A GPT-2 based writing tool for novel authors
- Project Proposal -

Gaël de Léséleuc de Kérouara
gael.de-leseleuc@student-cs.fr

Alexandre Duval
alexandre.duval@student-cs.fr

Thomas Lamson
thomas.lamson@student-cs.fr

ABSTRACT

With this project, we aim to create an automatic advanced writing tool to help authors in their task. In fact, the final intended product is a web-service where authors can write text and ask for paragraph automatic generation based on the previous text, the following text, desired size/theme and an optional summary of the content to generate. To conduct this task, we first create a relevant dataset before fine tuning and combining several state-of-the-art models (BERT, GPT-2...). The main technical difficulty lies in finding a way to contextualize a GPT-2 model on structured and unstructured data, using other models to generate good training data. We will assess the performance of our model through qualitative tests with amateur authors.

1 PROBLEM DEFINITION

1.1 Motivations

The field of Natural Language Processing has undergone amazing progress in recent years, especially through the use of deep neural networks. One key idea at the origin of this improvement consists in embedding the signification of a particular word into a vector of scalars, of chosen dimension. This word embedding, as we call it, made it possible to perform a wide variety of tasks such as machine translation, language modelling or sentiment analysis much better than existing ruled or grammar based approach. In addition to reaching higher performance in the vast majority of tasks, it also launched a new era of research. More recently, the use of a new architecture named Transformer, especially via BERT[1], revolutionised the field, improving dramatically our ability to capture the meaning of a word, sentence or text.

Building on this, in Feb. 2019, OpenAI described a neural network strictly focused on text generation: GPT-2[5]. The quality of generated text was such that OpenAI team decided not to release their full model to the public, having concerns about its possible misuses, e.g fake news generation. A few months later, they finally disclosed it with one main objective: empower researchers to better detect synthetic text. Nevertheless, in our case, we would like to utilise it to produce something useful and ethical that the society could profit from, as some other people did. Indeed, some of them began to implement web-based demonstrations[1] or even an infinite textual game based on it: AI-Dungeon[2], which encountered quite a success.

However, even state-of-the-art text generation models falls short when it comes to maintaining a consistent context across long paragraphs ; or when it comes to generating text using a predefined context. Being aware of such limitations, we want to join the recent trend of research papers that try to overcome these shortcomings. In particular, we would like to fill in a gap in the literature by building a tool aimed at helping an author write a novel, mixing different existing approaches.

[1]https://talktotransformer.com/
[2]https://aidungeon.io/

1.2 Proposed concept

The tool that we would like to create will take the form of an interactive online website. An user would be able to specify:

(1) a part of the previous text
(2) a list of important named entities or a summary of the paragraph to generate
(3) a part of the following text
(4) a size descriptor
(5) the novel type (romance, horror, etc.)

Then, considering all this information, the tool will automatically suggest several texts (for instance five) that the user can choose from, and which will fill in the gap between the user's start and end text while also respecting the following constraints:

- constitute a global consistent text
- be of the specified length
- use the given named entities and reflect the given summary

In the end, at any point when writing their novel, the user will be able to automatically generate new sections, consistent with the rest of the writing, either to replace an already existing part of the text or to generate a new piece of it. We will leave them the opportunity to scroll among the suggestions and edit them to fit their version of the story. On the UX side, the tool will aim to be as transparent as possible, where highlighting any part of the text and pressing a "Generate" button or shortcut will be enough to trigger the generation of new suggestions.

1.3 Related works

A neural network for text generation usually works by computing the probability distribution of the next word based on previous words:

$$P(x_i|x_1, x_2, ..., x_{i-1}) \tag{1}$$

What we want to do is to condition this probability on a structured and unstructured context that will be specified by the user:

$$P(x_i|x_1, x_2, ..., x_{i-1}, context) \tag{2}$$

The same issue arises in a lot of topics like translation or summarization. In our case however, the context does not simply involve an input sentence but rather a combination of structured (theme, length of text to generate) and unstructured features (summary sentence, previous and following text).

A first idea to address this problem would be to use a standard encoder/decoder structure. For instance, NaverLab's team used a transform encoder-decoder network to address the problem of generating text from structured data[6].

A second idea would be to take only the GPT-2 decoder framework, similarly to what Grover's team did to generate and discriminate fake news[7]. Their work is actually quite close to ours from the context perspective as they used information about domain, author, date and a small summary of the article to generate it.

One issue here, as evoked in these papers, is the computational resources needed to train these big models. Concerning Grover for instance, the training time lasted 2 weeks with 256 TPUs. Since we are very limited in this domain both in time and resources, we will not be able to train from scratch a powerful model that produces believable sentences. We consider that our best chance is actually to fine-tune existing pre-trained models and to adopt an original combinations of approaches.

