**ABSTRACTIVE TEXT SUMMARIZATION USING ATTENTIVE SEQ2SEQ MODELS**

**Abstract**

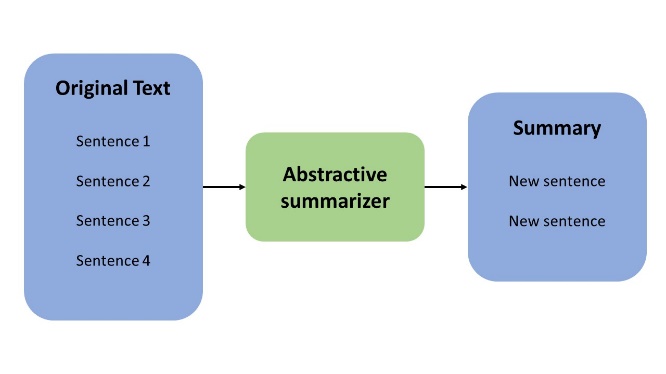
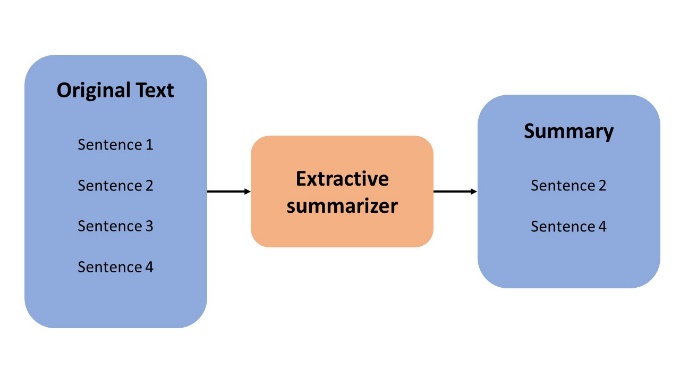
The aim of this project is to generate summaries for news articles taken from the CNN dataset, through a technique called abstractive text summarization. This involves creating models that create structurally correct and meaningful sentences from the text. Two different models are created and trained for 20 epochs, to see the results that such models could produce on a medium-high end consumer machine. We obtained that….

**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 and 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.

****

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.

**Dataset**

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 1000 stories-summaries that encountered the same preprocessing as before.

**Preprocessing**

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

* Normalization
  + Non-ASCII characters removal
  + Lowercase conversion
  + Punctuation removal
  + Numbers removal
  + Stop-words removal
  + Contractions replacement
  + Lemmatization
  + Tokenization

After applying all these steps, the resulting dataset looks like this:

