

Faculty of Engineering

Summer Research Project

EVALUATING THE DIVERSITY OF DIFFERENT LARGE LANGUAGE MODELS

Supervisor: Prof. Farzan Farnia & Prof. Chandra Niar

The Chinese University of Hong Kong, Hong Kong

August 2024

**Abstract**

*In recent years, the rapid advancement of large language models (LLMs) has revolutionized natural language processing, offering remarkable capabilities in generating human-like text. However, assessing the performance of these LLMs remains a critical challenge.*

*Here, we are interested in evaluating the diversity of outputs from different LLMs since it plays an important role in assessing the performance of the models. Unlike other criteria such as accuracy, the quantitative analysis of diversity to LLMs are often ignored. In this research, based on some existing evaluation metrics, we mainly focus on studying the diversity of some popular models, including different versions of GPT, Gemini and Claude. We will adjust a key parameter of these LLMs to study how diversity changes under different parameters. We will also study the relationship between readability and diversity of LLM output texts, and provide suggestions on how to choose suitable models appropriately to do specific tasks based on the research results.*

**Preliminaries and Background Information**

1. **Background Knowledges**
   1. **Large Language Models and Importance of Performance Evaluation**

Large language models (LLMs) are a class of artificial intelligence models designed to understand and generate human language. Built on deep learning architectures, particularly transformers, these models leverage vast amounts of data to learn complex patterns and nuances in text, then generate human language as outputs. LLMs have billions of parameters, which enables them to perform a wide range of tasks, including translation and creative writing. Their training involves unsupervised learning on diverse datasets, capturing a broad spectrum of linguistic features. However, the performance of LLMs is different, depending on their own mechanisms and datasets used.

The popular LLMs nowadays include GPT, Gemini and Claude. Despite the performance difference between them, various versions of the same model, for example GPT-3 and GPT-4, have different performance. More specifically, the performance of LLMs is defined by a group of metrics including accuracy, robustness, diversity, etc. These features perplex users when choosing a suitable model to finish tasks. For example, a model with better accuracy is more adaptable for translation, while a model with better diversity is more suitable for story writing. Moreover, understanding the performance of models is crucial for harnessing their full potential and guiding future innovations in the field.

* 1. **Diversity in generative LLMs**

A key challenge for generative large language models is diversity. Diversity in generative large language models refers to their ability to produce varied and contextually rich outputs from similar prompts. This characteristic is crucial for applications requiring creativity, adaptability, and natural interaction. Diverse outputs prevent redundancy, making models more effective in tasks such as dialogue systems, content creation, and brainstorming. when the user's prompt is not explicitly specified, the model may follow implicit assumptions when generating responses, which may lead to homogeneity of responses.

The diversity of LLMs is influenced by multiple factors, including model architecture, training data, and decoding strategies such as sampling and directed search. Compared to coherence, diversity is underestimated in development of many LLMs, as coherence is considered to be the key factor of human languages. However, striking a balance between diversity and coherence remains challenging. Overemphasizing diversity can lead to incoherent or off-topic responses, while insufficient diversity can lead to repetitive outputs. Evaluating and improving diversity involves using metrics that capture lexical variation and semantic richness to guide the development of more general and powerful language models.

1. **Previous works**
   1. **Human Evaluation**

Evaluating the diversity of output from LLMs can be a tricky work. Most of the models available these days are probabilistic in nature and are often seen to hallucinate factual information when generating text. So, one way to perform the evaluation is to do it with the help of humans. Testers can set specific questions as input and observe the consistency of the output text [1]. But this might not always be feasible as it is a costly and time-consuming process. Also, it does not guarantee re-producibility because of the sampling bias. When faced with a large amount of output text, human evaluation will not be a good option as it will consume a lot of time and manpower costs.

* 1. **Automatic metrics**

Following the disadvantage of human evaluation process, automatic metrics are usually preferred when evaluating text generation systems. Automatic metrics could be both trained and untrained. Trained metrics try to take into account the task specifics when doing the evaluation but at the cost of learning this function with relevant data. Untrained metrics are generic. They require no training process and are language-independent, which makes them faster to calculate. Untrained metrics are popular and widely used to measure the quality of generated text in industry and academia, covering many use cases such as machine translation, text summarization, story generation, image captioning, etc. Below are some of the famous automatic metrics related to diversity evaluation.