1.4 Considered technical solution

Our model will use a fine-tuned version of GPT-2, hopefully able to perform well on our task when given at least a part of the inputs detailed above. We consider this network for two main reasons:

(1) GPT-2 is a decoder, hence able to generate text, and is already producing meaningful sentences
(2) It has shown great results on several tasks with 0-shot setting and can preserve a context in the long term

Moreover, as Grover [7] demonstrated, it seems to be possible to condition the generation on a context even if GPT-2 is just a decoder. The idea is pretty simple: use special tokens to structure the pre-filled part of the sequence and insert the input data.

$$P(x_i | context_1, ..., context_n, x_1, x_2, ..., x_{i-1}) \tag{3}$$

By doing so, when computing the probability distribution for token x_i, GPT-2 model will have access to the context information, while our fine-tuning will have made it able to understand the tokens structuring the data.

The main objectives of our model are as follow:

(1) Produce a syntactically correct sentence
(2) Fill consistently the gap indicated by the author, conforming with the specified size and theme.
(3) Respect the context of surrounding sentences
(4) Use the named entities provided by the user

In the previous section, we explained that the author might specify a short summary of the paragraph to produce, conveying the main ideas of it. However, it can seem pretty difficult to find some data matching our specific case: fantasy writing where every paragraph has a short summary and/or a list of important named entities. We will discuss in the Data section how we will address this issue by using other models (BERT-NER, summarizers...) to generate this data from any novel dataset.

2 METHODOLOGY

2.1 Dataset generation

To acquire the textual data, we planned to download plenty of public-domain books from Project Gutenberg [3] and Open Library[4]. Those websites already classified each book by theme (horror, romance, etc).

Then, for each book, we will :

- browse the entire content
- split it into paragraphs of varying lengths
- generate triplets of contiguous paragraphs: P_1, P_2 and P_3
- apply Name Entity Recognition to extract character's names, locations and other key elements of the story
- (eventually obtain a summary of P_2)
- store each triplet (P_1, P_2, P_3) and associated information in a list of samples

When feeding the data to GPT-2, it will actually look like this:

```
<theme-horror> <size-long> <entities> Entity1
Entity2... <p1> text <sp2> summary <p3> text <gen>
```

Where <...> marks are special tokens indicating to GPT2 what information needs to be taken into consideration for the text generation task (book genre, size, existing entities, previous paragraph, next paragraph...). However, to make our model more robust, we will not always use all available tokens, as explained in the next section. The <gen> token indicates the beginning of the text generation.

2.2 GPT-2 fine-tuning

The next step consists in fine-tuning a large GPT-2 model (763M parameters) on the data formatted above. Sometimes, we remove one token between <sp2>, <p3> and <entities> as the author might not always give them. It also trains our model to be more robust. We fine-tune GPT-2 to product the exact same sequence as an output, but we add the original content of P_2 after the <gen> token. Therefore, the generator has to let the first part untouched and has to guess what the content of the original paragraph was. Practically, we will compute the cross-entropy loss between the output embedding vectors of the generator and its word embedding, capturing the distance between the generated vector and the vector representing the target word.

To propose several sentences to the author, the technique we will employ still has to be determined. Intuitively, we would like to exploit the feature of GPT-2 that allows it to avoid getting stuck in repetitive loops, that is to consider more than the best option (linked to beam search). Thanks to this, we could output a top k-samples. On top on that, we are thinking of clustering them by meaning or style using BERT and selecting the best paragraph (according to a metric like perplexity) per cluster. This will hopefully allow us to output suggestions that are quite different from each other, but yet all believable.

This final part remains open however as it mainly depends on the performance of our generator and our final users appreciation. We will experiment with different values and techniques, trying to confront our product with real authors as much as possible.

2.3 Summaries generation and use

Acquiring data for which every individual paragraphs have been summarized is really difficult. However, we thought of a technique that would let us generate such dataset from raw novel text, free of

[3]https://www.gutenberg.org/
[4]https://openlibrary.org/

any training. We would like to use various state-of-the-art summarizers such as BERTSUM[3], UniLM[2] or Refresh[4] (both abstractive and extractive) to elicit the essential content of the paragraph of interest P_2. Once this paragraph is summarized by the different summarizers, we would chose any one of them (either randomly to have different "styles", or using a metric detailed in the Evaluation section) and feed it into our GPT-2 model as <sp2> input.

This process is likely to make our model more robust to the type of summaries received. It will also probably make the generator's task easier as these summaries convey important information about what should be written. Lastly, it feels consistent with the final aim of the project as authors often want to keep control of the content being generated, and its underlying ideas.

