Università degli studi di Milano-Bicocca

Text Mining and Search

Project

Abstractive Text Summarization using Attentive SEQ2SEQ Models

Authors:
Federico De Servi - 812166 - f.deservi1@campus.unimib.it
Christian Uccheddu - 800428 - c.uccheddu@campus.unimib.it

February 7, 2021

Abstract

The aim of this project is to generate summaries for news articles taken from the CNN dataset, through a technique called abstractive text summarization. This involves creating models that generate structurally correct and meaningful summaries from the text. Two different models are created and trained for 20 epochs, to see the results that such models could produce when trained on a high-end consumer machine. The performance assessed in terms of ROUGE is low, and this confirms the training complexity of Seq2Seq models. However, some differences in terms of fluency between the two models are found. Also, it is known that in the context of text summarization, quantitative measures must always be accompanied by a human evaluation. After analyzing the predictions of both models it can be said that the bidirectional model is the one that produces lexically more complex results. Performance was then compared with pretrained models such as BART and T5 in order to evaluate the differences.

Keywords: *Machine Learning, text summarization, CNN news*

Contents

1 Introduction **2**

2 Datasets **2**

2.1 Preprocessing . . . 2

3 The Methodological Approach **3**

3.1 Models . . . 4

3.2 Training . . . 5

3.3 Inference . . . 5

3.4 Evaluation . . . 5

3.5 Results . . . 6

4 Conclusions **7**

List of Figures

1 Abstractive . . . 2

2 Extractive . . . 2

3 Encoder . . . 3

4 Decoder . . . 4

5 Bidirectional LSTM . . . 4

6 Loss bi-directional model . . . 6

7 Loss mono-directional model . . . 6

1 Introduction

Text summarization is a difficult challenge in natural language understanding, and it has seen amazing improvements in recent years thanks to the introduction of a specific kind of models that uses an encoder-decoder architecture (explained later). Also, after the introduction of the famous attention mechanism, and its following implementation in these models, further improvements has been done. There are two different approaches to text summarization:

- **Extractive**
- **Abstractive**

The first one tries to find meaningful parts of the original text and concatenates them together to generate a summary. The second one, the one used in these projects, generates the sentences from scratch in order to capture the most important facts contained in the original text.

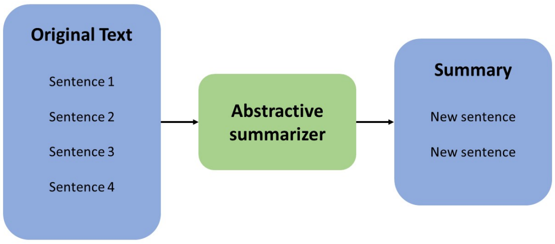

Figure 1: Abstractive

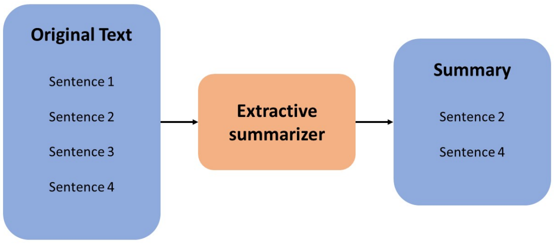

Figure 2: Extractive

The models will be evaluated using specific measures that established themselves as standard in the Natural Language Processing world, such as ROUGE, that measure the percentage of n-grams presents both in the original summary and in the generated one, and also evaluating the fluency of the generated summary through human interpretation of the results.

2 Datasets

The dataset is originally taken from the CNN dataset available here. This dataset contains news stories and accompanying summaries from the news articles of CNN. The first thing to decide is the numerosity of the dataset on which to train the models. Since, as we will see later, they contain millions and millions of parameters to be trained, only 10,000 stories-summaries has been retained from the original dataset for training. Bear in mind that each story contained more than one summary, so the dataset has been processed later in order to obtain a *one-to-one* relationship. This resulted in more than 10,000 records for the training dataset. Same applies for the test dataset, that contained originally 1,000 stories-summaries that encountered the same preprocessing steps as before.

2.1 Preprocessing

Before feeding the data to the models, the following preprocessing steps had to be applied.