**2.2.1 Bilingual Evaluation Understudy (BLEU) [2]**

Bilingual Evaluation Understudy (BLEU) is a widely used metric proposed in 2002. It is an automatic machine translation evaluation method that is fast, cheap, language-independent, highly correlated with human evaluation, and has a small marginal cost per run.

BLEU is defined as the degree of similarity between machine-generated text and human-generated text. That is, the closer the machine translation is to a professional human translation, the higher the quality. BLEU is calculated by counting the n-gram overlap between the candidate and the reference, that is, the accuracy of the n-gram.

Although BLEU score was designed to compare the similarity of machine translation text and human language, it can also be used to evaluate the diversity of given texts against reference text. When BLEU score of a sentence is low, it has low quality in terms of translation, but high diversity as it shows a significant difference with the reference texts.

However, BLEU is not suitable as an indicator for judging the diversity of generative LLM outputs. When calculating the BLEU score, a large amount of reference text with similar meaning to the original text is required. However, when the output of the generative LLM is random, BLEU will not work without reference text. In addition, as it works by simply counting n-gram overlap between the texts, although it provides a simple and general

measure, it fails to account for meaning-preserving lexical and compositional diversity. In the following article, we will introduce an implementation of BLEU, the role of self-BLEU in judging the diversity of random generative language model outputs.

**2.2.2 BERT Score [3]**

While the commonly used methods in automatic evaluation of generative LLMs, like BLEU which counts n-gram overlap between the candidate and the reference, rely on surface-form similarity only, BERT Score aims at accounting for meaning-preserving lexical and compositional diversity.

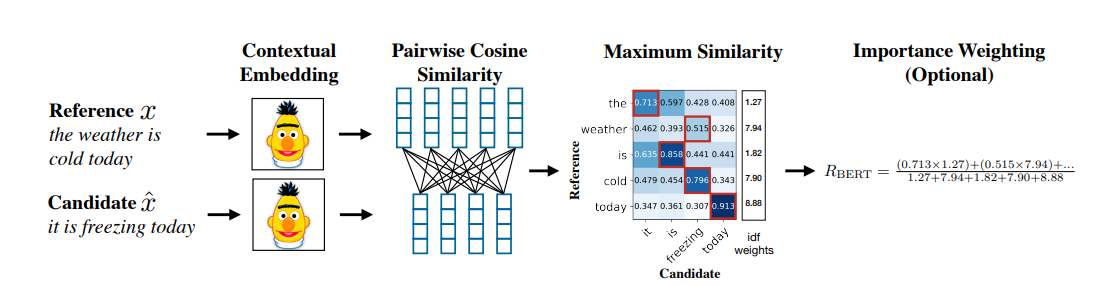
BERT Score is a language generation evaluation metric based on pretrained BERT contextual embeddings. Figure 1 illustrates the computation.

**Token Representation**

First, it uses contextual embeddings to represent the tokens in the input sentences: reference sentences and candidate sentences . The embeddings of both reference and candidate sentences are calculated using BERT embedding model. Given a tokenized reference sentence , the embedding model generates a sequence of vectors . Similarly, the tokenized candidate = is mapped to .

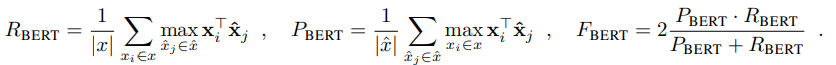
**Similarity Measure**

Then, we calculate the cosine similarity of the reference token and candidate token , which is . By using pre-normalized vectors, this calculation can be reduced to the inner product .

