Automatic Summarization of Scientific Texts

Maria Dobko
Ukrainian Catholic University
Faculty of Applied Sciences
Lviv, Ukraine
dobko_m@ucu.edu.ua

Oleksandr Zaytsev
Ukrainian Catholic University
Faculty of Applied Sciences
Lviv, Ukraine
oleks@ucu.edu.ua

Yuriy Pryima
Ukrainian Catholic University
Faculty of Applied Sciences
Lviv, Ukraine
y.pryima@ucu.edu.ua

Abstract

In this work we overview recent advancements in the field of abstractive text summarization and discuss the use of deep learning models, especially Generative Adversarial Networks (GAN). We apply different models to our dataset of scientifix papers, acquired from arXiv, and evaluate them in terms of simplicity and performance.

We propose our own metric based on word-embeddings, which we believe will be more suitable for evaluation of abstractive summaries.

Keywords text summarization, NLP

1 Introduction

Abstractive text summarization is ... as opposed to extractive summarization where

1.1 Abstractive vs Extractive summarization

Abstractive text summarization allows us to write summaries that are almost as good as those written by humans.

Abstractive text summarization is more challenging, but it is close to what humans do.

Data-driven approach

Automatic text summarization is the task of producing a concise and fluent summary while preserving key information content and overall meaning[1].

1.2 The power of extractive summarization

Though it will not be the focus of our work, extractive text summarization is dominant in this field and serves as a baseline for abstractive models. Before we move any further, we must understand what makes statistical techniques so powerful.

1.3 Structure of the paper

In this study we focus on abstractive text summarization. We do a brief overview of the related work that has been done in this field. Then we proceed to discussing different models for statistical and deep summarization. Then we present our dataset of scientific papers collected from arXiv and compare the performance of different models on this dataset.

2 Related work

Allahyari et al.[1] make a survey of the most successful text summarization techniques as of July 2017.

This September Li et al. [7] described their submission to the sentiment analysis sub-task of ?Build It, Break It: The Language Edition (BIBI)? where they successfully apply generative approach to the problem of sentiment analysis.

In their paper *Generative Adversarial Network for Abstractive Text Summarization* Liu et al.[9] built an adversarial model that achieved competitive ROUGE scores with the state-of-the-art methods on CNN/Daily Mail dataset. They compare the performance of their approach with three methods, including the abstractive model, the pointer-generator coverage networks, and the abstractive deep reinforced model.

In contrast Zhang et al.[4] don't use reinforcement learning but rather introduce TextGAN with an LSTM-based generator and kernelized discrepancy metric.

3 Data collection and preparation

Thanks to the open policy of arXiv we were able to collect our own dataset of scientific papers from *stat.ML* category[1]. We publish it together with this paper an encourage everyone to use it as they like in their own projects.

Structure of the data Each paper on arXiv has a unique identifier (for eample: 1801.01587). It can be used to extract any data that is available and related to that paper, including its metadata and full text as PDF.

Our dataset contains 2000 documents (papers). Each one of them is represented by a row with the following properties:

1. arXiv's unique identifier
2. title
3. abstract
4. body of the paper

Our script[2] can be used to collect more data, but for our problem a set of 2000 papers from one category is enough. For example, DUC (Document Understanding Conference) dataset which is among the most commonly used in the feld of text summarization has 30 sets with approximately 10 documents each[3].

Cleaning the text We collected our data as PDF files and parsed them to extract the raw text. This produced a huge amount of uniterpretable UTF-8 characters from mathematical expressions. After removing those extra characters.

[1] https://arxiv.org/list/stat.ML/recent
[2] Code and instructions for collecting the data is available in our repository: https://github.com/MachineLearningUCU/TextSummarization
[3] https://www-nlpir.nist.gov/projects/duc/guidelines/2001.html

We wanted to remove everething that is not a known English word, but that would filter out words like "GAN", "backprop" etc. as most standard corpuses of words, such as nltk.corpus, don't include specialized scientific terminology. So we filtered the words using manually created rules based on features like word length and frequence of vowels. This leaves some noise behind, but it will not affect our models.

4 Models for text summarization

Three main classes of models that are used for text summarization task are statistical frequency computation models (TFIDF etc.), graph methods (TextRank, LexRank etc.) and machine learning approach.

4.1 Statistical methods

These methods are based on the assumption that the importance of a word or sentence in a text depends on the total number of times it appears in the document. This means that this classical approach ignores context and lexical features of the text. Furthermore, they are able to perform only extractive summarization.