- **Normalization:** is the process of transforming a text into a standard form. In particular it has been done following these steps:
  - Non-ASCII characters removal
  - Lowercase conversion
  - Punctuation removal
  - Numbers removal

  - Stop-words removal
  - Contractions replacement

- **Lemmatization:** the process of taking into consideration the morphological analysis of the word.

- **Tokenization:** the process of breaking text documents apart into smaller pieces.

After applying the preprocessing step, the differences within the text are easily evident. Here is the example of transformation of a sentence after the preprocessing phase:

Before preprocessing	**After preprocessing**
`(CNN) -- Renowned radio personality`	`renowned radio personality casey`
`Casey Kasem is in critical condition`	`kasem critical condition`
`at a hospital in western Washington,`	`hospital western washington`
`a spokesman for St. Anthony Hospital`	`spokesman st anthony hospital`
`told CNN in a written statement`	`told cnn written statement`
`Thursday.\n`	`thursday`

3 The Methodological Approach

Models for abstractive summarization fall under a wider group of deep learning models called Sequence-to-Sequence models (Seq2Seq)[1], which map an input sequence to a target sequence. In particular, the models used in this project belong to the group called “Seq2Seq Attentive models”[2]. They are composed of three main elements:

- Encoder
- Decoder
- Attention layer

The baseline idea is the following. The encoder reads the input sequence one timestep at the time, to capture the contextual information present in the input and outputting a context vector. Generally, the encoder-decoder architecture makes use of gated RNNs, such as LSTMs.

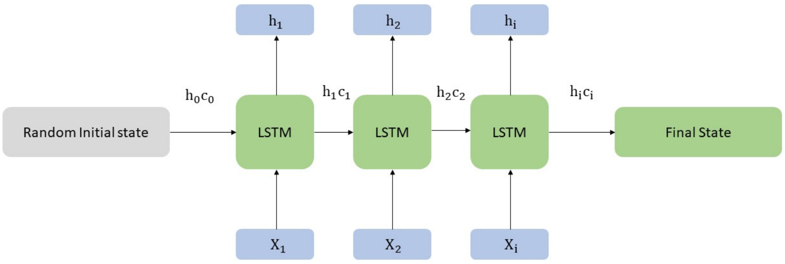

Figure 3: Encoder

The hidden states and the cell state of the final timestep are then saved and used to initialize the decoder. The decoder then extracts the output sequence from the resulting context vector outputted from the encoder. In other words, it predicts the next word given the previous one.

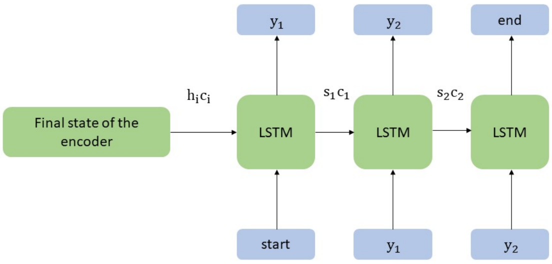

Figure 4: Decoder

Iterating this procedure produces the final output, the generated summary. The main issue with this model is that a neural network needs to be able to compress all the necessary information of a sentence into a fixed-length vector. This leads to the fact that long sentences make the model struggle, making the performance deteriorate quickly as the length of an input sentence increases. The attention mechanism tries to solve this issue by making the model predict the output word by paying attention at few specific parts of the sequence, rather than the entire one. After the construction from scratch of two models, the performances of two pre-trained models will also be shown: BART and T5.

3.1 Models

Two main models have been trained in this project, all while trying to keep them as simple as possible to be trainable from common computers.

First model The first model implemented a unidirectional LSTM encoder-decoder, with randomly initialized word embeddings and a global attention layer between the encoder and the decoder. This means that all hidden states of the encoder are considered for deriving the attended context vector.

Second model The second model improved from the first model by implementing a bidirectional LSTM encoder, while maintaining a unidirectional decoder. In particular, bidirectional LSTMs (BiLSTMs) run the input in two ways: one from past to future and one from future to past. Then, by using the two hidden states combined the BiLSTM is able in any point in time to preserve information from both past and future. In practice this means that they can understand context better than their unidirectional counterpart.