 Figure 1: Illustration of the computation of the BERT Score [3]

**BERT Score**

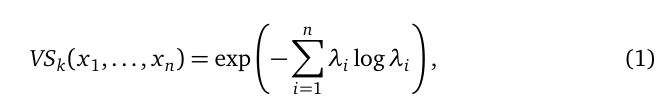
The complete score matches each token in to a token in to compute recall, and each token in to a token in to compute precision. It also uses greedy matching to maximize the matching similarity score, where each token is matched to the most similar token in the other sentence. Finally, a F1 measure is computed by combining precision and recall. For a reference and candidate , the recall, precision, and F1 scores are:



After the optional importance weighting, the is typically between 0 and 1, as well as the . Then, the final BERT Score well be a value between 0 and 1 too.

**2.2.3 Vendi Score [4]**

Vendi score is an automatic metric for evaluating diversity in machine learning. The input to metric is a collection of samples and a pairwise similarity function, and the output is a number, which can be interpreted as the effective number of unique elements in the sample. The Vendi Score is equal to the exponential of the von Neumann entropy of 𝐾/𝑛:



It can also be seen as the Shannon entropy of the eigenvalues, known as the effective rank [5], introduced by O. Roy and M. Vetterli in 2007.

Vendi Score achieve its maximum value when all samples are dissimilar and its minimum value when all samples are the same. Unlike BLEU and BERT score that require both reference text and candidate text, Vendi Score only relies on the samples to be evaluated for diversity. Thus, Vendi Score is suitable for evaluating the diversity of generative LLMs. In the following experiments, some indicators in Vendi Score will be used to help studying the diversity in the research.

**Main work**

1. **Diversity of LLM under default parameters**
   1. **Generation of Datasets**

Among lots of LLMs on markets, we chose some of the most well-known and widely used models from different companies. Different versions of a same model were included. Here is a list of the models:

1. GPT-3.5-Turbo, created by OpenAI, launched in 2022 [6]
2. GPT-4o, created by OpenAI, launched in May, 2024 [7]
3. Gemini-pro, created by Google DeepMind, launched in December, 2023 [8]
4. Gemini-1.5-pro, created by Google DeepMind, launched in February, 2024 [9]
5. Claude-instant-1.2, created by Anthropic, launched in August, 2023 [10]
6. Claude-3.5-sonnet, created by Anthropic, launched in June, 2024 [11]

**3.1.1 API Calls and Text Generation**

In creation of the datasets, the first step is to make API calls and get texts generated by different models. To ensure the randomness of the dataset, for each model listed above, it was asked to generate 700 random paragraphs. The texts were then extracted from the returned json file and be stored in a txt file for further usage.

For each model, we pose a query that is "Please generate a random one-paragraph essay, using characters in UTF-8 only". There is no specification to the length of each paragraph, but the maximum output token is set as 200 (around 150 to 200 words) to ensure standardization of the datasets. In this step, we keep all the parameters, including those parameters that may affect diversity, as default.

* 1. **Embedding**

Embedding refers to the process of mapping discrete objects (texts in this study) to a continuous vector space in the fields of natural language processing. This mapping can capture the inherent relationships and semantic associations between the objects. The main purpose of embedding is to transform discrete, high-dimensional texts into low-dimensional, continuous vector representations, to enable better processing and utilization of this data in machine learning models. In the vector space, similar words are represented by similar vectors which are closely placed. In that way, these vector representations can preserve the semantic information and patterns within the original data, allowing better processing of the data.

We used the pre-trained text-embedding-ada-002 embedding model provided by OpenAI. It takes the texts as input, and outputs a 1536-dimentional vector. The vectors returned from the model were stored with the corresponding texts for further usage.

* 1. **Clustering**

Although the texts are randomly generated, some of them may have common themes due to the feature of different models. In a machine learning system, grouping examples is often considered as a first step to understand a data set. This process of grouping unlabeled examples is called clustering [11]. From the many clustering method, we chose K-Mean clustering due to simplicity and efficiency.

K-means clustering is a popular unsupervised machine learning algorithm used for partitioning a dataset into K distinct clusters. The goal of the algorithm is to minimize the sum of the squared distances between data points and their assigned cluster centroids.

However, it is hard for human to perceive the distance of the embeddings, even they are in the same cluster, as the embeddings are vectors in high dimension. Thus, it is essential to visualize the result. Here, we used t-distributed stochastic neighbor embedding (t-SNE), a method visualizing high-dimensional data by giving each datapoint a location in a two-dimensional map [12].

