder than general supervised learning approaches based on deep learning at training time. To speed up training, we use prioritized experience replay memory [54] and double Q-network [29] techniques to train our DQN, which can alleviate data dependency and enhance the stability of training to guarantee convergence faster.

In addition, we employ DQN using two different hierarchical network architectures — CNN-RNN and RNN-RNN. CNN or RNN in bottom layer operates at word level. The advantage of RNN is their ability to implicitly model long term dependencies. But RNN operates sequentially, the output for the next input relies on the previous one and so we cannot parallelize an RNN. CNN has no such problem, each"patch" a convolutional kernel operates on is independent of the other meaning that we can go over the entire input layer concurrently. In our experiments, training $DQN_{rnn-rnn}$ costs about 1.5 times as much time as training $DQN_{cnn-rnn}$.

In our extractive summarization tasks, we use reinforcement learning methods to optimize ROUGE metric because it is not differentiable. Compared with DQN, policy gradient methods [55] seem to be able to do the same work, and it is more widely applied in NLP tasks [56,57] nowadays. However, DQN is more appropriate for our scenario than policy gradient methods. In this task, Each sentence in the source document is regarded as a potential action and DQN directly estimates the future expected reward of each potential action. The action with the max Q-value will be taken as the output to constitute the current summary, and then we estimate the Q-value function of next state based on the information content, the salience and the redundancy between a sentence and the summary. Policy gradient methods are more widely used to generate text sequence in NLP by combining with a pre-trained RNN based encoder-decoder [58] model using maximum likelihood estimation.

On some datasets, the results achieved by the DQN at 275 bytes are slightly worse than that of SummaRuNNer. The potential reason may be that it is hard to update the parameters of DQN only by a scalar guiding signal and the guiding signal lacks more information for text structure. We could use a semi-supervised learning approaches to improve the performance of our model or make the guiding signals more informative to guide the parameters updating. These work will be our future research direction.

Table 5
Example documents and gold summaries from Daily Mail (top) and DUC 2002 (bottom) corpora. The sentences chosen by $DQN_{cnn-rnn}$ and $DQN_{rnn-rnn}$ for extractive summarization are displayed in table respectively.

Source Document: @entity0 have revealed the first gameplay details for their latest golf game @entity0 @entity3 which is due out in july. @entity4 replaced @entity5 as the cover star for @entity8 video games this year. ⋯⋯ firmer, links - style courses like @entity42 will provide more bounce and roll, while softer courses such as @entity18 will have less bounce and be more receptive to spin. @entity0 @entity47 will be available for @entity48 and @entity49 on july 16, 2015 pre-order now at @entity50.

Ground Truth Summary: @entity4 will be on the front cover of @entity0' @entity8 2015 game. the game offers multiple ways to play, including arcade controls. fans can create their own custom gameplay style.

$DQN_{cnn-rnn}$: @entity4 replaced @entity5 as the cover star for @entity8 video games this year. fans can also mix and match all three settings to create their own custom gameplay style. @entity0 have revealed the first gameplay details for their latest golf game @entity0 @entity3 which is due out in july.

$DQN_{rnn-rnn}$: @entity4 replaced @entity5 as the cover star for @entity8 video games this year. fans can also mix and match all three settings to create their own custom gameplay style. the latest version of the long - running game series, due out on july 16, features a number of new features, including various gameplay styles as well as enhanced ball physics.

Source Document: The White House is making sure nobody will accuse it of taking this crisis lightly. In the aftermath of the California earthquake, President Bush and his aides flew into a whirlwind of earthquake-related activity yesterday morning. Some of it was necessary to get federal help flowing to victims, but some seemed designed mostly to project an image of a White House in action. ⋯⋯ Mr. Bush himself essentially acknowledged that he and his aides were trying to head off criticism. On his FEMA visit, Mr. Bush said that he hoped there would be “less carping” about the emergency office’s performance this time, adding that the agency “took a hit” for its reaction to Hurricane Hugo. The White House already is talking of Mr. Bush visiting the California earthquake site this weekend. He visited the Hugo devastation but not until after local leaders urged him to do so.

Ground Truth Summary: Yesterday, in response to the California earthquake, President Bush and his aides flew into a whirlwind of activity that seemed designed to dispel any thought that the White House was not engaged. After the Exxon Valdez oil spill and the devastation of Hurricane Hugo the administration was criticized for its slow response. Bush as much as admitted that the present flurry of activism was intended to counter such criticism, remarking during a visit to the Federal Emergency Management Agency that he hoped there would be “less carping” this time about the Agency’s performance. The President may visit the earthquake site this weekend.

