Can innovative prompt engineering with ChatGPT address imbalances in machine learning datasets?

Mateusz Kochanek¹, Przemysław Kazienko¹, Jan Kocoń¹, Igor Cichecki¹, Oliwier Kaszyca¹, and Dominika Szydło¹

¹Affiliation not available

December 22, 2023
Can innovative prompt engineering with ChatGPT address imbalances in machine learning datasets?

Mateusz Kochanek, Przemysław Kazienko, Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Dominika Szydło

Abstract—Large language models are experiencing a significant surge of attention and rapid development. It is happening mainly due to the publication of OpenAI’s ChatGPT models: GPT3.5-turbo and GPT-4. This article uses prompt engineering to present an innovative approach to synthetic data generation and knowledge distillation. Specifically, we focus on three methods: basic prompts, composite prompts, and similarity prompts. This research aims to investigate the potential of these techniques to address the problem of unbalanced datasets, a common issue in machine learning applications. Experimental results reveal that none of the prompt-based strategies achieve scores on par with the entire dataset. However, the similarity prompts method shows promising potential, outperforming other approaches. The study suggests a significant opportunity to develop these techniques further to generate more diverse synthetic data. Although the results are preliminary, they open up exciting possibilities for future research in this area, including integrating more advanced versions of Large Language Models and exploring other machine learning domains.

Index Terms—ChatGPT, Natural Language Processing (NLP), sentiment analysis, unbalanced datasets, text classification, large language model, prompting

I. INTRODUCTION

In recent months, the field of artificial intelligence has seen substantial advancements, particularly in the field of natural language processing (NLP). Generative language models like GPT (Generative Pretrained Transformers) have attracted significant attention among these developments. ChatGPT’s remarkable ability to understand and generate text that mimics human language, combined with easy access from the public domain, is the reason for this. These models have shown astounding effectiveness in numerous tasks, including text generation, translation, summarisation, and sentiment analysis. However, creating these sophisticated models is a resource-intensive process. It requires large amounts of computational power and the collection and storage of massive volumes of data. The need for extensive computational resources and storage capabilities poses significant challenges, especially considering the environmental impact and the high financial costs associated with power consumption and data storage. Only the most prominent organizations can afford such costs, making the research process difficult.

That is why it is essential to think about how these vast models can be used effectively via APIs or web applications, without the need to train them from the ground up. In addition, as promising as these models are, their application is not without challenges. A primary issue is controlling their output effectively and consistently. Ensuring that the generated text aligns with the desired goal is a nontrivial task, given their complexity and the vast solution spaces they operate in.

In response to this issue, the concept of using prompts to guide the model’s output has emerged. Prompts serve as a compass guiding the model to generate output that better aligns with the user’s objectives. However, while intuitive and effective, the science of prompt engineering remains largely uncharted territory. How different prompts influence the model’s behavior is under active research and development.

This article is set within this context. We explore prompt-based data generation techniques with GPT-3.5-turbo, a language model developed by OpenAI. ChatGPT is built upon the GPT foundation but includes unique features and capabilities designed specifically for conversational interactions. By delving deep into the potential of prompts, we aim to contribute valuable insights and methodologies that can help harness the power of these models more effectively, paving the way for their broader and more beneficial use in numerous applications.

Our main contributions include:

- Conceptualization and development of three prompting methods using prompt engineering.
- Methodically evaluating how effective different prompt patterns are in addressing the imbalance in the dataset for sentiment analysis tasks.
- Investigation of the impact of the prompt engineering method on the quality of synthetic data generated by GPT3.5-turbo.
- Creation of three synthetic negative review datasets.

II. RELATED WORK

Large language models have recently become a highly active area of research in AI, mainly due to the release of the ChatGPT chatbot [1]. This development has had a profound impact not only on the research community but also on the general public. It has spurred the evolution of prompt engineering methods for such models and offered a new perspective on knowledge distillation. This section presents an overview of potential resources for work involving large language models like ChatGPT [1] from OpenAI or Bard [2] from Google.

A. ChatGPT

Developing autoregressive models based on Transformer architecture [3] has been a fascinating journey marked by several significant milestones.

In the context of ChatGPT, the first significant step came in 2018 with the Generative Pre-Trained Transformer (GPT-1) model [4], which could generate coherent sentences and paragraphs. Next came GPT-2 [5] and GPT-3 [6], offering
more parameters and much better performances than their predecessors. The GPT models set new standards for the scale and performance of language models.