Through experimentation, it is discovered that for each model, around 8 clusters are formed, and k is set as 8. The clustering results are shown in figure 2, figure 3 and figure 4.

For each pair of models, the embeddings from the earlier versions of the models (GPT-3.5-turbo, Gemini-pro and Claude-instant-1.2) has a more dispersed distribution than the later versions of the models (GPT-4o, Gemini-1.5-pro and Claude-3.5-sonnet). It indicates that at default parameters, for each LLM above, earlier versions have higher diversity than the later versions.

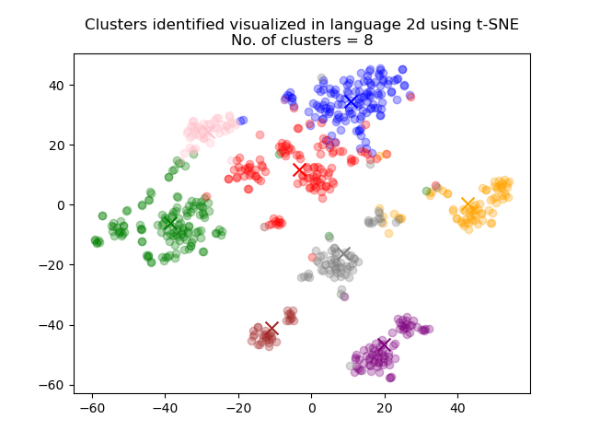
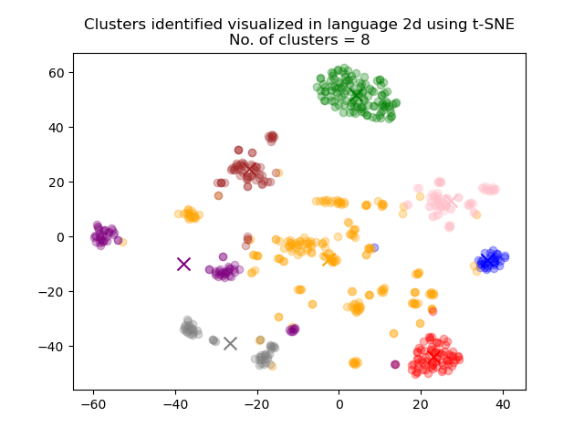


Figure 2: Clustering results for GPT-3.5-turbo (left) and GPT-4o (right). Embeddings from GPT-3.5-turbo has a more dispersed distribution.

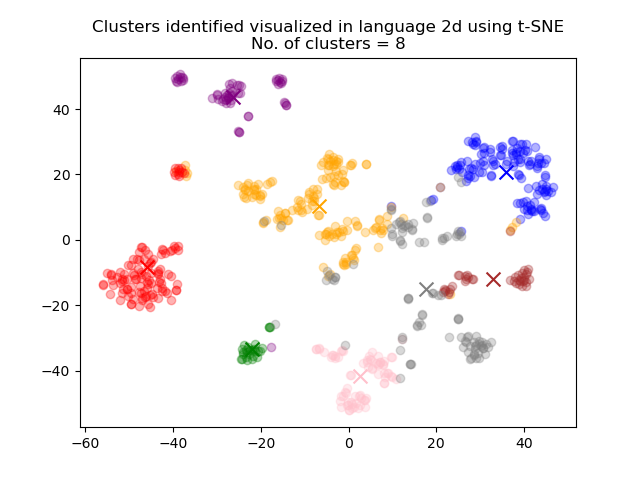
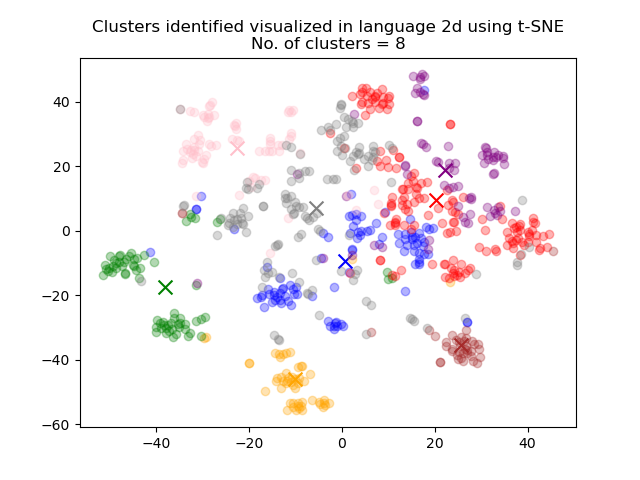


Figure 3: Clustering results for Gemini-pro (left) and Gemini-1.5-pro (right). Embeddings from Gemini-pro has a more dispersed distribution.