4.2 Graph models

We can build a graph of each document where words or sentences are nodes and the edges are the connections between each pair of nodes. The weights on these edges represent similarity between words or sentences in the whole text. While proving to have better results than simple frequency-based methods, graph models are still bounded by the absence of lexic understanding and ability to perform extractive summary only.

4.3 Machine learning approach

In the last years, lots of attention was focused on learning how to apply neural networks to NLP tasks, including text summarization. Using encoder-decoder models it is now possible to produce abstract summaries. In [10] the off-the-shelf attentional encoder-decoder RNN that was originally developed for machine translation was applied to summarization and outperformed state of-the-art systems on two different English corpora. However, there is not much of information about neural networks usage in scientific text summarization.

4.3.1 Generative adversarial networks

Generative adversarial networks (GAN) have shown a lot of success in image generation. However until recent years they were considered inapplicable to the discrete problems of natural language processing (NLP). The latest papers introduce novel approaches to overcoming these issues by combining GANs with reinforcement learning models and lay the foundation for the whole new field of research of adversarial language processing.

Recent studies have shown that neural networks can be used for solving NLP problems. However, models that were mostly considered for this task were convolutional neural networks and recurrent neural networks.

Applying generative adversarial networks to the problems of NLP is considered to be a complicated task because GANs are only defined for real-valued data, and all NLP is based on discrete values like words, characters, or bytes.

> For example, if you output an image with a pixel value of 1.0, you can change that pixel value to 1.0001 on the next step. If you output the word "penguin", you can't change that to "penguin + .001" on the next step, because there is no such word as "penguin + .001". You have to go all the way from "penguin" to "ostrich".[4]

However, in their latest paper Fedus, Goodfellow, and Dai[3] overcome this problem by using reinforcement learning to train the generator while the discriminator is still trained via maximum likelihood and stochastic gradient descent, and use it to fill the gaps in the text.

Li et al.[6] take a different approach...

However, this approach has not been used for abstract text summarization of scientific papers, yet.

5 Evaluation metrics

Evaluating a summary is a difficult task because there is no such thing as a single summary that would be ideal for a given document. In most cases even human evaluators can not agree on which of the given summaries is better[2]. Unlike other NLP problems, such as translation or parsing, when it comes to text summarization we can not clearly define what makes a summary good or bad. Therefore we must make assumptions about the space of good summaries.

1. assume that a good summary would be close to some *ideal* summary manually created by humans
2. assume that the goodness of summary can be measured as the ammount of important information it contains (this assumption can be infered from the definition of text summarization).

In the following sections we briefly describe some of the commonly used metrics for text summarization that is based on the first assumption and propose our own metric that is based on second one (in fact, we will show that the proposed metric can be formulated in a different way to work with the first assumption).

5.1 ROUGE

The most widely used score for evaluating text summarizations is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) inroduced by Chin-Yew Lin in 2004[8][11].

[4] Ian Goodfellow's answer to the related question on Reddit: https://www.reddit.com/r/MachineLearning/comments/40ldq6/generative_adversarial_networks_for_text/

$$\text{ROUGE-N}(s) = \frac{\sum_{r \in R} \langle \Phi_n(r), \Phi_n(s) \rangle}{\sum_{r \in R} \langle \Phi_n(r), \Phi_n(r) \rangle}$$

$$\text{ROUGE-L}(s) = \frac{(1 + \beta^2) R_{LCS} P_{LCS}}{R_{LCS} + \beta^2 P_{LCS}}$$

ROUGE works well for extractive text summarization. But if we need to evaluate the generated summary which contains different words from the ones that occured in paper, the score will always be small because, even though the new words can be close to the expected ones, two summaries don't overlap in terms of word equality.

For example, if the human-created summary is *"The great paper"* and our model produces *"A wonderful article"*, ROUGE score will be 0, even though the summary is perfect.

5.2 Embedding-based metric

We propose a metric that uses word embeddings to evaluate summaries based on their semmantic distance to the space of good summaries. Our assumption is that a good summary of a document contains words that are semantically close to the most important words or n-grams in that document.

Let s be the generated summary. If $|s|$ is the number of words in summary s, then $m = |s| - n + 1$ is the number of n-grams in this summary. Let $s_1, s_2, \ldots, s_m$ be all n-grams of summary s and let $c_1, c_2, \ldots, c_k$ be the k most important n-grams in the document. As we will show in section 5.2.1, there are many ways of measuring the importance of n-grams in a document.

The metric we propose is in fact a continuous version of ROUGE-N. Instead of testing the equality of n-grams in the compared summaries we use the continuous measure of semmantic distance between those n-grams.