An interesting variant of these models, InstructGPT [7], was fine-tuned based on human feedback. Remarkably, InstructGPT (1.8B parameters) produced outputs that the labelers evaluated as better than GPT-3 (175B parameters). This method was further refined in ChatGPT [1], an interactive conversational model trained with even more human feedback. Finally, GPT-4 [8] represents the latest evolution in large language models from OpenAI. It is a large-scale multimodal model capable of processing text and image inputs. Furthermore, OpenAI has recently introduced ChatGPT plugins [9], such as connecting GPT-4 to the Internet. It makes the model highly responsive to recent events and new information.

As the complexity and capabilities of these models have grown, so have their potential applications. The ability to generate coherent and contextually relevant text opens up numerous possibilities. GPT models are already used for synthetic data generation as seen in [10], [11] or data augmentation in [12]–[14]. Furthermore, researchers test the capabilities of these models to solve multiple different NLP tasks, presented in [15]–[17].

Although these advances are promising, they also raise a number of practical and ethical considerations. Ensuring responsible and equitable use of these technologies will be a substantial challenge in the future, as can be seen in [18]–[20].

However, it should be noted that most research concerning data generation or augmentation has primarily involved GPT-3 and GPT-2 models. Additionally, prompt engineering could become much more important in the coming months. In this article, we tested multiple methods of generating data with GPT-3.5-turbo.

B. Prompt Engineering

Prompt Engineering represents a crucial aspect of working with large language models such as GPT-3.5 and GPT-4. It involves designing and optimizing prompts given to these models to generate the desired responses, as shown in Figure 1. This section provides an overview of recent research developments in prompt engineering. We also point towards areas that require further investigation, highlighting the ongoing challenges and opportunities in this rapidly evolving field.

There is already interesting research on multiple types of prompt patterns collected as a comprehensive catalog [21]. These patterns are useful for researchers and users of LLM chatbot technologies. An example of a pattern called "The Persona Pattern" can be seen in Figure 2. It gives the LLM a 'persona' that assists in determining the output format and the specific details to prioritize. Some pattern catalogs are more specialized, as in White et al. [22], where the focus was on prompts for tasks like code generation, code refactoring, or software design.

Out of these patterns, a new paradigm in natural language processing is emerging, widely known as prompt-based learning. In [23], the authors argue that this framework is very powerful, as it allows the language model to be pretrained on massive amounts of raw text. Next, by defining a new prompting function, the model can perform a few-shot or even zero-shot learning, adapting to new scenarios with few or no labeled data. This approach was used for multiple nlp tasks shown in the work [15]. Some more specific use cases include relation classification [24] or dialogue systems [25].

Although the opportunities for progress in prompt engineering are vast, the field also presents unique challenges that must be addressed. It is a difficult task to identify the prompts that lead to the desired outputs. Even if the prompt seems to work, there is an inherent challenge of unpredictability of model responses. Even with well-constructed prompts, there is a non-negligible chance that a model might generate off-topic or uncoherent responses, complicating the task of ensuring quality control. This issue became particularly apparent with prompts that specified the output as a Python list. These prompts got much worse outputs than prompts specifying the output format as JSON.
C. Knowledge Distillation

Knowledge Distillation was introduced in the research paper by Hinton et al. [26] as a procedure where knowledge of a large model is distilled into a smaller student model. Of course, there were works before and after, where a small student model was generally supervised by a large teacher model [27]–[29]. This technique should result in a lightweight and computationally efficient student model that mimics the teacher’s behavior by assimilating the extracted knowledge. It is a broad and crucial field of artificial intelligence.

![Diagram of the classic knowledge distillation framework.](image)

**Fig. 3:** Scheme for classic knowledge distillation framework; based on [30].

An excellent survey on knowledge distillation is available by Gou et al. [30], which provides a comprehensive look at current developments and challenges in this field. It discusses different types of knowledge to distill, such as response-based [26], [31] that use logits from a large model, feature-based [32]–[34] that use features of intermediate layers, and relation-based [35]–[37] that use relationships between different activations, neurons or pairs of samples. The survey discusses different distillation schemas, teacher-student architectures, and algorithms used for knowledge distillation.

Some additional notable examples of this technique involve the use of the distillation process for commonsense models [38] or the improvement of the few-shot ability of small language models [39]. In the first example, the authors used prompt engineering and a separately trained critic model to distill high-quality causal commonsense from GPT-3 selectively. They surpassed a human-authored commonsense knowledge graph with their automatically distilled variant. The second study demonstrated that prompt-based distillation can improve performance on few-shot datasets.