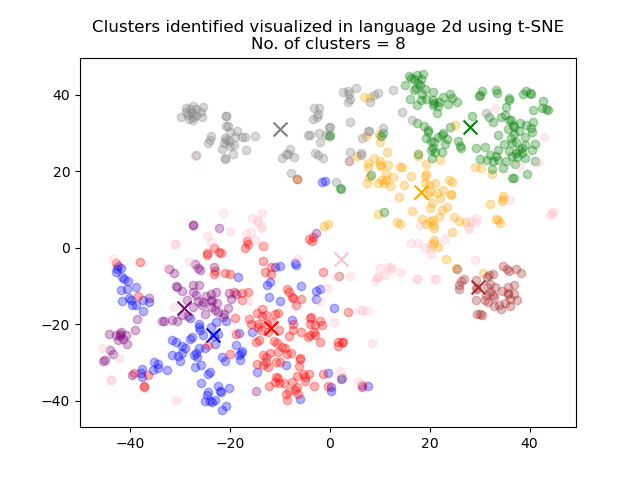
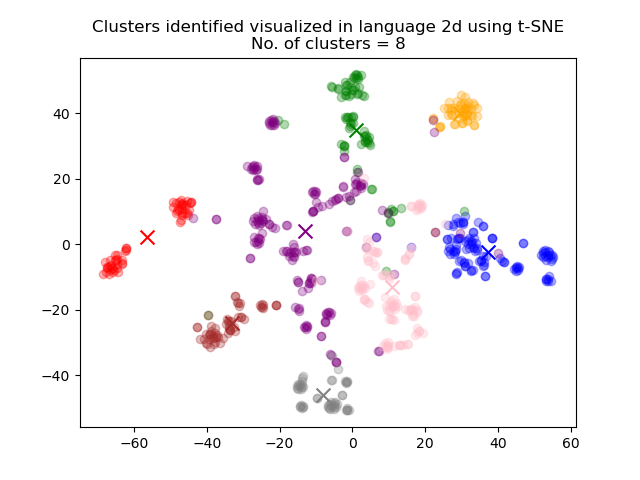
 

Figure 4: Clustering results for Claude-instant-1.2 (left) and Claude-3.5-sonnet (right). Embeddings from Claude-instant-1.2 has a more dispersed distribution.

* 1. **Theme difference in LLM**

It is also discovered that there exist some highly compressed clusters, and we are interested in studying the meaning behind these clusters. 5 samples of paragraph from each cluster are randomly selected to be checked. Then, they are sent to the GPT-4o model to let GPT conclude the theme of this cluster. Take the green cluster in GPT-3.5-turbo which is significantly away from other clusters in figure 2 as an example. From the explanation and sample paragraphs, it is clear that they all have the same theme “the concept of time”.

