Extractive Text Summarization with Q-learning and Text Rank

Niket Kathiriya
Computer Science department
University of Massachusetts, Lowell
Niket_Kathiriya@student.uml.edu

Ashwin Sankarasubramanian
Computer Science department
University of Massachusetts, Lowell
ashwin_sankarasubramanian@student.uml.edu

Purvang Shah
Computer Science department
University of Massachusetts, Lowell
purvangvipulbhai_shah1@student.uml.edu

***Abstract*—In this document, we first begin by introducing text summarization and its two types followed by related work done in the similar task by other researchers. Then, we propose two different approaches for the extractive text summarization problem. (1)Q-learning and (2)Text rank. While text rank algorithm is widely known for the extractive text summarization task, the approach to use q-learning for the same has only been tried in a very few experiments. We define the problem in such a way that it can be fit into the q-learning parameters. Then we have shown the ROUGE scores for both methods on the 'CNN-DailyMail News Text Summarization' data set in the experimental results section.**

***Index Terms*—Extractive Text Summarization, Text rank, Q-learning, Reinforcement Learning**

I. Introduction

People are obsessed with emerging technologies and social media in today's world. Everyone in today's generation uses their phone even if they only have a few seconds to spare. The majority of consumers prefer quick, concise news than extensive news on television or in the newspaper. As a result, developing a system that extracts summary from a given text or document without affecting the content's meaning is critical.

Abstractive Text Summarization and Extractive Text Summarization are two approaches for extracting summary from a given text. The system provides a short summary without affecting the meaning of the material from the given text in Abstractive Text Summarization. In the case of Extractive Text Summarization, the system extracts a few key sentences from the given text that make the most sense of the whole.

Text summarization is difficult because, when we humans summarize a piece of text, we normally read it completely to have a thorough comprehension of it before writing a summary highlighting its important points. Because computers lack human understanding and linguistic skills, text summarization is a difficult and time-consuming task.

Using a Q-network, we present a reinforcement learning framework for extractive text summarization which will generate a candidate summary with a pre-defined k number of sentences.

II. Related Work

A. Extractive Text Summarization Using Deep Learning

This paper [2], proposed in 2018 suggests the use of a lateral combination of a deep neural network and a Fuzzy logic system. The suggested neural network type is the Restricted Boltzmann Machine (RBM). The sentences are independently fed into the neural network and fuzzy logic system to produce two different summaries. After performing a set union operation, we produce a final summary for the document.

B. Hybrid auto text summarization using deep neural network and fuzzy logic system

This paper [3] proposes a very similar architecture to approach the problem. They also make use of fuzzy logic systems and RBM. The key difference consists of a sequential processing rather than lateral. So the sentences are first classified by the fuzzy rules before being processed by the network. They also make use of distinct sentence features such as title similarity and named entity or numerical data sentence weights to produce a more accurate summary.

C. Extractive Text Summarization from Web pages using Selenium and TF-IDF algorithm

This paper [4] provides an approach to the data scraping process rather than the processing itself. The key concept is to input a search query rather than a whole input document. The query is then searched using one of the common web search engines. The top web page results of the search engines are retrieved and scrapped together to form the input document. This document is then summarized to produce the short summary of available web information on the search query.

D. Extractive Text Summarization Using Word Vector Embedding

This paper [5] introduces the use of pre-trained GLoVe vectors to form sentence representations. These pre-trained vectors are trained in an unsupervised manner over thousands or millions of sentences to produce billions of word tokens with varying dimensions. These word vectors are processed using fully connected multi-layer perceptron models to predict the probability of a sentence being included in the summary. This approach had better performance than all previous existing online summarization tools.

III. Proposed Approach

We have presented two approaches for tackling the problem of automatic text summarization: Q-Learning and Text Rank.

Q-Learning is a reinforcement learning algorithm and Text Rank is a graph based ranking algorithm.

A. Q-Learning

1) Introduction: While neural networks have recently made advancements in areas such as computer vision, machine translation, and time series prediction, they can also be combined with reinforcement learning techniques to create something truly remarkable, such as AlphaGo.