For each n-gram in the generated summary we calculate the embedding-based score as its distance to the closest important n-gram in the document.

$$\text{CROUGE-N}(s_i) = \min_j ||s_i - c_j||_2$$

Now we define the score of the whole summary as the average score of its words

$$\text{CROUGE-N}(S) = \frac{1}{n} \sum_{i=1}^{n} \min_j ||s_i - c_j||_2$$

5.2.1 Measuring word importance

To choose which words are important we can use an algorithm of extractive text summarization. We will use **term frequency-inverse document frequency** (tf-idf) as a measure of importance because of its simplicity and importance (according to [5], tf-idf is one of the universally used terminologies in extractive summarization).

6 Experiments and results

We have tried different methods of text summarization, both extractive (TextRank, TF-IDF) and abstractive (Seq2Seq, SeqGAN).

6.1 TextRank

6.2 Seq2Seq

We have used seq2seq model with attention for a task of text summarization. Seq2seq have proven to provide state-of-art result in tasks of sequence generation. As input model takes paper abstract converted to the vectorized representation using word embeddings. The input sequence is limited by 600 words. All abstracts that are bigger than limit are omitted. All smaller abstracts are padded with ⟨SOS⟩ word that represents the end of a sequence. Model outputs sequence derived from the probability distribution. Each output word samples from this distribution having input sequence and previously generated samples. Output sequence is limited by 30 words. First ⟨SOS⟩ word represent the end of a generated summary.

7 Evaluating results

Model	ROUGE-1	ROUGE-2	ROUGE-L
TextRank	0	0	0
LSTM	0	0	0
SeqGAN	0	0	0

8 Baseline models

As a baselin model we selected a seq2seq model with deep LSTM[12] together with beam search and attention. At this point it is too hard to produce an abstract from the text of a paper, so we started with a simpler task of generating a title from the text of an abstract. We represented each absract as a numeric vector using word embedings and fed it to the model together with a corresponding title. After some training our model was able to generate meaningful titles for most abstracts. Take a look at this example:

Abstract this is great popcorn and i too have the whirly pop. the unk packs work wonderfully. i have not found it too salty or the packages leak. i have found the recent price of $35 too expensive and have purchased direct from great american for half the price.

Predicted summary great popcorn!!

Actual summary great unk american popcorn

It is not clear yet if we will be able to produce abstracts based on the whole text of an article. Such problems have very big feature space which might overcomplicate our task. So on the next stage of our project we will continue generating titles from abstracts and try doing that with discrete GANs. If our experiments prove to be successful, we will try scaling up to full texts of papers in our dataset.

9 Strength and weakness of the study

GANs have proved to be the most successful when it comes to generative images, but their application to the problems of the text generation is not well studied. So we expect our research to introduce novel approaches and original ideas that may advance the field of natural language processing. However, there are high risks that this approach may not give good results at all, because there are still lots of issues about usage of neural networks with language data. The other possible weakness is a difficulty to compare the results with other papers, as there are not a lot of researches concerning this or relative subject. We are also currently looking for supervisors who might be interested in the following topic, so we could have a mentorship during the research.

Conclusions

In this paper we demonstrated how Generative Adversarial Networks can be used for abstractive text summarization.

References

[1] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. Text summarization techniques: A brief survey, 2017.

[2] Dipanjan Das and André F. T. Martins. A survey on automatic text summarization. 2007.

[3] William Fedus, Ian Goodfellow, and Andrew M. Dai. Maskgan: Better text generation via filling in the gaps, 2018.

[4] Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation, 2017.

[5] Yogan Jaya Kumar, Ong Sing Goh, Halizah Basiron, Ngo Hea Choon, and Puspalata C Suppiah. A review on automatic text summarization approaches, 2016.

[6] Yang Li, Quan Pan, Suhang Wang, Tao Yang, and Erik Cambria. A generative model for category text generation, 2018.

[7] Yitong Li, Trevor Cohn, and Timothy Baldwin. Bibi system description: Building with cnns and breaking with deep reinforcement learning, 2017.

[8] Chin-Yew Lin. Rouge: a package for automatic evaluation of summaries. July 2004.

[9] Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. Generative adversarial network for abstractive text summarization, 2017.

[10] Ramesh Nallapati, Bing Xiang, and Bowen Zhou. Sequence-to-sequence rnns for text summarization. *CoRR*, abs/1602.06023, 2016.

[11] Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. Bleu: a method for automatic evaluation of machine translation. pages 311–318, 2002.

[12] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014.