To train our GPT-2 generator more specifically on the "de- summarization" task, we will also fine-tune it for some steps on a much smaller input:

```
<sp2> summary <gen>
```

3 EVALUATION

There are three main steps where our work has to be assessed to ensure its quality along all the process.

- Data generation
- Generator fine-tuning
- Final product

3.1 Evaluation of generated data

As most of our data will be generated through other NLP models, it is important to validate its quality. More specifically, we want it to be as representative as possible of what a real author could do.

An important focus has to be driven towards the summarization task as those summaries will convey the most information about the paragraphs to generate. The main question is: would a human be able to produce a paragraph containing the same information as the original one from this summary and the context, as we are asking our generator to do. However, it is also interesting to note that if the summaries are not perfect, it might make our generator more robust, hence working better with real summaries from authors.

Practically, is it hard if not impossible to check at the time if the summary contains the same ideas as the original paragraph, although we thought of one technique that could assess the contextual proximity of those two sequences: by feeding them separately in BERT and comparing the final output vectors together, we can compute a contextual distance. On a similar note, we will make use of the ROUGE metric. This way, we might be able to measure the performance of several summarizers.

It will also be essential to take a human look at these summaries in a wide range of writing styles.

3.2 Evaluation of the generator

As explained in the fine-tuning section, we will use a cross-entropy loss between the generated output and the original paragraph. A first way to evaluate our model is simply to run it in evaluation mode on a validation set of samples, which we will do. We will then compare it to simpler architectures: non fine-tuned GPT-2 on same input, non fine-tuned GPT-2 on previous text only, other generators, etc. through the same metric.

A few other points to check considering our criteria are the length of the generated paragraph and the use or non-use of listed named entities. Even though our loss remains the same, it is especially interesting to see if named entities are always added to the paragraph or not and if length requirement is respected most of the time. Moreover, after generating several sentences thanks to GPT-2, it will be possible to apply hard-coded filters, checking for example these metrics. It is therefore important to know how often these constraints are fulfilled by the generator.

3.3 Evaluation of the final product

With the trained model and a way to select the best suggestions, we need to test our tool with real authors, if possible from outside our team. We are in contact with several English-writing amateurs working on the web-platform WorldAnvil[5] on their own different universes, all passionate about fiction writing. We decided to make our service available on the web during the test phases to let writers from any country try it and give us feedback. However, during the time of the project, we will focus on qualitative feedbacks with chosen authors.

This process will hopefully lead to enriching iterations and will help us build a more promising tool.

4 CONCLUSION

This project represents a great challenge for us as it would make use of the most advanced NLP algorithms available nowadays in a novel and useful way. Indeed, we have not found any paper describing what we plan to do. So we look forward to contributing to the NLP community with this research. Moreover, if the results are promising, this could represent a great opportunity as this kind of tools might have a lot of applications, from novel writing to articles, news writing, etc. It seems clear to us that it answers a real need.

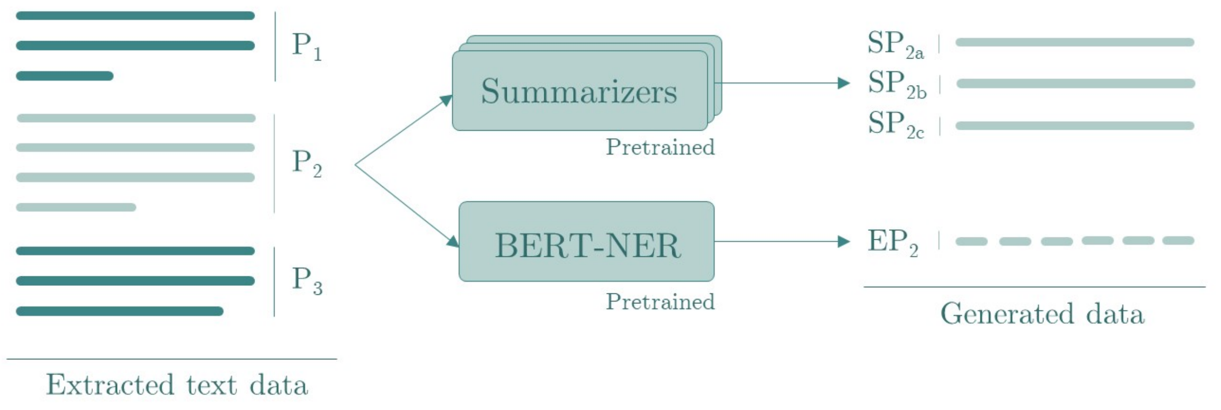
We hope that we convinced you about both our determination and our ability to reach our objective. We are and remain fully open to any advice, critics or questions that would help us progress in this project and avoid pitholes we might have missed.