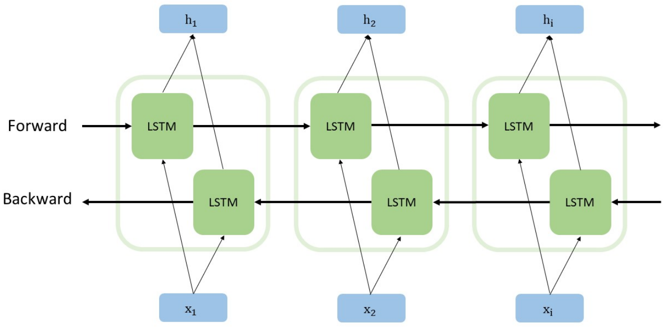

Figure 5: Bidirectional LSTM

The forward and reverse hidden states of the bidirectional encoder are then concatenated two by two and then fed as initial states to the unidirectional decoder. This should give the model better understanding of the patterns in the text.

We then used the following two pre-trained models, in order to evaluate the differences in terms of performance with respect to our models.

BART This model uses a standard seq2seq/machine translation architecture with a bidirectional encoder (like BERT) and a left-to-right decoder (like GPT)[3]. This model partially reflects the structure of the model built by us. In particular, BART is particularly effective when fine tuned for text generation but also works well for comprehension tasks.

T5 T5 is a Google pre-trained model, which propose reframing all NLP tasks into a unified text-to-text-format where the input and output are always text strings, in contrast to BERT-style models that can only output either a class label or a span of the input.[4]

3.2 Training

During training, the model trains the encoder and the decoder simultaneously. Apart from the epochs, the selected optimal hyperparameters for training has been chosen after various trials, and resulted in:

- Optimizer: *RMSPROP*[5]: It's been decided to use RMSPROP because empirically it has been seen to work better than other proven optimizers such as ADAM
- Learning Rate: 0.001. The choice of this learning rate was made empirically after seeing that it was a good compromise between execution speed and having avoided overfitting.
- Batch size: 32. The choice of this batch size was made empirically because it produced better results.
- Epochs: 20. The number of epochs, meanwhile, has been chosen to be set to 20, since the long training time that each epoch took on a medium/high-end consumer PC made unpractical to do otherwise.

3.3 Inference

During the inference phase, each model was constrained to produce summaries that took as input the first 300 tokens of the original story, and output a text with a maximum lenght of 12 token. This happened for both of our models and for the pre-trained models as well.

3.4 Evaluation

The results obtained from the models has been evaluated using two different ways. The first one has been to use the ROUGE scores[6].

$$\text{Rouge - N} = \frac{\sum_r \sum_s \text{match}(\text{gram}_{s,c})}{\sum_r \sum_s \text{count}(\text{gram}_s)} \tag{1}$$

This measure tries to assess how well a system-generated summary covers the content present in one or more human-generated model summaries known as references, by simply counting how many n-grams in the generated summary matches the n-grams in the reference summary. Many different versions of this measure exist, based on the length of the n-grams to be used. *Since the summaries in the dataset are particularly brief, we decided to use ROUGE-1 and ROUGE-2 scores.* The issue with ROUGE scores is that they merely assess the adequacy of the words covered in the generated summary, but they cannot determine if the result is coherent or the sentences flow together in a sensible manner. This has to be determined by a human operator[7], and to do that evaluate the results and their relative fluency and correctness[8].

3.5 Results

The performance obtained by the two models is predictably *not high enough to produce adequate results, compared to the state of the art summarization models.* It is obvious that Seq2Seq models require lots of computational power and facilities in order to produce high-quality results, given the fact that they are composed of dozens of millions of parameters that have to be trained. In this case, the models have been trained for 20 epochs with just a small subset of documents on a high-end consumer computer, and this took approximately more than 30 hours of training time.

However, the models produced a few summaries that impressed for their ability to capture the important topics in the original text, and a few of them sounded also quite fluent.

