Towards the Use of Deep Reinforcement Learning with Global Policy For Query-based Extractive Summarisation*

Diego Mollá
Department of Computing
Macquarie University
Sydney
diego.molla-aliod@mq.edu.au

Abstract

Supervised approaches for text summarisation suffer from the problem of mismatch between the target labels/scores of individual sentences and the evaluation score of the final summary. Reinforcement learning can solve this problem by providing a learning mechanism that uses the score of the final summary as a guide to determine the decisions made at the time of selection of each sentence. In this paper we present a proof-of-concept approach that applies a policy-gradient algorithm to learn a stochastic policy using an undiscounted reward. The method has been applied to a policy consisting of a simple neural network and simple features. The resulting deep reinforcement learning system is able to learn a global policy and obtain encouraging results.

1 Introduction

Common supervised machine learning approaches to extractive summarisation attempt to label individual text extracts (usually sentences or phrases; in this paper we will use sentences). In a subsequent stage, a summary is generated based on the predicted labels of the individual sentences and other factors such as redundancy of information.

The process of obtaining the annotated data can be complex. Data sets often contain complete summaries written manually. Well-known examples of data sets of this type are the DUC and TAC data sets (Dang, 2006, 2008). In such cases the task of labelling individual sentences is not straightforward and needs to be derived from the full summaries. Alternatively, annotations can be obtained through highlights made by the annotators (Woodsend and Lapata, 2010, for example).

Regardless of the means used to annotate individual sentences, the final evaluation of the system compares the output summary with a set of target summaries, either by using human judges or automatically by using packages such as ROUGE (Lin, 2004). However, machine learning approaches designed to minimise the prediction error of individual sentences would not necessarily minimise the prediction error of the final summary evaluation metric.

In this paper we propose a proof-of-concept method that uses reinforcement learning with global policy as a means to use the ROUGE_L evaluation of the final summary directly in the training process. Section 2 introduces reinforcement learning and mentions past work on the use of reinforcement learning for summarisation. Section 3 describes our proposal for the use of reinforcement learning for query-based summarisation. Section 4 presents the results of our experiments, and Section 5 concludes this paper.

2 Reinforcement Learning

Reinforcement Learning (RL) is a machine learning approach that is designed to train systems that aim to maximise a long-term goal, even when there is no knowledge (or little knowledge) of the impact of the individual decisions that are made to achieve the goal. A RL task (Figure 1) consists of an environment that can be observed and can be acted on, and an agent that makes a sequence of actions. The effect of undertaking an action (a) on the environment will result in an observed state (s) and a reward (r). The agent then needs to learn the sequence of actions that maximises the cumulative reward.

The task of query-based summarisation can be

* Code available at https://github.com/dmollaaliod/alta2017-rl

Diego Mollá-Alliod. 2017. Towards the Use of Deep Reinforcement Learning with Global Policy For Query-based Extractive Summarisation. In *Proceedings of Australasian Language Technology Association Workshop*, pages 103–107.

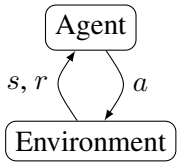

Figure 1: The reinforcement learning process.

reduced to a RL task by assigning null reward $r = 0$ to the decision of selecting each individual sentence or not, until the point at which a final summary has been extracted. At the moment that a final summary has been extracted, the reward r is the actual evaluation score of the full summary. The RL approach should learn a policy π such that the agent can determine how the individual decisions made at the time of selecting (or not) a sentence would impact on the evaluation score of the final summary.

Ryang and Abekawa (2012) and Rioux and Hasan (2014) propose the learning of a local policy π that is specific to each summary. For this purpose, the reward r of the entire summary is calculated based on measures of similarity between the summary and the source document. Thus, Ryang and Abekawa (2012) uses information such as coverage, redundancy, length and position. Rioux and Hasan (2014) uses a reward system that is more similar to the ROUGE set of metrics, but again using only information from the source text and the generated summary. Effectively, these approaches use RL as a means to search the space of possible selections of sentences by training a local policy that needs to be re-trained each time a new summary needs to be generated.

Ryang and Abekawa (2012) mentions the possibility of training a global policy in the section of further work provided that there is a mean to provide a feature representation of a summary. In this paper we show a simple way to represent the state of the environment, including the summary, such that the system can train a global policy. We use a training set annotated with target summaries to train a global policy that uses the direct ROUGE_L score as the reward. Once a global policy has been learnt, it is applied to unseen text for evaluation. By using a global policy instead of a local policy, the system can use the direct ROUGE_L score instead of an approximation, and the computational cost shifts to the training stage, enabling a faster generation of summaries after the system has been trained.

There is also research that use other mechanisms in order to train a summarisation system using the direct ROUGE score (Aker et al., 2010) or an approximation (Peyrard and Eckle-Kohler, 2016).

3 Reinforcement Learning for Query-based Extractive Summarisation

This section describes our proposal for the adaptation of query-based summarisation to RL with global policy.