In the context of our article, the concept of knowledge distillation is more akin to the extraction of knowledge from the GPT-3.5 model. But, as already shown, prompt engineering can and will most certainly impact this research area in the future. That is why taking a closer look at this process after recent breakthroughs in large language models may be beneficial.

III. KNOWLEDGE DISTILLATION USING PROMPTS

Our work utilizes the IMDB [40] movie reviews dataset, which we describe in detail in IV-A. We generated synthetic movie reviews using the official OpenAI API, specifically by employing the GPT-3.5-turbo model used by ChatGPT. We set the temperature of the chosen model to 0.8, as it is commonly assumed that ChatGPT has a model temperature set to be around 0.6 - 0.8. The procedure consisted of dataset analysis, prompt engineering, prompt generation, querying the API, results processing, and dataset rebalancing, as shown in Figure 5.

During prompt engineering, we carefully crafted the queries to elicit the desired responses from the model. During the prompt generation phase, we prepared the prompt in JSON format for basic prompts, or we generated the prompts and saved them to a CSV file for composite and similarity prompts. We queried the API by sending requests with prompts to the OpenAI API and retrieving the model’s answers. Finally, we processed and prepared the resulting synthetic reviews for inclusion in the dataset.

We proposed and considered three different prompting methods:

1) **Basic prompts** – a simple, single prompt that generates three negative reviews about an unspecified movie.

2) **Composite prompts** – different prompts generated from sentences and probabilities specified in the Listing 2, providing better diversity in the generated dataset.

3) **Similarity prompts** – different prompts that ask for reviews similar to one from the original dataset.

### A. Basic Prompts

We designed a basic prompt as a simple and straightforward way to generate data. We show its structure, which aligns with the OpenAI API requirements, in the JSON listing 1.

**Example of negative movie reviews from basic prompt:**

**Negative review 1:**

_The special effects were cheap and unconvincing._ The dialogue was clichéd, and the pacing was slow. I checked my watch constantly, waiting for the film to end.

**Negative review 2:**

_The acting was terrible, the special effects were cheap and unconvincing,_ and the script was full of plot holes. Save your money and skip this film.

Fig. 4: An example of movie reviews generated by using basic prompt shown in listing 1. We can see that both reviews have the same sentence.

The system message helps set the behavior of the assistant, even if, in the OpenAI documentation, it was written at the time that the GPT-3.5-turbo model does not pay strong attention to system messages. The user message is the main instruction that explicitly defines the task for the model and outlines how it should structure the output. Two reviews generated using basic prompts are presented in example 4.

It is crucial to note that the reviews created by this method have pretty obvious similarities. For instance, in example 4, we can see that the sentence “The special effects were cheap and unconvincing” is identical in both reviews despite them being generated in separate API calls.
ChatGPT dataset rebalancing diagram

Fig. 5: Flow diagram showing the three stages of dataset rebalancing with ChatGPT: 1) analyzing the dataset and choosing classes for synthetic data generation; 2) choosing prompting strategy, querying (prompting) the ChatGPT service and extracting answers; 3) rebalancing the dataset by integrating extracted answers.

**Listing 1** Basic prompt in JSON format, used as the input for OpenAI API

```
[
  {
    "role": "system",
    "content": "You are a negative review generator. Return your output as a json, where every review is one element. Output should look like this: \"\"review1\" : \"", \"review2\" : \"", \"review3\" : \""). Do not explain yourself."
  },
  {
    "role": "user",
    "content": "Create three film reviews about the same film. Film can be real or imaginary. Reviews must be negative. Return your output as a json, where every review is one element. Output should look like this: [\"review1\", \"review2\", \"review3\"]
    Do not explain yourself."
  }
]
```

The described method undeniably provides a quick, cheap, and straightforward way to generate synthetic data. Unfortunately, it has limitations. Given its simplicity, it does not allow for a high degree of control over the content of the generated reviews. Furthermore, as we represent in example 4, the basic prompt method can lead to repetition in the content of the reviews. A lack of diversity in training data could massively limit the robustness of machine learning models trained on such datasets.

**B. Composite Prompts**

We proposed composite prompts as a natural progression from the basic prompting method. Unlike basic prompts, which typically consist of straightforward, singular requests, composite prompts integrate various elements within a single instruction set. The schema for the composite prompt method is shown in example 6, illustrating how these components form a cohesive and effective prompt.

**Composite prompt structure:**

```
{film sentence} {attitude sentence} {additional sentences} {length sentences}
```

Return your output as a Python list, where every review is one element in that list. Output should look like this: ["review1", "review2", "review3"]
Do not explain yourself.