For what concernes the difference between the two models, the monodirectional one achivied a slightly better Rouge-2 score inspite of the Rouge-1, compared to the bidirectional one. (Table 1)

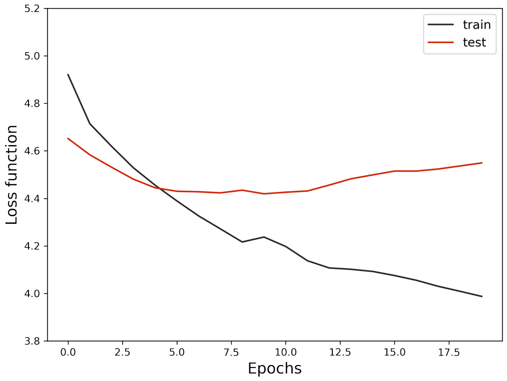

Figure 6: Loss bi-directional model

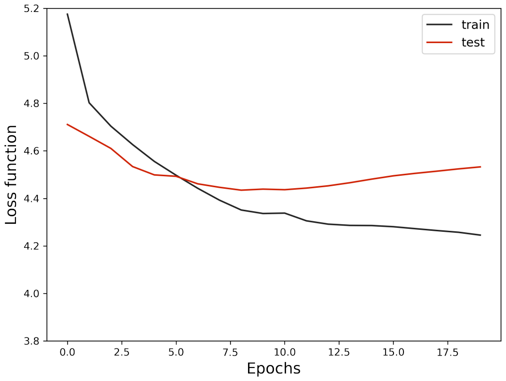

Figure 7: Loss mono-directional model

Model	**Rouge 1**	**Rouge 2**
Mono-directional	7.56 %	0.80 %
Bi-directional	6.62 %	0.66 %

Table 1: Rouge results

As is known from the theory, in the field of text mining, the values of the evaluation metrics must be combined with a human evaluation to evaluate the semantic goodness of the summaries. For this reason it's been decided to use the bidirectional model to build the summaries. This points that, while it failed more often than its counterpart to identify the important topics in the original text, **it also produced outputs that sounded more fluent.** In fact, a higher Rouge-2 score means that the number of 2-grams matches are higher, and this means that the outputs resemble the original sintactic structure of the sentence more. This result has also been confirmed by ourselves, looking at the generated summaries of the model. Below is the example of the prediction of a summary respect to the original summary. Remember that both summaries are lemmatized, and as such contains only dictionary-form words.

Original summary	**Bi-directional summary**
`new missiles travel kilometers`	`north korea nuclear launch`

The two pretrained models are then also evaluated, in order to investigate the performance gap between them and our models. As it can be seen, the two obtain higher performances in ROUGE-1 Scores, but it is in ROUGE-2 performance that the highest gap can be seen. This means that the models can generate higher quality summaries for most of the stories, while ours generated high-quality summaries only for few of them.

Model	Rouge 1	Rouge 2
BART	17.46 %	7.23 %
T5	20.51 %	8.81 %

Table 2: Rouge results of pre-trained models

However, it is worth noting that, whenever our models created good enough summaries, they resembled what the two pre-trained models produced, as it can be seen below. This might hint to the possibility that training our two models for a longer time, with more computational power and more data to feed into the training, they could present a noticeable improvement in performance.

BART summary

```
pyongyang north korea really
look like much
```

T5 summary

```
north korea says rocket
launch essentially
```

4 Conclusions

It is clear how these Seq2Seq models need *more training time* and far more computational power than what is available to a normal developer/consumer. However, what stood out the most is that they were capable to produce a few summaries that impressed, given the limited resources that we had to deal with. In particular, the model that made use of the bidirectional LSTM generated more complex outputs, and confirms its greater capacity in recognizing patterns in text. As easily understood, the models that perform best both in terms of ROUGE and in terms of readability of the summaries are the two pre-trained networks. The fact that, whenever the bidirectional model produced good results, they resembled the summaries generated by the pretrained networks might hint to the possibility that with more computational resources, data to train with and longer training time, it could present a big improvement in performances.

References

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

[2] Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K. Reddy. Neural abstractive text summarization with sequence-to-sequence models. 2020.

[3] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

[4] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

[5] Sebastian Ruder. An overview of gradient descent optimization algorithms. 2017.

[6] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Proceedings of the ACL Workshop: Text Summarization Braches Out 2004*, page 10, 01 2004.

[7] Feifan Liu and Yang Liu. Correlation between rouge and human evaluation of extractive meeting summaries. 201-204:201–204, 01 2008.

[8] Jun-Ping Ng and Viktoria Abrecht. Better summarization evaluation with word embeddings for rouge. 2015.