Book Title Generation for an Author

Tasio Aguirre Blanco

Abstract. *Artists can have a particular way of naming their creations (movies, books, musical albums or song titles...) and it is common to sometimes feel that some of those titles remind you of a particular author, maybe because they follow a similar style, or because the topics are already familiar. In this paper we present a system capable of generating non-existing book titles for a given author following a similar style and topics, that feel like books that author could have written.*

1 Introduction

As we mention in the abstract, artists usually name their works in a particular fashion to make their creations more recognizable for the general public. When a music band we like releases a new album, or when our favorite movie director releases a new film, it is common to feel familiar with the title because it uses a particular naming style or because it reminds us of their previous works, even though we have never seen it before.

An interesting idea is to use Artificial Intelligence to generate new titles given a list of previously published works an artist; maybe to help an artist that has run out of ideas by suggesting new topics, or just as a fun experiment to see if we can identify what it is that makes some artists recognizable. In this work we present a system that does exactly that, generate realistic sounding book titles using already published books of an author, covering similar style and topics. We want to generate completely original book titles that do not exist.

In this paper we will explain the tools we used to achieve this, and we will show examples of the results we obtained along the way. The aim of this paper is not to provide a profound explanation and analysis of the methods and results we got, but to provide a view of how to build a system like this, and to share the interesting things we found while doing so.

2 Methods

In this section we will show how we achieved each one of the requirements we want our system to have. Shortly, they are these three:

- Generate realistic sounding book titles.

- Get titles following the same writing style an author usually uses.

- Get titles that cover similar topics to the ones used by that author.

We will take advantage of the GoodReads dataset which includes information of a high amount of books, including author, publisher, ratings... The dataset includes more than 180k books written in English that we will use in this project.

2.1 Generating Text: GPT-2

Before trying to generate any kind of book title, we need an actual way to generate any sort of text, and for this task we will use the famous Generative Pre-Trained Transformer (GPT-2) model [1] from OpenAI released in 2019. Even though a newer version (GPT-3) has been developed, its source code has never been made available and can only be used through an API by Microsoft; so we will be using the older, but still powerful version.

GPT has been quite popular in the media these last years given its ability to generate human-like speech, and its impressive performance in different tasks such as text translation, question answering, summarizing passages... GPT-2 is an "unsupervised transformer model trained to generate text by predicting the next word in a sentence of tokens" [2] that uses the context extracted from the previous words to generate new ones.

For this task we can use already existing book titles from a given author as the context that GPT-2 will use to generate new text that follows, more-or-less, the same style and topic, by simply creating a list of elements (concatenating book titles using the newline character), feeding it to the model, and asking it to generate text using that list as context. The list could look something like this:

```
Book title 01 \n
Book title 02 \n
...
Book title 10 \n
```

We found 10 titles to be a good balance where the model receives enough information and variety to generate realistic text, but not too much so the model confuses different topics and produces a more noisy output. For each title we want to generate we will choose 10 published book titles randomly, except for the cases where the bibliography of an author is smaller than that. In those cases we will take all book titles in a different order each time to generate new ones. An obvious consequence is that the system will generate more diverse titles for authors with huge bibliographies compared to those authors that have published fewer books.

We will compare the output the model produces with the set of all book titles in the GoodReads dataset to remove already existing titles from the output. *Table 1* shows the output GPT-2 produces given books published by *J.K. Rowling*. We manually evaluated the output marking with an asterisk (*****) incomplete or incorrect titles, with a plus symbol (**+**) grammatically correct titles that we consider bad, and unmarked titles are ones that we consider as good for that author.

GPT-2 output for J.K. Rowling
* Harry
+ Harry Potter
* Harry Potter Power Ups in
Harry Potter Volumes
* Harry Potter and the
* Harry Potter and the Prisoner of Azkaban (Harry
* Harry Potter: The Half-blood Prince (Harry Potter
Harry Potter: The Time After Death (Harry Potter, #10)

Table 1. Output of the original GPT-2 model for the books of JK Rowling. * Incomplete title. + Not a good title.