$DQN_{cnn-rnn}$: In the aftermath of the California earthquake, President Bush and his aides flew into a whirlwind of earthquake-related activity yesterday morning. Mr. Bush himself essentially acknowledged that he and his aides were trying to head off criticism. The White House already is talking of Mr. Bush visiting the California earthquake site this weekend.

$DQN_{rnn-rnn}$: In the aftermath of the California earthquake, President Bush and his aides flew into a whirlwind of earthquake-related activity yesterday morning. Mr. Bush himself essentially acknowledged that he and his aides were trying to head off criticism. The White House already is talking of Mr. Bush visiting the California earthquake site this weekend.

We display a couple of example documents from the Daily Mail and DUC corpora and compare the sentences chosen by DQN model with the gold summary in Table 5. Our model has an advantage over long summary generation due to the fact that it models information content, salience and redundancy of sentences based on the current summary content.

7. Conclusion

In this paper, we develop a novel extractive summarization approach based on deep reinforcement learning - a Deep Q-Network(DQN) and explore two different hierarchical network architectures to deploy the DQN. At the same time, the abstract features such as information content, salience and redundancy of sentences is captured in the Q-value approximation. The experimental results show that our deep reinforcement learning approach is comparable to the state-of-the-art deep learning models. What’s more, the performance on RNN-RNN structure is slightly better than CNN-RNN structure. The reason for this situation may be that RNN can better capture global information for text due to long-term dependencies. But CNN can be trained more effectively because it can process the entire input layer concurrently.

Unlike the supervised learning approaches, our extractive summarization system based on deep reinforcement learning can be directly optimized using the Rouge metrics and eliminate the need for extractive sentence-level labels. In the future, we plan to explore a word-level based extractive approach through deep reinforcement learning. Further, we plan to combine extractive and abstractive approaches to construct a joint extractive-abstractive model. What’s more, we also intend to further explore the performance difference between CNN and RNN on different natural language processing tasks.

Acknowledgments

This work was supported by the National Education Information Technology Research (No. 171140001).

References

[1] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, CoRR abs/1409.0473 (2014).
[2] A.M. Rush, S. Chopra, J. Weston, A neural attention model for abstractive sentence summarization, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17–21, 2015, 2015, pp. 379–389.
[3] R. Nallapati, B. Xiang, B. Zhou, Sequence-to-sequence rnns for text summarization, CoRR abs/1602.06023 (2016).
[4] R. Nallapati, B. Zhou, C.N. dos Santos, Ç. Gülçehre, B. Xiang, Abstractive text summarization using sequence-to-sequence rnns and beyond, in: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11–12, 2016, 2016, pp. 280–290.
[5] K.M. Hermann, T. Kociský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, P. Blunsom, Teaching machines to read and comprehend, in: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Quebec, Canada, December 7–12, 2015, 2015, pp. 1693–1701.
[6] J.G. Carbonell, J. Goldstein, The use of mmr, diversity-based reranking for reordering documents and producing summaries, in: SIGIR ’98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia, August 24–28 1998, 1998, pp. 335–336, doi:10.1145/290941.291025.
[7] J.M. Conroy, D.P. O’Leary, Text summarization via hidden markov models, in: SIGIR 2001: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, USA, September 9–13, 2001, 2001, pp. 406–407, doi:10.1145/383952.384042.
[8] G. Erkan, D.R. Radev, Lexpagerank: prestige in multi-document text summarization, in: Conference on Empirical Methods in Natural Language Processing, EMNLP 2004, A Meeting of Sigdat, A Special Interest Group of the Acl, Held in Conjunction with ACL 2004, Barcelona, Spain, 25–26 July 2004, 2004, pp. 365–371.
[9] M.M. Veloso, S. Kambhampati (Eds.), Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9–13, 2005, AAAI Press / The MIT Press, Pittsburgh, Pennsylvania, USA, 2005.
[10] R.T. McDonald, A study of global inference algorithms in multi-document summarization, in: Proceedings of the Advances in Information Retrieval, 29th European Conference on IR Research, ECIR 2007, Rome, Italy, April 2–5, 2007, 2007, pp. 557–564, doi:10.1007/978-3-540-71496-5_51.
[11] K. Woodsend, M. Lapata, Automatic generation of story highlights, in: ACL 2010 - Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, July 11–16, 2010, 2010, pp. 565–574.
[12] Mikael, O. Mogren, N. Tahmasebi, D. Dubhashi, Extractive summarization using continuous vector space models, in: Cvsc at Eacl, 2014.