Reinforcement learning refers to goal-oriented algorithms that learn how to achieve a complex aim (goal) or maximize along a certain dimension over a number of steps, such as maximizing the points earned in a game over a number of moves. They can start from scratch and achieve superhuman performance given the appropriate circumstances. These algorithms are penalized when they make the incorrect judgments and rewarded when they make the right ones, similar to a toddler who is rewarded with spankings and candy.

Deep learning-based reinforcement learning algorithms have beaten world champions in the game of Go as well as human professionals in a variety of Atari video games. While this may appear insignificant, it represents a significant improvement over their past achievements, and the state of the art is rapidly evolving.

The tough problem of linking current actions with the delayed returns they provide is solved through reinforcement learning. Reinforcement learning algorithms, like people, must sometimes wait a long period to see the results of their decisions. They work in a delayed-response environment, which makes it impossible to know which action leads to which result over a long period of time.

Reinforcement learning algorithms are supposed to perform better and better in uncertain, real-life situations where they can choose from an arbitrary number of alternative actions rather than the limited options of a video game. That is, over time, we expect them to be useful in achieving real-world objectives.

Deep-Q learning and A3C are two reinforcement learning algorithms that have been implemented in the RL4J Deeplearning4j package. It's already capable of playing Doom.

2) Definitions: The concepts of agents, environments, states, actions, and rewards, all of which are fully explained below, can be used to understand reinforcement learning. Capital letters tend to denote sets of things, and lower-case letters denote a specific instance of that thing.

- Agent: A drone making a delivery or Super Mario navigating a video game are examples of agents. The agent is the algorithm. You are the agent in your own life.
- Action(A): A is the collection of all conceivable motions that the agent can make. Although an action is almost self-explanatory, it should be noted that agents select from a set of options. Running right or left, jumping high or low, crouching or standing still are all possible actions in video games. The list in the stock markets could involve purchasing, selling, or holding any of a variety of assets and derivatives. Many various velocities and accelerations in 3D space would be actions while operating airborne drones.
- Discount factor: To reduce the effect of future benefits on the agent's choice of action, the discount factor is multiplied by future rewards as discovered by the agent. Why? Its purpose is to make future rewards less valuable than immediate rewards, so instilling in the agent a sense of short-term hedonism. The lower-case Greek letter gamma(γ) is frequently used to signify this: If γ is 0.8, and a reward of 10 points is offered after three time steps, the present value of the reward is 0.8 x 10. Future incentives would be valued the same as immediate prizes if the discount factor was set to one. Here, we're fighting against the concept of delayed gratification.
- Environment: The world in which the agent travels. The environment receives the agent's current state and action as input and returns the reward and the agent's next state as output. If you're the agent, the environment could be the physical laws and social regulations that process your activities and decide their outcomes.
- State(S): A state is a precise location and time in which the agent finds itself; it is an instantaneous configuration that places the agent in relation to other significant things such as tools, barriers, opponents, or prizes. It could be the current state, or any future situation, as determined by the environment. Have you ever found yourself in an awkward situation? That is a state.
- Reward(R): A reward is a form of feedback that allows us to assess whether an agent's actions were successful or unsuccessful. When Mario, for example, touches a coin in a video game, he earns points. An agent sends output in the form of actions to the environment from any given state, and the environment returns the agent's new state (which is the result of acting on the previous state) as well as any rewards, if any. Rewards might be given right away or over time. They effectively assess the actions of the actors.
- Policy: The policy is the method by which the agent decides what to do next based on the present situation. It connects states to behaviours, focusing on the ones with the greatest potential for reward.
- Value(V): In contrast to the short-term reward R, the predicted long-term return with discount is expected. The predicted long-term return of the existing state under policy is defined as V(s). The further into the future rewards materialize, the more we discount them, or lower their anticipated value.
- Q-value(Q): Q-value is similar to value, with the exception that it requires an additional parameter, the current action, a. The long-term return of the existing state s, performing action an under policy is referred to as Q(s, a). Q assigns rewards to state-action pairings.
- Trajectory: A series of states and actions that have an impact on those states.