We can see that GPT-2 is capable of recognizing patterns in the list of book titles we fed, but the output is still pretty noisy, and there are a lot of incorrect titles. We chose *J.K. Rowling* because she is famous for writing the *Harry Potter* books, and most of her books are named in a very similar way. However, for authors with a more diverse naming style, it becomes harder for the model to generate shorter sentences so we are forced to limit the output to a maximum of 20 words. If we take *George Orwell*, as we see in *Table 2* the model wants to generate longer text that does not sound like a plausible book title.

GPT-2 output for George Orwell
* For a Second time I was appalled to read the last five books of his
+ I, Book One: Crosswords
* Orwell's London Papers, Volume 1,
Syndrome: The Mystery of Anna Karenina, Alfred Hitchcock and Douglas
Temple in the Underground
* The Chullamier Sea by Wigan writer Leo Perrett for the
The Dictatorship of the United States

Table 2. Output of the original GPT-2 model for the books of George Orwell. * Incomplete title. + Not a good title.

Since the base model of GPT-2 is not capable of correctly generalizing the book titles we are feeding it and generate a good sounding book title every time, in the next section we will see how to teach a model to do exactly that, by fine-tuning our base model to the specific task of generating book titles.

2.2 Fine-Tuning our Model

Before we start training our model, we will create a dataset to store and tokenize all the English book titles we have in the GoodReads dataset.

To train the model we will use the Gradient Accumulation technique since GPT-2 is huge and we would run out of memory otherwise. After training for 5 epochs with a batch size of 16 and a learning rate of 2e-5, we can proceed to store the model and check if the output has improved. *Table 3* shows the output it produces for *George Orwell*, so these results can be directly compared with what we saw on *Table 2*.

Fine-Tuned GPT-2 output for George Orwell
Animal Farm: History of the Eighteenth Century
+ Bangalore, 2002
+ Beautiful Orgasm and America
Before the World was a Great Land
Chronicles of Richard Carrier
Coca-Cola: The End of Prohibition
Ranger's Stairs
Television and the 'Word of God'
Flower in the Dust

Table 3. Output of the fine-tuned GPT-2 model for the books of George Orwell. + Not a good title.

As we can see the performance of the model has improved a lot, as basically all of the titles it generates feel like plausible real book titles: there are practically no cases where the title is incomplete, almost all of them are short and they follow the typical naming style that we can expect of a book title.

Keeping in mind that these tables only show one random small sample of results, we can assure that the model is quite good at the task of generating realistic sounding book titles, and that by feeding just 10 random books of an author the model is able to create titles in a similar style to previous works. This fine-tuned model can be downloaded for other uses via this link.

2.3 Evaluation of the Generated Titles

Once we generate a list of book titles we want a way to separate the good from bad book titles. This task, however, proves to be extremely difficult since almost all the ouptuts the model generates use correct grammar[1], punctuation style, and they make sense as titles of books; we cannot use these things as filters. So evaluating if a title is good or bad becomes a subjective task where a very high knowledge of a language, literature, history... is required, making it completely out of our scope.

What we can do, however, is to see how similar a generated title is to already published books for the same author, so we can compare their similarity in topics and word usage. To see how similar two book titles are between each other, we can take advantage of the same model we are using to generate titles by the use of embeddings.

Word embeddings [3] allow us to represent words in the form of vectors that encode the meaning of the words. Words closer to each other in the vector space indicate that, usually, they are similar in meaning. Since language models work with the principle of encoding words, we can extract the weights the model produces for a given input as embeddings, which we can use to calculate distances in the vector space.

The model will return an embedding for each one of the words [2] that compose a sentence. By feeding the model with a book title, we can then

[1] Look at *Table 5* where it wrote *Statehood* as *Estatehood*

[2] Technically words can also be divided into tokens

average all the embeddings so we get a vector representation for a given book title.

As pictured in *Figure 1*, we can calculate an embedding to every book published by the author, and for each one of the titles generated by our model. We can then calculate the distance between two titles (vectors x and y) using the formula of cosine similarity, based on the angle two vectors produce:

$$\cos(x,y) = \frac{x \cdot y}{||x|| \cdot ||y||}$$

The value given by this formula is between 0 and 1, and a higher value indicates that the two vectors are closer together. We will take the highest value to any book title as an score for that generated title. Once we calculate the similarity score for every generated title, we can sort the list from highest to lowest value.