3.1 Environment

After applying a decision whether sentence i is to be selected as a summary or not, the environment records the decision and issues a reward $r = 0$. After all decisions have been made, the environment builds the summary by concatenating all selected sentences in linear order. Then, the environment returns the ROUGE_L score of the summary as the reward. More formally, and assuming that the total number of sentences in the input text is n, the reward is computed as follows:

$$r = \begin{cases} 0 & \text{if} \quad i < n \\ \text{ROUGE\_L} & \text{if} \quad i = n \end{cases}$$

This process is inspired in Ryang and Abekawa (2012)'s framework, the difference being that, in our work, the reward returned when $i = n$ is the actual ROUGE_L score of the summary instead of an approximation.

For the purposes of this paper, the environment is implemented as an object `env` that allows the following operations:

- $s \leftarrow$ `env.reset(sample)`: reset to sample `sample` and return an initial state s.
- $s, r,$ `done` $\leftarrow$ `env.step`(a): perform action a and return state s, reward r, and a Boolean value $True$ if all input sentences have been processed.

3.2 Action Space

At each step of the RL process, the agent will decide whether a particular sentence is to be selected (1) or not (0).

3.3 State

The RL framework is greedy in the sense that, once a decision is made about sentence i, it cannot be undone. The agent should therefore have

the information necessary to make the right decision, including information about what sentences are yet to process. Since the agent uses a global policy, the state should be able to encode information about any number of input sentences, and any number of remaining sentences. We resolved this by building vectors that represent sequences of sentences. In this paper we use *tf.idf*, but other methods could be used, such as sentence embeddings learnt by training deep neural networks.

In concrete, the environment provides the following state:

1. *tf.idf* of the candidate sentence i.
2. *tf.idf* of the entire input text to summarise.
3. *tf.idf* of the summary generated so far.
4. *tf.idf* of the candidate sentences that are yet to be processed.
5. *tf.idf* of the question.

Information 2. and 3. would be useful to determine whether the current summary is representative of the input text. Information 4. would be useful to determine whether there is still important information that could be added to the summary in future steps. The agent could then, in principle, contrast 1. with 2., 3., 4. and 5. to determine whether sentence i should be selected or not.

3.4 Global Policy

The global policy is implemented as a neural network that predicts the probability of each action a available in the action space $\{0, 1\}$. In practice, the system only needs to predict $Pr(a = 0)$. As a proof of concept, the neural network implemented in this paper is simply a multi-layer network with one hidden layer that uses a relu activation, and the output unit is a Bernoulli logistic unit. Thus, given a state s formed by concatenating all the items listed in Section 3.3, the network predicts $Pr(a = 0)$ as follows.

$$\begin{aligned} Pr(a = 0) &= \sigma(h \cdot W_h + b_h) \\ h &= \max(0, s \cdot W_s + b_s) \end{aligned}$$

In our experiments, the size of the hidden layer is 200.

3.5 Learning Algorithm

The learning algorithm for the global policy is a variant of the REINFORCE algorithm (Williams, 1992) that uses gradient descent with cross-entropy gradients that are multiplied with the reward (Géron, 2017, Chapter 16). This is shown in Algorithm 1.

Data: `train_data`
Result: θ
`sample` $\sim Uniform($`train_data`$)$;
$s \leftarrow$ `env.reset(sample)`;
`all_gradients` $\leftarrow \emptyset$;
`episode` $\leftarrow 0$;
while *True* **do**
  $\xi \sim Bernoulli\left(\frac{Pr(a=0)+p}{1+2\times p}\right)$;
  $y \leftarrow 1 - \xi$;
  `gradient` $\leftarrow \frac{\nabla(\text{cross\_entropy}(y, Pr(a=0))}{\nabla\theta}$;
  `all_gradients.append(gradient)`;
  $s, r, done \leftarrow$ `env.step`(ξ);
  `episode` $\leftarrow$ `episode` $+ 1$;
  if *done* **then**
    $\theta \leftarrow$
     $\theta - \alpha \times r \times$ mean(`all_gradients`);
    `sample` $\sim Uniform($`train_data`$)$;
    $s \leftarrow$ `env.reset(sample)`;
    `all_gradients` $\leftarrow \emptyset$;
  end
end

Algorithm 1: Training by Policy Gradient, where $\theta = (W_h, b_h, W_s, b_s)$.

In Algorithm 1, the neural net predicts $Pr(a = 0)$. The action chosen during training is sampled from a Bernoulli distribution with probability $Pr(a = 0)$ that has a perturbation p, such that p slowly decreases at each training episode. By adding this perturbation the system explores the two possible actions in the early stages of training and delays locking in possible local minima. In our implementation, p is computed with an initial value of 0.2 and decreasing using the formula:

$$p = 0.2 \times 3000/(3000 + \text{episode})$$

Thus, $p = 0.1$ after 3000 episodes, and so on.