Fig. 6: A schema that was used for the composite prompts generation. Listing 2 presents specific sentences

The core of this method involved choosing sentences based on probability to parse into variables of the schema structure. Sentences and their associated probabilities were presented in the listing 2. It is also important to note that one of the film sentences has an empty variable that we filled with randomly chosen movie titles from the IMDB dataset. This element introduces an element of randomness and unpredictability, further mirroring the natural variability found in human-generated content.
These prompts offer better control over the content of generated reviews. The ability to incorporate specific sentence structures and film titles into the prompts adds an extra layer of realism to the generated reviews. Due to that, this method provides a greater level of diversity in the generated training dataset. Two examples of prompts and model outputs can be seen in Figures 7 and 8.

**Composite prompt example 1:**

Descriptive part of the prompt:

Create three film reviews about the same film. Film can be real or imaginary. Reviews must be negative and sarcastic. Focus on bad cinematography.

Negative review:

The film was so badly shot that I wondered if the cinematographer was a blindfolded toddler. The camera movements were so shaky that I thought I was watching an earthquake simulation. The lighting was so terrible that I wanted to donate a lamp to the production team.

**Fig. 7:** First example showing a generated part of a prompt and one of resulting reviews

We gathered composite prompt data before basic prompt data. This is why the output structure of the composite prompts is specified as a Python list. During testing, it became apparent that the JSON format is easier for GPT models to understand and format correctly. This change demonstrates the iterative nature of developing methods for synthetic data generation as improvements and optimizations are identified and implemented over time.

Although the composite prompt method offers improved control and diversity compared to the basic prompt method, it has limitations. For example, the generated reviews may still have some repetitiveness due to the predefined set of sentences used in the prompts. Future improvements could explore methods for dynamically generating or selecting diverse sentences to incorporate into prompts, reducing the risk of repetitive content. By refining the method for choosing probabilities, movie titles, or even some specific actors, we could enhance the overall quality and realism of the generated reviews.

**Composite prompt example 2:**

Descriptive part of the prompt:

Create three film reviews about the same film. Film can be real or imaginary. Reviews must be negative but ambiguous. Focus on bad CGI effects.

Negative review:

The film had potential, but unfortunately, the CGI effects were distracting and took away from the overall experience. It’s a shame because the storyline was intriguing, but the poorly executed effects were a major letdown.

**Fig. 8:** Second example showing a generated part of a prompt and one of the resulting reviews

C. Similarity Prompts

The Similarity Prompt method represents another evolution in strategies we developed for synthetic data generation. The main feature of this method is providing an example of a negative review from the under-represented class to the generative model. We illustrate the prompts and the resulting output in the example 9.

This method provides a way to generate synthetic reviews that more closely align with the original dataset’s domain. Providing an example review of the model allows it to mimic the sentiment of the chosen review and its style, tone, and vocabulary. In this way, generating data that better retains the complexity of original reviews may be possible.

Using this method, we chose samples from an already artificially reduced dataset to generate data. This approach mitigated the risk of data leakage and is the closest way to simulate real-world scenarios. Additionally, we chose JSON as the output format for this method, aligning with the findings from previous methods that showed that JSON is a more effective format for GPT models to understand and format correctly.

Despite some undeniable advantages, this method also presents some challenges and limitations. Most importantly, it heavily depends on the quality of the original data used as the basis for generation. Furthermore, similarity prompts could potentially limit the diversity of the synthesized data. It heavily depends on the number of original reviews available and the amount of synthetic reviews needed.
Fig. 9: An example of a prompt and a resulting movie review generated by the model.

**IV. EXPERIMENTAL SETUP**

This section presents a detailed description of the experimental setup we used to gather results for this article. Implementation was coded in the Python language using various libraries to speed up the development process. We provide an overview of the entire experimental process used in this article. It should give the reader a comprehensive understanding of how these experiments were conducted, enabling them to understand better the results presented in the following sections. Figure 11 shows the general experimental setup.

**A. Dataset**

IMDB [40] movie reviews dataset is a collection of English reviews from the Internet Movie Database (IMDB) website labeled as positive or negative. On this website, users can write an opinion on a film they have seen and rate it on a scale from 1 to 10. In this specific dataset, reviews with a score of 4 or less are labeled negative, while those with a score of 7 or higher are considered positive. No more than 30 reviews are included per film because reviews for the same movie tend to have correlated ratings.

The dataset contains 50,000 labeled reviews, evenly split with 25,000 negative and 25,000 positive reviews. It is divided into training and test datasets, each comprising 25,000 entries. The train and test sets contain a disjoint set of movies.