3) The Learning Process: Agents are functions that turn the new state and reward into the next action, while environments are functions that transform the new state and reward into the next action. We can figure out how the agents work, but we can't figure out how the environment works. It's a black box with only the inputs and outputs visible. It's similar to how most people feel about technology: we understand what it does but not how it works. Reinforcement learning is an agent's attempt to approximate the function of the environment so that we can send behaviors into the black-box environment that maximize the rewards it produces. Unlike other types of machine learning, such as supervised and unsupervised learning, reinforcement learning can only be thought of in terms of state-action pairings that follow one another.

Reinforcement learning evaluates actions based on the outcomes. It is goal-oriented, with the goal of learning action sequences that will lead an agent to attain their goal or maximize their objective function. Some instances are as follows:

- In video games, the goal is to gather as many points as possible, each additional point acquired during the game may influence the agent's subsequent behavior; for example, the agent may learn that blasting battleships, touching coins, or evading meteors will boost its score.
- In the real world, a robot's goal might be to get from point A to point B, and every inch the robot gets closer to point B could be counted as a point.

4) Concept of Reward and Discount: Why is it that the agent's purpose is to maximize the total expected reward? Reinforcement Learning, on the other hand, is based on the reward theory. The maximization of the expected cumulative reward can be applied to any goal.

The cumulative reward at each time step t can be written as:

$$G_t = R_{t+1} + R_{t+2} + ... \tag{1}$$

which is equivalent to:

$$G_t = \sum_{k=0}^{T} R_{t+k+1} \tag{2}$$

However, we can't just add the incentives like that in reality. Prizes that appear sooner (at the start of the game) are more likely to occur since they are more predictable than long-term prospective rewards.

To discount the rewards, we proceed as follows: Gamma is a discount rate that we define. It has to be a number between 0 and 1.

- The smaller the discount, the higher the gamma. This indicates that the learning agent is more concerned with the long-term reward.
- The larger the discount, on the other side, the smaller the gamma.This indicates that our agent is more concerned with the short-term gain.

Our discounted total expected benefits are as follows:

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{3}$$

where $\gamma \in [0,1)$

5) Exploration and Exploitation: Exploration is the process of learning more about the environment. Exploitation is the process of making the most of known knowledge in order to maximize the reward. Remember that our RL agent's purpose is to maximize the total predicted reward. We can, however, slip into a frequent trap. It is possible that if we look at the immediate reward without exploiting the environment, we can end up with small cumulative reward. The exploration/exploitation trade-off is a term used to describe this situation.

6) Q-Learning: Assume a robot must navigate a maze to reach its destination. Mines are present, and the robot is limited to moving one tile at a time. If the robot steps on a mine, it will be killed. The robot must arrive at the destination in the minimum amount of time.

The scoring/reward system is as below:

- The robot loses 1 point at each step. This is done so that the robot takes the shortest path and reaches the goal as fast as possible.
- If the robot steps on a mine, the point loss is 100 and the game ends.
- If the robot gets power, it gains 1 point.
- If the robot reaches the end goal, the robot gets 100 points.

The obvious challenge now is: how can we teach a robot to take the shortest path to the ultimate objective without stepping on a mine?

- The Q-Table The Q-Table is simply a fancy term for a simple lookup table in which we calculate the maximum projected future rewards for each state's behavior. In essence, this table will direct us to the optimum course of action in each state. The actions are the columns in the Q-Table, and the states are the rows. Each Q-table score represents the robot's maximum projected future reward if it does that action in that condition. We must improve the Q-Table at each iteration, so this is an iterative process.
- The Q-function The Q-function uses the Bellman equation and takes two inputs: state (s) and action (a).

$$Q^{\pi}(s_t, a_t) = E[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ...][s_t, a_t] \tag{4}$$

Using the above equation, we get the values of Q for the cells in the table. When we start, all the values in the Q-table are zeros.