REFERENCES

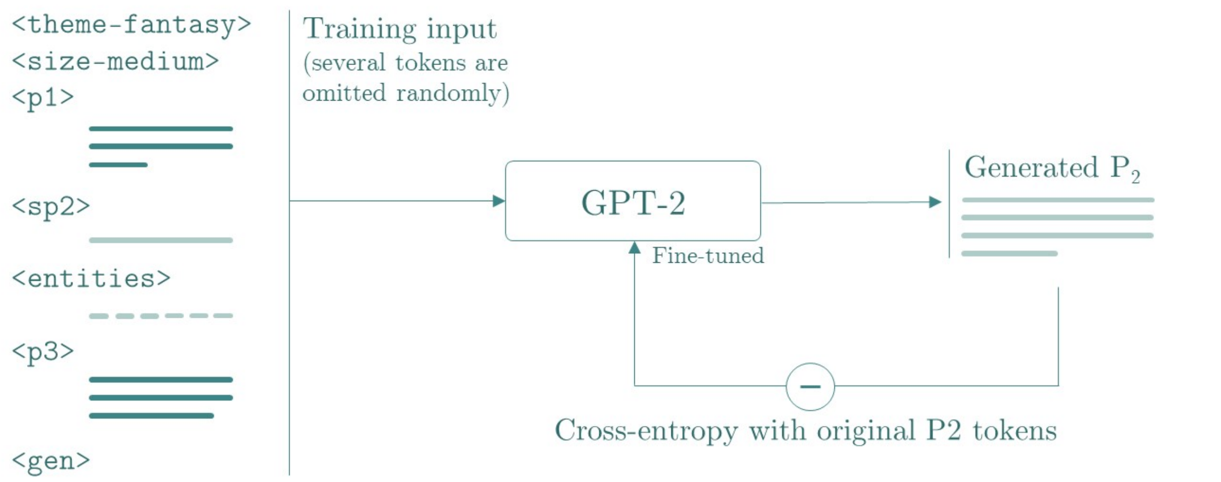
[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805
[2] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. (2019). arXiv:1905.03197
[3] Yang Liu. 2019. Fine-tune BERT for Extractive Summarization. (2019). arXiv:1903.10318
[4] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking Sentences for Extractive Summarization with Reinforcement Learning. (2018). arXiv:1802.08636
[5] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language Models are Unsupervised Multitask Learners. (2018).
[6] Fahimeh Saleh, Alexandre Bérard, Ioan Calapodescu, and Laurent Besacier. 2019. Naver Labs Europe's Systems for the Document-Level Generation and Translation Task at WNGT 2019. arXiv:cs.CL/1910.14539
[7] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending Against Neural Fake News. (2019). arXiv:1905.12616

[5]https://www.worldanvil.com/

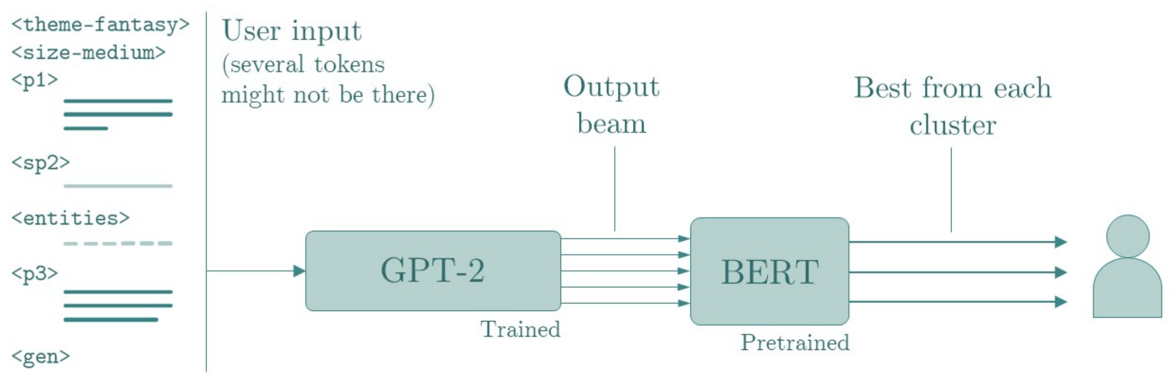
DATA GENERATION



FINE-TUNING



EVALUATION


<theme-fantasy>
<size-medium>
<p1>
<sp2>
<entities>
<p3>
<gen>
User input
(several tokens might not be there)
Output beam
Best from each cluster
GPT-2
Trained
BERT
Pretrained