**B. Experimental Design**

In this work, we divided experiments into three main groups: experiments on a complete dataset, experiments on an artificially unbalanced dataset, or experiments on a dataset that was artificially unbalanced and replenished to a balanced state with synthetically generated data. We used all these tests to see whether the designed methods of prompt-based synthetic data generation can be used to balance skewed datasets.

We assembled each experiment in a controlled manner. First, we prepared the training dataset. The first 10%, 20%, 30%, or all original negative review samples, depending on our chosen value. For the unbalanced dataset, we took the same negative reviews into the training dataset for each experiment. This way made it easier to compare results from all experiments. Furthermore, we prepared five stratified folds (balanced or unbalanced) from the training dataset to ensure that every method was reliably tested. Next, for each experiment, we performed a 5-fold stratified cross-validation. We also evaluated the classifiers from each cross-validation iteration on the original IMDB testing dataset.

In the following listing, we described different experimental scenarios used to get the results shown in Table I. Each was repeated for BERT, Random Forest, and Naive Bayes models.

- **Full:** Baseline tests on a complete and unchanged IMDB training dataset.
- **Unbalanced 10% / 20%:** Baseline tests on IMDB training dataset with 10% or 20% of original negative class samples.
- **Basic 10% / 20%:** Setup with 10% or 20% of the original negative reviews left in training dataset. Negative reviews used for training were extended with samples generated using the Basic prompting method described in Section III-A.
- **Composite 10% / 20%:** Setup with 10% or 20% of original negative reviews left in training dataset. Negative reviews used for training were extended with samples generated using the Composite prompting method described in Section III-B.
- **Similar 10% / 20%:** Setup with 10% or 20% of original negative reviews left in training dataset. Negative reviews used for training were extended with samples generated using the Similarity prompting method described in Section III-C.

**Examples of movie reviews:**

**Positive review:**

Julie Andrews and Rock Hudson were great in this movie / musical. The opening song by Ms. Andrews, "Whistling Away the Dark," will always be in the back roads of my mind. The plot line during World War I is great and suspenseful. If you are a romantic, you will love this movie. This is a movie that I have always enjoyed seeing again and again.

**Negative review:**

This fanciful horror flick has Vincent Price playing a mad magician who realizes his vocational talents have been sold to another. He devises ways of avenging all those that have wronged him. His master scheme seems to backfire on him. Price is a little below par compared to his masterpieces, but it is still the only reason to watch this thriller. The supporting cast includes Patrick O’Neal, Mary Murphy, Eva Gabor, and Jay Novello.

**Examples of both positive and negative reviews from IMDB dataset:**

The movie Ned aKelly is a perfect example of how not to make a historical film. It fails to capture the essence of Australian history and its characters. The film looks like an American Western movie with kangaroos, and the portrayal of Ned with an Irish accent is completely disrespectful to Australian culture.

**Similarity prompt example:**

**Prompt:**

Create three film reviews about the same film. Film can be real or imaginary. Reviews must be negative. Base your reviews on this review: "Ned aKelly is such an important story to Australians but this movie is awful. It’s an Australian story yet it seems like it was set in America. Also Ned was an Australian yet he has an Irish accent...it is the worst film I have seen in a long time." Return your output as a JSON, where every review is one element. Output should look like this: {
"review1" : "Ned aKelly is such an important story to Australians but this movie is awful. It’s an Australian story yet it seems like it was set in America. Also Ned was an Australian yet he has an Irish accent...it is the worst film I have seen in a long time."
"review2" : ",", 
"review3" : ",".
Do not explain yourself.

**Negative review:**

The movie Ned aKelly is a perfect example of how not to make a historical film. It fails to capture the essence of Australian history and its characters. The film looks like an American Western movie with kangaroos, and the portrayal of Ned with an Irish accent is completely disrespectful to Australian culture.
C. Model Parameters

We created a BERT classifier using the \texttt{BertForSequenceClassification} class from the \texttt{transformers} library. We initialized it with pre-trained parameters from 'bert-base-uncased' with a number of labels equal to 2. We trained each classifier for four epochs with the learning rate parameter set to $2e^{-5}$. We chose Adam for an optimizer from the \texttt{Pytorch} [41] library.

Implementing the Random Forest and Naive Bayes Classifiers was imported from the \texttt{sklearn} [42] library with default training parameters.