There is an iterative process of updating the values. As we start to explore the environment, the Q-function gives us better and better approximations by continuously updating the Q-values in the table. Now, lets understand how the updating takes place.

- The Q-Learning Algorithm

Algorithm 1 The Q-Learning Algorithm

Initialize **Q(s,a)** arbitrarily
Repeat (for each episode):
  Initialize **S**
  Repeat (for each step of episode):
    Choose **a** from **S** using policy derived from **Q**
    Take action **a**, observe **r, S'**
    Q(s,a) $\leftarrow$ **Q(s,a) +** α**[r +** γ **max**αQ(s',a')]
  s $\leftarrow$ **s'**

B. Text Rank

1) Introduction: Text ranking is a graph based ranking algorithm that can be used to determine the sentences which can best construct our document summary. The algorithm has been used by Google and other search engines to produce the most relevant search result for the user search query. This is done by scoring the different search results with a similarity measure with respect to the search query and choosing the best scores. The chosen web pages with the best similarity measure scores would consist of the most relevant search results for the user.

2) Methodology: The algorithm works by constructing a graph with the sentences as vertices. An edge or connection between two vertices would represent a similarity between them. The higher the number of connections, the more similar a sentence is to the rest of the whole document. Therefore, the vertices with the highest degrees would be the highest ranking sentences for the text ranking algorithm. The corresponding sentences for these vertices would be used to construct the most relevant summary for the input document.

3) GLoVe Vectors: Another important concept that we make use of in the text ranking implementation is the usage of GLoVe vectors. GLoVe vectors are word vectors that have been formulated through unsupervised learning on vocabularies ranging from four hundred thousand words to one million two hundred thousand words. These pre-trained vectors form a universal representation of the words which can be used in our text ranking algorithm implementation.

The main advantage of using GLoVe vectors is to overcome the incapability of the model to understand the similarity in meaning and context between two words with very different vector representation values. For instance if we have a sentence consisting of the word emperor, the corresponding word vector will be tokenized and provided to the model as inputs. Another tokenized sentence consisting of the word king, would have a very different value from the previous sentence. This is because there are no common characters between the words king and emperor. But these two words are very similar in meaning and can be used interchangeably. The model would be unable to understand the similarity between these word vectors and would therefore consider them as two non-related entities.

By using GLoVe vectors, which have been pre-trained in an unsupervised manner, the clusters of words with similar meanings would be formulated into the similar word vectors. These similar vectors can then be used by the model to understand the contextual or meaning similarities between these two synonyms.

IV. Implementation

A. Q - Learning

For implementation of the solution with q-learning approach, we assigned the attributes as below:

- Environment = input text
- States = Each possible combination of the sentences resulting into the candidate summary
- Actions = Selection of a sentence to be added to or removed from the candidate summary
- Reward = (k * Similarity(C,T)) - ((1-k) * Redundancy(C)); where Similarity(C,T) refers to the similarity between the candidate summary and the input text sentences. Redundancy(C) refers to the similarity between sentences within the candidate summary. Both are measured with cosine similarity metric. k is the trade-off factor (0.9 in our experiments)
- Discount factor (gamma) = 0.5

For the number of iterations in exploration phase, we provided a threshold in order to make the program terminate within reasonable time; If the number of sentences in the input text are less than 15, then the exploration phase will explore $2^{no.of sentences}$ candidate summaries but if the number of sentences are more than 15, then the exploration phase will only explore 2^{15} candidate summaries. This impacts the performance negatively, but if the threshold is not applied, then the program does not converge for large text inputs.

B. Text Rank

The data is initially pre-processed by using two modules. First we convert the words in the input document to their respective GLoVe word vector representations. We use the glove.6B.100D pre-trained word vectors which have been trained on a vocabulary size of four hundred thousand words. There are six billion different word tokens available in this file and the word representations consist of hundred dimensional vectors.

Then we make use of the NLTK module for the second phase of pre-processing. In the next step we remove the universal list of stopwords that are available in the module from the document. These are words that do not contribute any significant meaning to the summary and are generally articles, conjunctions or prepositions. After removing the stopwords, the resulting document is used as the input to our algorithm.