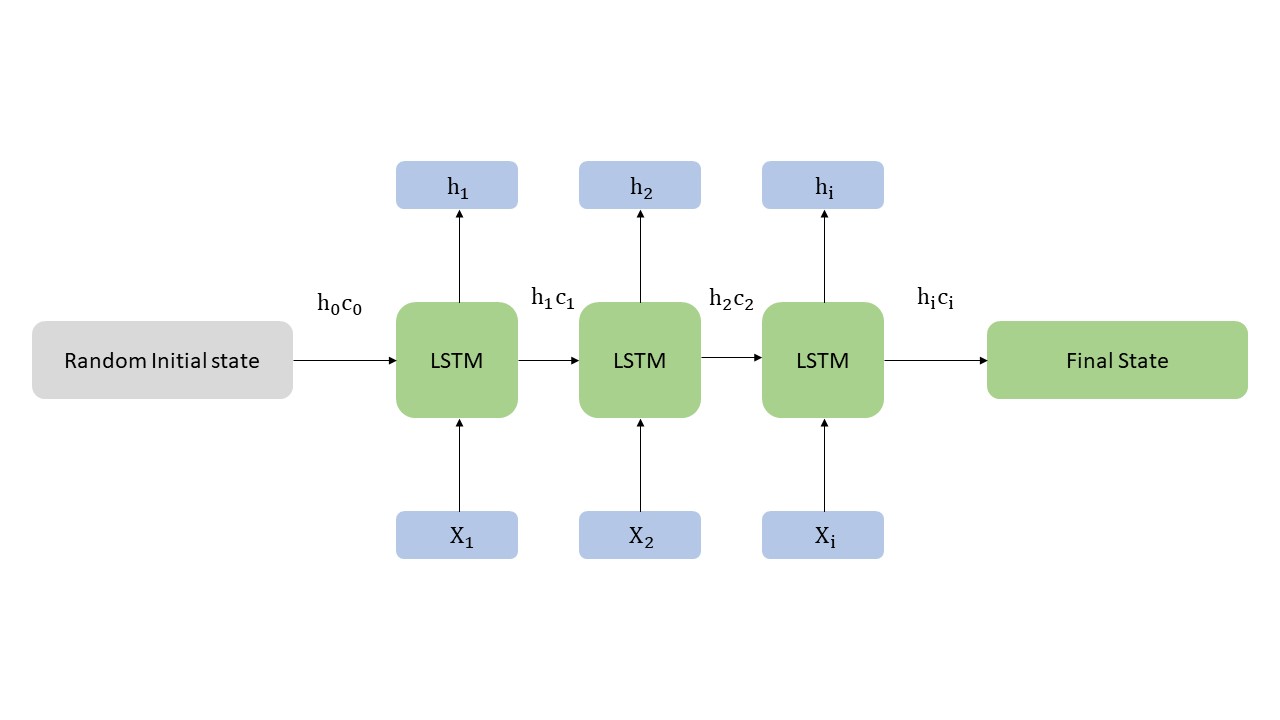
ESEMPIO SCREENSHOT

**Approach**

Models for abstractive summarization fall under a wider group of deep learning models called Sequence-to-Sequence models (Seq2Seq), 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”. They are composed of three main elements:

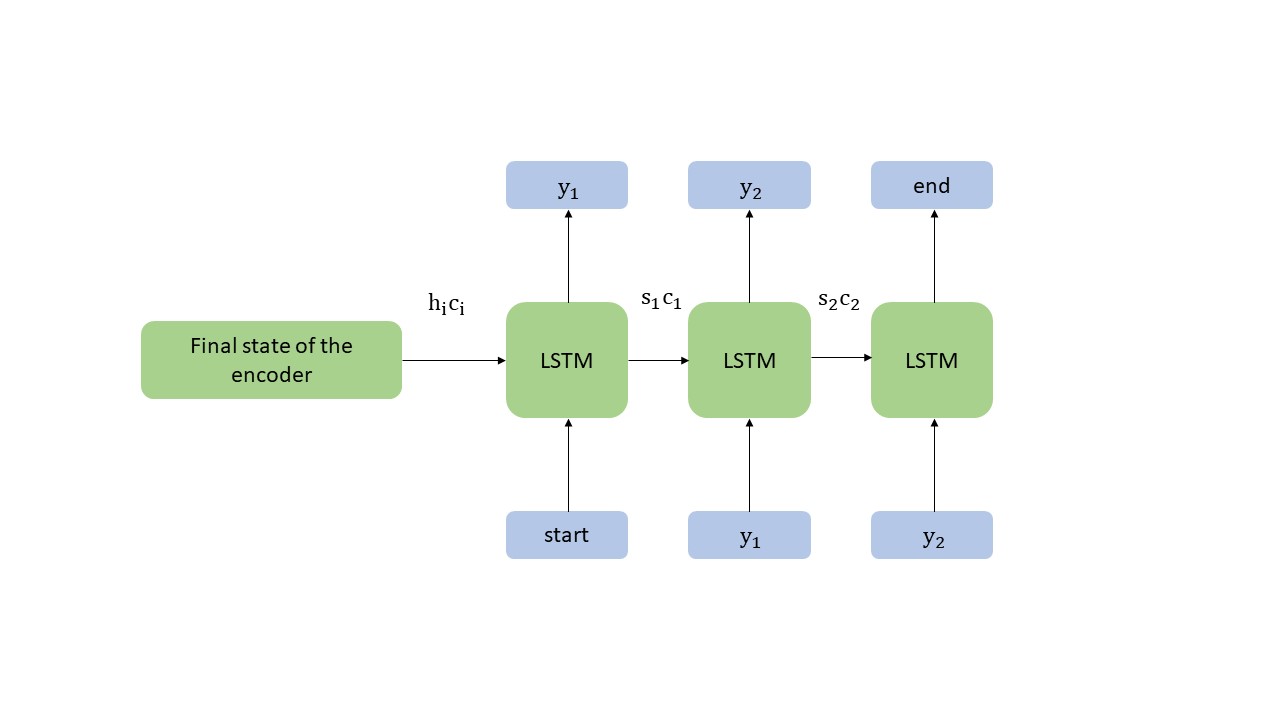
* Encoder
* Decoder
* Attention layer

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.



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.



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.

**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.

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.

IMMAGINE

The second model improved from the first model by implementing a bidirectional LSTM encoder, while maintaining a unidirectional decoder. 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.

IMMAGINE

**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
* Learning Rate:
* Batch size:
* 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.

**Evaluation**

The results obtained from the models has been evaluated using two different ways.

The first one has been to use the ROUGE scores. 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, and to do that evaluate the results and their relative fluency and correctness.

**Results**

Results…

**Conclusions**

Conclusion…

References