[13] R. Socher, E.H. Huang, J. Pennington, A.Y. Ng, C.D. Manning, Dynamic pooling and unfolding recursive autoencoders for paraphrase detection, in: Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12–14 December 2011, Granada, Spain., 2011, pp. 801–809.
[14] K. Ganesan, C. Zhai, J. Han, Opinosis: A graph based approach to abstractive summarization of highly redundant opinions, in: COLING 2010 - Proceedings of the Conference on 23rd International Conference on Computational Linguistics, Beijing, China, 23–27 August 2010, 2010, pp. 340–348.
[15] W. Yin, Y. Pei, Optimizing sentence modeling and selection for document summarization, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25–31, 2015, 2015, pp. 1383–1389.
[16] J. Cheng, M. Lapata, Neural summarization by extracting sentences and words, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Volume 1: Long Papers, Berlin, Germany, August 7–12, 2016, 2016.
[17] R. Nallapati, F. Zhai, B. Zhou, Summarunner: a recurrent neural network based sequence model for extractive summarization of documents, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA. February 4–9, 2017, 2017, pp. 3075–3081.
[18] S. Ryang, T. Abekawa, Framework of automatic text summarization using reinforcement learning, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, Jeju Island, Korea, July 12–14, 2012, 2012, pp. 256–265.
[19] C. Rioux, S.A. Hasan, Y. Chali, Fear the REAPER: a system for automatic multi--document summarization with reinforcement learning, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, A meeting of SIGDAT, a Special Interest Group of the ACL, Doha, Qatar, October 25–29, 2014, 2014, pp. 681–690.
[20] S. Chopra, M. Auli, A.M. Rush, Abstractive sentence summarization with attentive recurrent neural networks, in: NAACL HLT 2016 - The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12–17, 2016, 2016, pp. 93–98.
[21] C. Flick, Rouge: a package for automatic evaluation of summaries, in: The Workshop on Text Summarization Branches Out, 2004, p. 10.
[22] C. Amato, G. Shani, High-level reinforcement learning in strategy games, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Volume 1–3, Toronto, Canada, May 10–14, 2010, 2010, pp. 75–82, doi:10.1145/1838206.1838217.
[23] D. Silver, R.S. Sutton, M. Müller, Reinforcement learning of local shape in the game of go, in: IJCAI 2007 - Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6–12, 2007, 2007, pp. 1053–1058.
[24] W. Gao, Y. Jiang, Z. Jiang, T. Chai, Output-feedback adaptive optimal control of interconnected systems based on robust adaptive dynamic programming, Automatica 72 (2016) 37–45, doi:10.1016/j.automatica.2016.05.008.
[25] W. Gao, Z. Jiang, Adaptive dynamic programming and adaptive optimal output regulation of linear systems, IEEE Trans. Automat. Contr. 61 (12) (2016) 4164–4169, doi:10.1109/TAC.2016.2548662.
[26] C.J.C.H. Watkins, P. Dayan, Q -learning, Mach. Learn. 8 (3–4) (1992) 279–292.
[27] S.R.K. Branavan, D. Silver, R. Barzilay, Learning to win by reading manuals in a monte-carlo framework, in: Proceedings of the Conference on 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, 19–24 June, 2011, 2011, pp. 268–277.
[28] R. Sutton, A. Barto, Reinforcement Learning: An Introduction, Adaptive Computation and Machine Learning Series, IEEE Transactions on Neural Networks, 1998.
[29] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M.A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533, doi:10.1038/nature14236.
[30] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, Studies in Computational Intelligence, 385, Springer, 2012, doi:10.1007/978-3-642-24797-2.
[31] J. Chung, Ç. Gülçehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, CoRR abs/1412.3555 (2014).
[32] N. Kalchbrenner, P. Blunsom, Recurrent convolutional neural networks for discourse compositionality, CoRR abs/1306.3584 (2013).
[33] X. Zhang, M. Lapata, Chinese poetry generation with recurrent neural networks, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp. 670–680.
[34] Y. Kim, Y. Jernite, D. Sontag, A.M. Rush, Character-aware neural language models, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA. February 12–17, 2016, 2016, pp. 2741–2749.
[35] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp. 1746–1751.
[36] R.K. Srivastava, K. Greff, J. Schmidhuber, Training very deep networks, in: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Quebec, Canada, December 7–12, 2015, 2015, pp. 2377–2385.
[37] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M.A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533, doi:10.1038/nature14236.