A similarity matrix is calculated between these formulated sentence vectors. The measure used for similarity is cosine similarity. If two sentences are considered similar, they would have a higher cosine similarity value. If a sentence is similar to many other sentence vectors, it will be considered as relevant to the whole document.

The best five or maximum, whichever is the lowest set of sentences would be selected to generate the extractive text

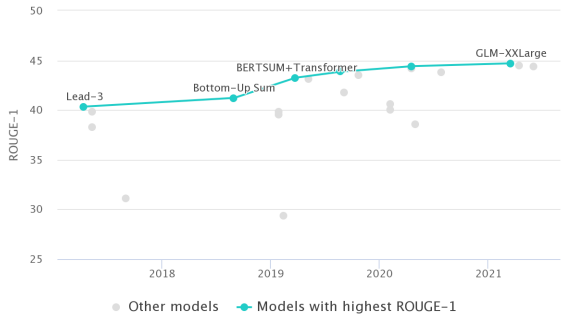

Fig. 1. State-of-the-art performance on the CNN-Dailymail News data set

summary. The accuracy of this summary is then compared with the human generated labels available in the dataset.

V. Experimental results

For evaluation, we used the test.csv file of the 'CNN-DailyMail News Text Summarization' data set from Kaggle. We have chosen the f1 score of ROUGE-1 metric as the evaluation method. Figure 1 shows the performances of various algorithms on the same data set. As per the chart, the leading algorithms have the ROUGE-1 score in the range of 40 to 45.

For Q-learning, average ROUGE score of 31.7 was achieved when it was executed on all the records. As mentioned in the implementation section, the Q-learning approach has a threshold in exploration phase which can decrease the score. In another experiment, we applied the same algorithm on only the texts in the test.csv file that have less than 15 sentences. For that, since the exploration phase will explore more candidates, the average ROUGE score was 37.8, which is close to the state-of-the-art range of 40 to 45.

For Text Rank, the average ROUGE score obtained was 27.7. This score was achieved after execution of the algorithm on the entire testing dataset consisting of 11490 articles. The approach shows that the score is slightly lower than Q-learning in terms of summary accuracy. But this algorithm is a more scale-able solution as we were able to accommodate documents consisting of almost 100 sentences.

VI. Conclusion and Future Work

From the experiments, it can be easily seen that the q-learning approach performs much better than the widely known text rank approach. However, there are some problems with the q-learning approach.

- The exploration phase is necessary for any new input text. It should be possible to learn from a large data set and exploit the learning from that to extract sentences from new input data.
- The run time is exponential, making it impractical for large texts.
- The ROUGE-1 score still needs to be improved.

For future work, the problems mentioned above can be addressed.

References

[1] https://www.kaggle.com/datasets/gowrishankarp/newspaper-text-summarization-cnn-dailymail
[2] Nikhil S. Shirwandkar and Samidha Kulkarni, "Extractive Text Summarization Using Deep Learning" 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)
[3] Heena A. Chopade and Meera Narvekar, "Hybrid auto text summarization using deep neural network and fuzzy logic system" 2017 International Conference on Inventive Computing and Informatics (ICICI)
[4] K Usha Manjari, Syed Rousha, Dasi Sumanth and J Sirisha Devi, "Extractive Text Summarization from Web pages using Selenium and TF-IDF algorithm" 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)
[5] Aditya Jain, Divij Bhatia and Manish K Thakur, "Extractive Text Summarization Using Word Vector Embedding" 2017 International Conference on Machine Learning and Data Science (MLDS)
[6] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pages 404–411, Barcelona, Spain. Association for Computational Linguistics
[7] Ryang, Seonggi, and Takeshi Abekawa. "Framework of automatic text summarization using reinforcement learning." Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012.
[8] Das, Dipanjan, and Andr FT Martins. "A survey on automatic text summarization." Literature Survey for the Language and Statistics II course at CMU 4.192-195 (2007): 57.