V. EXPERIMENTAL RESULTS

In this section, we provide a detailed analysis of the experimental results presented to evaluate the effectiveness of prompt-based data generation strategies in dealing with unbalanced datasets. The focus is on determining which tested methods (Basic, Composite, or Similarity prompts) are best by comparing models evaluated on the IMDB test dataset. We illustrated all results in Table I. We additionally illustrated F1 score of the experiments in Figures 12, 13, 14.

A. BERT

The BERT model, known for its strong performance on various natural language processing tasks, lived up to expectations, demonstrating considerable efficacy on the complete dataset. This is evidenced by its highest scores on all metrics: F1 score, accuracy, and precision.

When considering the 10% unbalanced experiments, the "Composite 10%" configuration emerges as the top performer in the F1 score, as illustrated in Figure 12a. This suggests that composite prompts might be valuable when dealing with a significant class imbalance. However, a peculiar finding is that in the 20% unbalanced experiments, the unbalanced dataset itself, ironically, outperforms the other methods. This result raises an intriguing hypothesis: beyond a certain threshold, allowing the dataset to remain unbalanced might be preferable to introducing potential noise through synthetic data generation.

Although the differences between the various methods in both the 10% and 20% unbalanced setups are relatively small, it should be noted that the composite prompts performed well, emerging as the preferred choice for generating synthetic data when using the BERT model.

B. Random Forest Classifier

The performance of the Random Forest classifier diverges noticeably from the full dataset compared to the other unbalanced experimental setups. Interestingly, for both the 10% and 20% unbalanced datasets, the Similarity prompts approach outperforms the other methods, particularly evident in the F1 score, as shown in Figure 13.

The advantage of the similarity prompt method also extends to accuracy and recall metrics, as highlighted in Table I. On the contrary, the basic prompts method barely impacts the results in the 10% setup and degrades performance in the 20% setup. These results highlight the potential benefits of incorporating similar instances into the minority class when dealing with a class imbalance in a Random Forest model. On the other hand, the basic method barely has any impact and can make the results much worse.

C. Naive Bayes Classifier

The Naive Bayes classifier performs similarly to the Random Forest model outcomes. Again, the similarity prompts method achieved the highest scores across all metrics, suggesting the consistent effectiveness of this strategy across different types of classifiers. The composite prompts approach is second in terms of performance, followed by basic prompts. The experiment with an unbalanced dataset yielded the worst results, reinforcing the previous observation that leaving the data set unbalanced negatively impacted performance compared to data generation methods.

VI. ANALYSIS AND DISCUSSION

The experiments evaluated the effectiveness of the proposed prompt-based synthetic data generation techniques by checking how well they helped unbalanced datasets in the sentiment analysis task. We tested three proposed methods: Basic prompts, Composite prompts, and Similarity prompts in multiple configurations on three models: BERT, Random Forest Classifier, and Naive Bayes Classifier.

For the BERT model, in the case of 10% imbalance, composite prompts provided the best results compared to the unbalanced baseline. Unfortunately, its effectiveness was still far from the full dataset results. Interestingly, when the data set was in 20% imbalance, the unbalanced baseline produced the best results, suggesting that beyond a certain threshold, it may be better to leave the data unbalanced rather than add the generated samples.

