NUANS: Miniproject
Episodes' Title Generation

Federica COCCI
ID:1802435

June 9th, 2023

Abstract

This miniproject aims at the generation of titles giving in input the plot of a TV series episode. I use the pre-trained model distilGPT2 for title generation and then compare the obtained results with the use of the pre-trained model T5 focusing on the metrics to make some comparisons. As metrics for the evaluation I have implemented the Catchiness Score and also the Cosine Similarity in order to measure the semantic similarity. The dataset is custom-made, using a scraper that I have implemented which takes the data directly from Wikipedia.

1 Introduction

The idea behind the project was born reading *"Building a Slogan Generator with GPT-2"* (Bgn, 2020) where the author exposes his project about the generation of new slogans giving as input `context` (company's name and description) along with the `slogan` and performing the fine-tune of the GPT-2 model. My project, following the wake of the slogan generator, wants to generate a new title for the plot in input to the fine-tuned model and since in the above cited project the GPT-2 model seems to work in this field then I decided to use distilGPT2. To study the ability to generate catchy titles, I decide to also fine-tune the model T5 making some comparisons between their results and applying two types of metrics in order to study the semantic meaning besides the catchiness.

2 Dataset

The data for dataset building has been collected using a custom-made scraper which gets entries from Wikipedia. In particular each entry is a pair plot-title from 30 American TV series for a total of 4560 entries. Using the MediaWiki API and in particular GET requests I am able to retrieve a list of page's sections in JSON format ready to be parsed; in order to get the series' Wikipedia page I use directly the name of the TV series making it compatible with the GET request. After the JSON retrieval, I look for the section with the seasons list and from it through other GET requests I obtain the title-plot entries. In particular I need to apply several regex in order to clean the resulting titles and plots. At the end of the scraping I get a csv file made of two columns: PLOT and TITLE.

3 Fine-tuning of distilGPT2

DistilGPT2 is a model developed by Hugging Face and is the smallest version of GPT-2. Like GPT-2, distilGPT2 can be used to generate text. I work with this language model using classes that exist in Huggingface Transformers/GPT-2.
First of all I need to process the data to fine-tune the model and since distilGPT2 will generate a title given a plot, I concatenate the entries of the csv file in order to get:

<bos> *plot* <sep> *title* <eos>

and thus I introduce a new special token for the tokenizer which is title token <|*title*|>. Each resulting sequence is then passed to the tokenizer and it returns the corresponding input_ids and attention_mask. Then I train the model using the HuggingFace Transformers API in order to get the data collator and the trainer's objects. The training phase takes 5 epochs: due to the limit in Google Colab regarding the GPU, I tried without success to perform both the validation during the training and the evaluation after the training.

4 Fine-tuning of T5

As reported in Huggingface Transformers/T5 "*[...]we always need an input sequence and a corresponding target sequence. The input sequence is fed to the model using input_ids. [...]the target sequence corresponds to the labels*". For this reason in this case it is not needed to concatenate the plot with the corresponding title: I tokenize separately both plot and title, setting in both cases the parameter max_length to a number computed before which takes into account all the samples in the dataset. After the computation of titles' input_ids, I set to -100 all the values corresponding to the pad token and in this way I obtain the final label, which will be passed to T5 together with the input_ids obtained from the tokenization of the plot. During the training of this model, the validation and the evaluation are successfully performed.

5 Generation of new titles

For the generation of new titles, the input to fine-tuned models consists only of a plot which is pre-processed in the case of distilGPT2. I use the already implemented method `generate()` (from Hugging Face) to generate new titles with both the models.

5.1 DistilGPT2

In this case the plot is pre-processed with the addition of bos token and title token:

<|bos|> *plot* <|title|>

Since the `generate()` method returns the input concatenated to the generated title, I need to set the max_length parameter corresponding to the length of the tokenized input + max_new_tokens. In this case I have decided that titles must have a maximum length equal to 6. Taking into account what has just been said, I need to post-process the generated response of distilGPT2 in order to separate the input sequence from the new generated title.

6 Evaluation

About the evaluation I decided to implement two kinds of metrics:

1. Catchiness Score
2. Cosine Similarity
   (a) title-title (T-T)
   (b) plot-title (P-T)

Since the generated title won't be the same as the original one, maybe it's more convenient to use metrics which work in semantics rather than the metrics that take in account the words, but not the meanings behind them nor the context e.g. ROUGE. The generated title can be compared with the actual title from the point of view of context and catchiness, and with the plot from the point of view of context.

6.1 Catchiness Score

From the paper *TiZen: Neural Title Generation for Scientific Papers* (Jain et al.) I have taken the metric to measure the catchiness of generated titles. Here in the paper, the authors affirm *"the basic intuition behind the definition of catchiness is that less frequent or rare content words make a title catchy"* and thus the following formulae are from:

$$TC_G = -\frac{\sum_{i=1}^{m} \text{plot}_{\text{count}}[actual[i]]}{m}$$

$$TC_P = -\frac{\sum_{i=1}^{n} \text{plot}_{\text{count}}[predicted[i]]}{n}$$

$$CS = TC_G - TC_P$$

where $\text{plot}_{\text{count}}$ contains the counts of words in the given plot, actual[i]/predicted[i] represents the i-th word in the actual/generated title, m is the number of words in the actual title and n is the number of words in the generated title, TC_G is the Title Catchiness Score of the actual title, TC_P is the Title Catchiness Score of generated title, CS is the Catchiness Score.

6.2 Cosine Similarity

For the computation of this metric I use GloVe embeddings and the `cosine_similarity()` function of scikit-learn.

6.3 Comparisons

In the table 1 I report the results from metrics.

Metric	Model	Value
Catchiness Score	distilGPT2	0.492
Catchiness Score	T5	-1.645
Cosine Similarity T-T	distilGPT2	0.233
Cosine Similarity T-T	T5	0.230
Cosine Similarity P-T	distilGPT2	0.272
Cosine Similarity P-T	T5	0.239

Table 1: Metrics results.

From the Catchiness Score it is evident how distilGPT2 generates titles which have a greater level of catchiness compared to T5. This great difference is not reflected in the computation of Cosine Similarity where values are all around 0.2. This means that although the distilGPT's titles are catchy, they are not in the same context of plots.

7 Final considerations

Looking at Table 2, one can notice that the semantic similarity between plots and their original title is low, and therefore low results in the Cosine Similarity calculations are justifiable. In the Internet there are projects with the same goal as mine but they have a dataset where the title is related to the newspaper article/scientific paper as, in general, this kind of titles are semantically related to their text. In my case the problem is that, in reality, the title of an episode (but this issue can be extended to movies) is not usually made to be correlated to the plot because it may be catchy and, to obtain this, the title can be composed of words uncommon (and so, not really correlated) to the plot. With that being said, my project shows that a catchy title that is *also* semantically correlated to the given plot cannot *generally* be obtained due to the constraints of the dataset itself (as initially said) even if, as shown in Table 1, the catchiness score for distilGPT2 is relatively high.

Metric	**Value**
Cosine Similarity P-T	0.216

Table 2: Cosine Similarity between plots and original titles.

References

Jonathan Bgn. 2020. Building a slogan generator with GPT-2.

Huggingface Transformers/GPT-2. Openai gpt2.

Huggingface Transformers/T5. T5.

Harshil Jain, Anubhav Jain, Chandan Maji, Abhisht Tiwari, and Naman Jain. Tizen: Neural title generation for scientific papers.

MediaWiki API. Official site.