[38] J. Oh, X. Guo, H. Lee, R.L. Lewis, S.P. Singh, Action-conditional video prediction using deep networks in atari games, in: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, Quebec, Canada, December 7–12, 2015, 2015, pp. 2863–2871.
[39] L.J. Lin, Reinforcement learning for robots using neural networks (1993).
[40] L.P. Kaelbling, M.L. Littman, A.W. Moore, An introduction to reinforcement learning, Mach. Learn. 8 (3–4) (1995) 225–227.
[41] R. Mihalcea, P. Tarau, Textrank: Bringing order into texts, Unt Scholarly Works (2004) 404–411.
[42] D. Shen, J. Sun, H. Li, Q. Yang, Z. Chen, Document summarization using conditional random fields, in: IJCAI 2007 - Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6–12, 2007, 2007, pp. 2862–2867.
[43] K.M. Svore, L. Vanderwende, C.J.C. Burges, Enhancing single-document summarization by combining ranknet and third-party sources, in: EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic, June 28–30, 2007, 2007, pp. 448–457.
[44] Z. Cao, C. Chen, W. Li, S. Li, F. Wei, M. Zhou, Tgsum: Build tweet guided multi--document summarization dataset, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA. February 12–17, 2016, 2016, pp. 2906–2912.
[45] M.J. Hausknecht, P. Stone, Deep recurrent q-learning for partially observable mdps, CoRR abs/1507.06527 (2015).
[46] J. Peng, R.J. Williams, Incremental multi-step q-learning, Mach. Learn. 22 (1–3) (1994) 226–232.
[47] S.R.K. Branavan, D. Silver, R. Barzilay, Learning to win by reading manuals in a monte-carlo framework, J. Artif. Intell. Res. 43 (2012) 661–704, doi:10.1613/jair.3484.
[48] J. Eisenstein, J. Clarke, D. Goldwasser, D. Roth, Reading to learn: Constructing features from semantic abstracts, in: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6–7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL, 2009, pp. 958–967.
[49] K. Narasimhan, T.D. Kulkarni, R. Barzilay, Language understanding for text-based games using deep reinforcement learning, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17–21, 2015, 2015, pp. 1–11.
[50] R. Nallapati, B. Zhou, M. Ma, Classify or select: neural architectures for extractive document summarization, CoRR abs/1611.04244 (2016).
[51] D. Parveen, H. Ramsl, M. Strube, Topical coherence for graph-based extractive summarization, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17–21, 2015, 2015, pp. 1949–1954.
[52] X. Wan, Towards a unified approach to simultaneous single-document and multi-document summarizations, in: Proceedings of the 23rd international conference on computational linguistics. Association for computational Linguistics, 2010, pp. 1137–1145.
[53] J. Pennington, R. Socher, C.D. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25–29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, 2014, pp. 1532–1543.
[54] T. Schaul, J. Quan, I. Antonoglou, D. Silver, Prioritized experience replay, CoRR abs/1511.05952 (2015).
[55] S.A. Solla, T.K. Leen, K. Müller (Eds.), Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999], The MIT Press, 2000.
[56] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, K. Murphy, Optimization of image description metrics using policy gradient methods, CoRR abs/1612.00370 (2016).
[57] J. Su, X. Carreras, K. Duh (Eds.), Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, November 1–4, 2016, The Association for Computational Linguistics, Austin, Texas, USA, 2016.
[58] Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, Montreal, Quebec, Canada, December 8–13 2014, 2014.

Kaichun Yao, is a Ph.D. candidate in School of Computer and control Engineering, University of Chinese Academy of Sciences, Beijing, China. His research interests include natural language processing, knowledge graph, deep learning and reinforcement learning.

Libo Zhang, received the PhD. degree in Computer Software and Theory from University of Chinese Academy of Sciences. Now He is an assistant research professor in the Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include pattern recognition, knowledge graph and deep learning.

Tiejian Luo, is currently a professor in School of Computer and control Engineering, University of Chinese Academy of Sciences (UCAS), Beijing, China. He is also the Director of the Information Dynamic and Engineering Applications Laboratory in UCAS. His main research topics include web mining, large scale web performance optimization and distributed storage systems.

Yanjun Wu, received his PhD. degree in computer software and theory from Institute of Software, Chinese Academy of Sciences, Beijing, in 2006. Currently, he is a research professor at Institute of Software, Chinese Academy of Sciences, Beijing. His research interests include operating system and artificial intelligence.