The performance of the Random Forest classifier on the full dataset was far better than in other unbalanced experimental setups. Both unbalanced experiments, 10% and 20% – showed that the similarity prompts provided the best synthetic data. The composite prompts were slightly worse, and the basic prompts had minimal impact on the 10% imbalance, even degrading performance in the 20% imbalance scenario.
<table>
<thead>
<tr>
<th>Model</th>
<th>Experiment</th>
<th>F1</th>
<th>Recall</th>
<th>Precision</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>Full</td>
<td>0.922±0.013</td>
<td>0.905±0.043</td>
<td>0.941±0.025</td>
<td>0.923±0.010</td>
</tr>
<tr>
<td></td>
<td>Unbalanced 10%</td>
<td>0.842±0.024</td>
<td>0.988±0.005</td>
<td>0.735±0.040</td>
<td>0.814±0.035</td>
</tr>
<tr>
<td></td>
<td>Basic 10%</td>
<td>0.857±0.022</td>
<td>0.985±0.009</td>
<td>0.759±0.043</td>
<td>0.835±0.033</td>
</tr>
<tr>
<td></td>
<td>Composite 10%</td>
<td>0.865±0.022</td>
<td>0.981±0.009</td>
<td>0.775±0.038</td>
<td>0.846±0.028</td>
</tr>
<tr>
<td></td>
<td>Similar 10%</td>
<td>0.863±0.026</td>
<td>0.983±0.009</td>
<td>0.77±0.045</td>
<td>0.842±0.037</td>
</tr>
<tr>
<td></td>
<td>Unbalanced 20%</td>
<td>0.901±0.013</td>
<td>0.969±0.015</td>
<td>0.842±0.035</td>
<td>0.893±0.017</td>
</tr>
<tr>
<td></td>
<td>Basic 20%</td>
<td>0.887±0.024</td>
<td>0.978±0.013</td>
<td>0.814±0.049</td>
<td>0.875±0.032</td>
</tr>
<tr>
<td></td>
<td>Composite 20%</td>
<td>0.896±0.026</td>
<td>0.965±0.017</td>
<td>0.839±0.056</td>
<td>0.887±0.035</td>
</tr>
<tr>
<td></td>
<td>Similar 20%</td>
<td>0.891±0.015</td>
<td>0.978±0.007</td>
<td>0.819±0.033</td>
<td>0.88±0.019</td>
</tr>
<tr>
<td>RandomForest</td>
<td>Full</td>
<td>0.841±0.001</td>
<td>0.85±0.003</td>
<td>0.832±0.005</td>
<td>0.842±0.001</td>
</tr>
<tr>
<td></td>
<td>Unbalanced 10%</td>
<td>0.669±0.001</td>
<td>0.503±0.001</td>
<td>1.0±0.0</td>
<td>0.506±0.001</td>
</tr>
<tr>
<td></td>
<td>Basic 10%</td>
<td>0.67±0.001</td>
<td>0.504±0.001</td>
<td>0.998±0.000</td>
<td>0.509±0.003</td>
</tr>
<tr>
<td></td>
<td>Composite 10%</td>
<td>0.675±0.003</td>
<td>0.514±0.004</td>
<td>0.985±0.001</td>
<td>0.526±0.008</td>
</tr>
<tr>
<td></td>
<td>Similar 10%</td>
<td>0.681±0.004</td>
<td>0.521±0.005</td>
<td>0.986±0.001</td>
<td>0.539±0.001</td>
</tr>
<tr>
<td></td>
<td>Unbalanced 20%</td>
<td>0.692±0.001</td>
<td>0.529±0.001</td>
<td>0.999±0.0</td>
<td>0.555±0.002</td>
</tr>
<tr>
<td></td>
<td>Basic 20%</td>
<td>0.682±0.008</td>
<td>0.518±0.009</td>
<td>0.997±0.001</td>
<td>0.535±0.017</td>
</tr>
<tr>
<td></td>
<td>Composite 20%</td>
<td>0.692±0.013</td>
<td>0.535±0.016</td>
<td>0.983±0.002</td>
<td>0.563±0.028</td>
</tr>
<tr>
<td></td>
<td>Similar 20%</td>
<td>0.7±0.015</td>
<td>0.544±0.019</td>
<td>0.983±0.003</td>
<td>0.578±0.032</td>
</tr>
<tr>
<td>NaiveBayes</td>
<td>Full</td>
<td>0.811±0.002</td>
<td>0.857±0.003</td>
<td>0.77±0.004</td>
<td>0.821±0.002</td>
</tr>
<tr>
<td></td>
<td>Unbalanced 10%</td>
<td>0.667±0.0</td>
<td>0.5±0.0</td>
<td>1.0±0.0</td>
<td>0.5±0.0</td>
</tr>
<tr>
<td></td>
<td>Basic 10%</td>
<td>0.671±0.002</td>
<td>0.505±0.002</td>
<td>1.0±0.0</td>
<td>0.509±0.004</td>
</tr>
<tr>
<td></td>
<td>Composite 10%</td>
<td>0.679±0.005</td>
<td>0.514±0.006</td>
<td>0.999±0.0</td>
<td>0.527±0.011</td>
</tr>
<tr>
<td></td>
<td>Similar 10%</td>
<td>0.687±0.007</td>
<td>0.524±0.008</td>
<td>0.998±0.001</td>
<td>0.546±0.015</td>
</tr>
<tr>
<td></td>
<td>Unbalanced 20%</td>
<td>0.68±0.007</td>
<td>0.5±0.0</td>
<td>1.0±0.0</td>
<td>0.5±0.0</td>
</tr>
<tr>
<td></td>
<td>Basic 20%</td>
<td>0.694±0.013</td>
<td>0.532±0.016</td>
<td>0.997±0.001</td>
<td>0.559±0.029</td>
</tr>
<tr>
<td></td>
<td>Composite 20%</td>
<td>0.709±0.019</td>
<td>0.55±0.023</td>
<td>0.996±0.002</td>
<td>0.59±0.040</td>
</tr>
</tbody>
</table>

TABLE I: Table that shows results of carried out experiments. The table is divided into three main parts, each holding results for a specified model. Experiments per model are divided into three groups and are described with a keyword. "Full" means that a full dataset was used in this experiment. "Unbalanced" means that an unbalanced dataset was used. "Basic", "Composite", and "Similar" means that an unbalanced dataset was replenished with data generated by a specified type of prompt. 10% or 20% after the keyword specifies how many negative reviews from the original dataset were left in the training dataset.