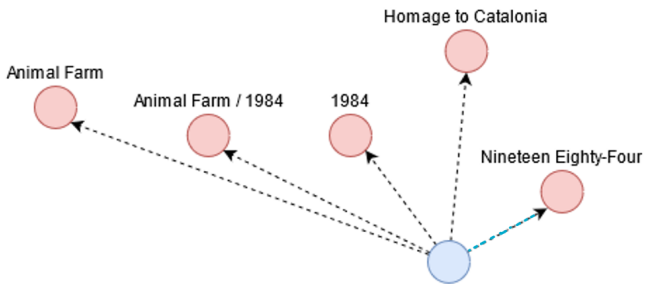

Fig. 1. Calculating similarity from a newly generated title (blue) to all published titles of an author (only some pictured in red). Highest similarity marked in blue.

2.4 Results

Tables 4 & 5 show the final output of our system: a list of book titles sorted by similarity. We can see that in general, it makes a lot of sense: the top elements of the list seem to be more relevant than the bottom ones, in the sense that they are more similar in topics and style to the books published by that author, and by that reason feel more familiar. This means that the embeddings we are using generated with GPT-2 work quite well to encode the message behind a book title.

Simil.	Output for George Orwell
0.68	The Life And Times Of Aldous Huxley
0.67	The Postal Service's Official Tales
0.67	Journey to the Bottom
0.63	Truth, Liberty, and Equality
0.62	Audacity, Simplicity, and Greatness
0.59	A Christmas Carol
0.56	Uncle Tom's Cabin
0.54	Analysis of Comparative Literature
0.47	Royal Wedding
0.46	Black Knights

Table 4. Sorted output of our system for the books of George Orwell

3 Conclusions

In this project we have seen how useful language models can be for the task of generating realistic text that could have been written by a human. We have seen not only how to use them, but also how we can take advantage of the benefits they offer, by fine-tuning a model for a specific task and using it to encode and compare the meaning behind the words the model itself generated.

Working in this project has been a very interesting and fun experience. Even though evaluating the titles generated by this system can be difficult and very subjective, choosing an author you really like, and watching how a computer can generate non-existing book titles that feel familiar can be a weird experience, but understanding the bases of how the system works makes it even more interesting, making you want to execute the code again just to see what title generates next.

It also makes you appreciate the work that goes behind these open-source projects, and the community behind them. Still, latest advances make it harder for ordinary people to use these kind of models; bigger models require an amount of resources not available for most users, so only big companies and corporations can benefit from these advances.

Similarity	Output for Noam Chomsky
0.73	The American Dream: From The Harlem Renaissance to the Vietnam War
0.71	We the People: Ten Underclass Struggle to Reclaim Our Democracy
0.70	About the Leader: The Politics of a Genuine Nation
0.69	A Short History of Psychology
0.67	A Colorful and Revolutionary Story
0.63	Estatehood: Federalism, Labor and Labor History
0.60	Socialism: Political and Economic Interpretation
0.57	The Shadowlands
0.43	Crossing Souls

Table 5. Sorted output of our system for the books of Noam Chomsky

4 Future Work

Even though we have managed to complete all the objectives we proposed at the beginning of the project, some possible improvements remain in the horizon.

1. We could also test if the model can predict titles of the next books in a saga by selecting a small sample of book titles, and then compare generated titles to the actual ones.

2. It could be interesting to move this idea into different domains, such as movie names, song titles... and compare if a model fine-tuned with book titles could be used to generate realistic sounding movie titles without having to fine-tune a new model. Experiments comparing different models in different domains could be performed to analyze how well these models can generalize.

3. We also saw that sometimes, cosine similarity seems to benefit longer titles. This could be a problem similar to the Hubness Problem, where some embeddings are closer to a high number of other embeddings. Using different metrics such as CSLS instead of the Cosine Similarity could alleviate this problem [4], so it would be interesting to see if the final results of our system differ using different metrics.

References

[1] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I., 2019. "Language models are unsupervised multitask learners".

[2] Wikipedia, 2021. GPT-2 — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/GPT-2. [Online; accessed 29-May-2021].

[3] Wikipedia, 2021. Word embedding — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Word_embedding. [Online; accessed 31-May-2021].

[4] Dinu, G., Lazaridou, A., and Baroni, M., 2015. Improving zero-shot learning by mitigating the hubness problem.