When a full summary has been produced, the mean of all cross-entropy gradients used in all the steps that lead to the summary is computed and multiplied by the summary reward to update the neural network trainable parameters. Using

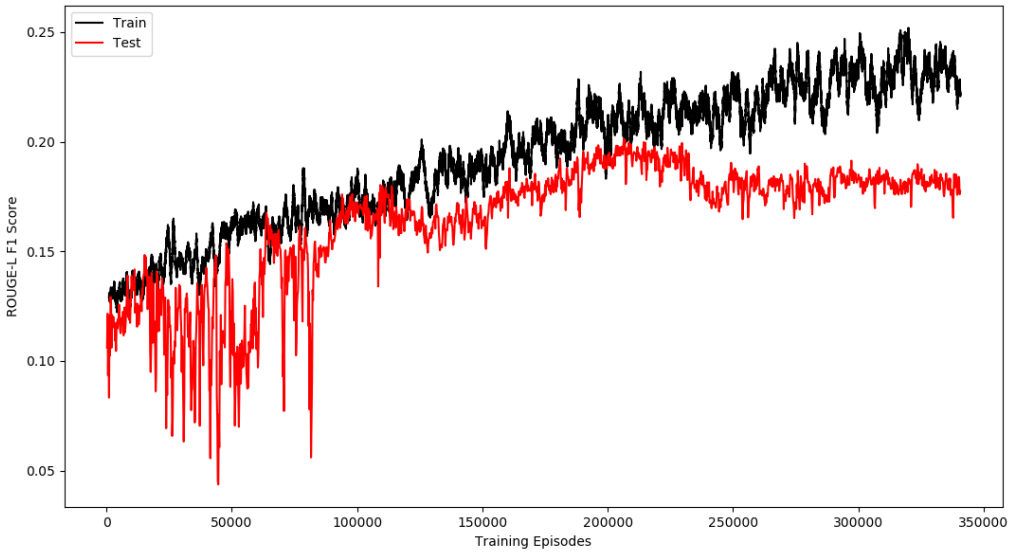

Figure 2: Results of the system. The results of training (black line) are the average ROUGE_L of the last 1000 chosen training samples at every point. The results of testing (red line) are the average ROUGE_L of the test set.

RL terminology, the method uses undiscounted reward.

At run time, the action a chosen is simply the action a with highest probability.

4 Experiments and Results

We have used the data provided by BioASQ 5b Phase B (Tsatsaronis et al., 2015). The dataset has 1799 questions together with input text and ideal answers. These ideal answers form the target summaries. We have split the data into a training an a test set.

Algorithm 1 updates the parameters θ by applying standard gradient descent. In our experiments, we have used the Adam optimiser instead, which has been shown to converge rapidly in many applications (Kingma and Ba, 2015). Also, due to computing limitations, our implementation only processes the first 30 sentences of the input text.

Figure 2 shows the progress of training and evaluation. We can observe that the neural net learns a global policy that improves the ROUGE_L results of the training data (black line). More importantly, it also improves the ROUGE_L results when presented with the test data (red line). It appears that the system starts overfitting after about 200,000 training steps.

Considering that the state does not have direct information about the sentence position or the length of the summary, and given the relatively small training data, these results are encouraging. It is well known that sentence position carries important information for the task of summarisation. Also, preliminary experiments adding summary length to the state showed quicker convergence to better values. In this paper we chose not to incorporate any of this information to test the capabilities of the use of reinforcement learning.

5 Conclusions

We have presented a reinforcement learning approach that learns a global policy for the task of query-based summarisation. Our experiments used fairly simple features to represent the state of the environment. Also, the neural network implemented to model the global policy is fairly simple. Yet, the system was able to effectively learn a global policy. In further work we will explore the use of more sophisticated features such as word or sentence embeddings, and more sophisticated neural networks.

Further work will also explore the use of variants of reinforcement learning algorithms in order to speed up the learning process.

References

Ahmet Aker, Trevor Cohn, and Robert Gaizauskas. 2010. Multi-document summarization using A* search and discriminative training. In *EMNLP 2010*. October, pages 482–491.

Hoa Trang Dang. 2006. Overview of DUC 2006. In *Proc. Document Understanding Workshop*. NIST.

Hoa Trang Dang. 2008. Overview of the TAC 2008 opinion question answering and summarization tasks. In *Proc. TAC 2008*.

Aurélien Géron. 2017. *Hands-on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR 2015*. pages 1–15.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *ACL Workshop on Tech Summarisation Branches Out*.

Maxime Peyrard and Judith Eckle-Kohler. 2016. Optimizing an approximation of ROUGE — a problem-reduction approach to extractive multi-document summarization. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. pages 1825–1836.

Cody Rioux and Sadid A Hasan. 2014. Fear the REAPER: A system for automatic multi-document summarization with reinforcement learning. In *EMNLP 2014*. 2010, pages 681–690.

Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In *EMNLP 2012*. July, pages 256–265.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artiéres, Axel-Cyrille Ngonga Ngomo, Norman Heino, Eric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics* 16(1):138.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8:229–256.

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