The Naive Bayes classifier showed the most consistent results. Data generated with similarity prompts achieved the highest results across all metrics, with the composite prompts being the second-best approach. The basic prompts had a better result than the unbalanced dataset.

In general, composite and similarity prompts showed the potential to generate synthetic data when dealing with class imbalances. The choice of method may depend on the specific
model used and the degree of imbalance in the dataset. However, care should be taken not to introduce potential noise through synthetic data generation, as this could degrade performance.

Finally, it is worth noting that prompt-based synthetic data generation could be used in other machine learning domains beyond sentiment analysis. This method has enormous potential in image recognition, spam detection, and more. Future research should explore these possibilities in more depth.

VII. Conclusions and Future

This article presented the review of three prompt-based synthetic data generation methods using the GPT-3.5-turbo model: basic prompts, composite prompts, and similarity prompts. We found a formatting improvement from Python list to JSON during the synthetic data-gathering process with composite prompts. We prepared the following prompts with that finding in mind. We conducted the experiments to check to what extent generated synthetic data can be used to rebalance the chosen sentiment analysis dataset. We used the effectivity of the models trained on this dataset as a metric to measure which of the three methods has the most potential.

During the experiments, it became apparent that none of the proposed methods is advanced enough to attain results close to the full dataset. The Similarity prompts method provided the best results, but it scored 8 to 20 percentage points lower than the full dataset. Composite prompts looked promising, and basic prompts were useless, with only minor possible advantages, such as a simple structure and low cost. The output diversity is too low. Even if the generative model does not repeat generated data word for word, outputs are still very similar. That being said, it is important to note that even with only those three methods, there are improvements in metrics compared to unbalanced datasets.

Because of that, multiple research areas could help develop this approach further. First, the synthetic data generation methods can be greatly improved and combined. Composite prompts can easily improve their diversity by adding new
variables, and similarity prompts could be combined with composite prompts to generate diverse output that is better embedded in the task domain. Second, there are multiple ways to deal with unbalanced datasets that do not involve data generation. There is a possibility that if classic methods were combined with prompt-based approaches, the results could be better. Third, as said above, the prompts may generate data that is out of domain. But what if those data were not used as rebalancing samples but for dataset augmentation? It may improve the robustness of models and even increase the results achieved on different test datasets.

Another aspect of future work is the exploration of GPT-4 or future iterations of GPT models, as they are likely to provide improved performance and greater versatility in generating synthetic data. As language models become more sophisticated and their understanding of context and nuances deepens, more coherent and diverse synthetic data could be expected.

The present work offers a preliminary exploration of the potential of prompt-based synthetic data generation using the GPT-3.5-turbo. While the results have been modest so far, future studies are hoped to continue to improve these methods by expanding their application, thus contributing to the broader field of machine learning and artificial intelligence.

CRediT authorship contribution statement
Mateusz Kochanek: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing, Visualization. Jan Kocön: Conceptualization, Methodology, Investigation, Writing, Funding acquisition, Supervision. Przemysław Kazienko: Conceptualization, Methodology, Investigation, Writing, Funding acquisition, Supervision. Igor Cichecki: Conceptualization, Software, Writing: Review & Editing Oliwier Kaszyca: Conceptualization, Software, Writing: Review & Editing Dominika Szydro: Conceptualization, Software, Writing: Review & Editing

Acknowledgements
This work was financed by (1) the National Science Centre, Poland, project no. 2021/41/B/ST6/04471 (JK, PK); (2) the Polish Ministry of Education and Science, CLARIN-PL; (3) the European Regional Development Fund as a part of the 2014-2020 Smart Growth Operational Programme, projects no. POIR.04.02.00-00C002/19 and POIR.01.01.00-0288/22; (4) the statutory funds of the Department of Artificial Intelligence, Wroclaw University of Science and Technology; (5) the Polish Ministry of Education and Science within the programme "International Projects Co-Funded"; (6) the European Union under the Horizon Europe, grant no. 101086321 (OMINO). However, the views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency. Neither the European Union nor European Research Executive Agency can be held responsible for them.

References