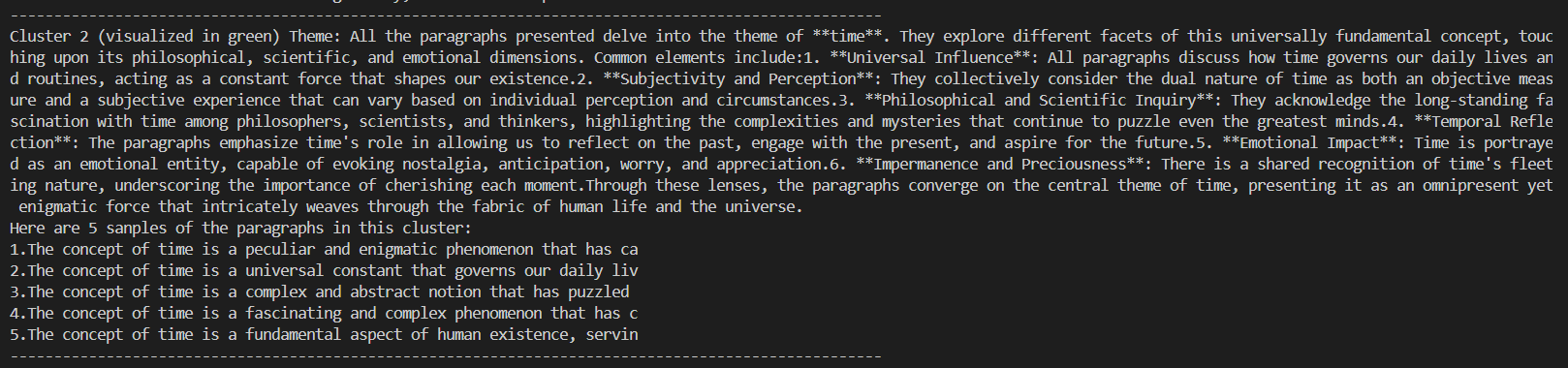


Figure 5: Details of cluster colored in green in figure 2.

Using this method, we can conclude the theme of texts generated by each model, and we are interested in finding the difference of theme between different version of the LLM. Here, we take the themes of texts from GPT-3.5-turbo and GPT-4o as an example. The theme of texts from GPT-3.5-turbo are:

(1) The importance of education

(2) Importance of cultivating positive traits

(3) Subjectivity and multifaceted nature of beauty

(4) The concept of time

(5) The complex and multifaceted nature of fundamental human concepts

(6) The impact of technology

(7) The importance of prioritizing self-care and well-being

(8) Luck, free will, and randomness

The theme of texts from GPT-4o are:

(1) Describe the beauty of the natural environment

(2) The interplay between technological advancements and human connections

(3) The serene and timeless charm of small-town life

(4) The beauty and tranquility of nature

(5) Finding tranquility and respite in small urban parks nestled within bustling cityscapes

(6) Finding solace, inspiration, and a sense of timelessness in serene, often quaint environments amidst the hustle and bustle of modern life

(7) The depiction of urban life and its inherent contrasts and connections

(8) Autumn and its associated sensory experiences, emotions, and natural beauty

GPT-3.5-Turbo's themes are more focused on abstract concepts, such as education, human traits, the subjectivity of beauty, the complexity of human nature, and the impact of technology. On the contrary, GPT-4o's themes tend to be more focused on describing actual scenarios and environments, such as natural environments, small-town life, urban parks, and the relationship between humans and nature.

Through the above study, we can conclude that GPT-3.5-Turbo is more focused on the realm of ideas and concepts, and GPT-4o is more skilled at vivid and concrete scene description and the interaction between humans and their environment. This difference can due to the differences in their training data and design objectives.

* 1. **Automatic Metrics Evaluation**

In addition to the above evaluation, we hope to give a quantitative assessment of the diversity of LLMs. Based on the preliminary works, four automatic metrics are selected:

**N-gram Vendi Score**

Based on n-gram language model, it is designed to evaluate the diversity of a set of sentences by calculating a Vendi Score [4]. A word n-gram language model is a purely statistical model of language, which is based on an assumption that the probability of the next word in a sequence depends only on a fixed size window of previous words. If only one previous word was considered, it was called a bigram model (n = 2); if two words, a trigram model (n = 3). The pseudo-code for calculating N-gram Vendi Score is shown below:

**Algorithm 1** N-gram Vendi Score

def ngram\_vendi\_score(sents, ns, tokenizer):

Ks = []

For n in ns:

X = normalize(get\_ngrams(sents, n, tokenizer))

similarity\_matrix = X \* X.T

Ks.append(similarity\_matrix.to\_array())

K = average(Ks)

return vendi.score\_K(K)

In the algorithm above, ns is a list specifying the n-gram sizes to consider, that is [1, 2, 3, 4] for unigrams to four-grams. Sents is a list of sentences to evaluate. In the for loop, the function iterates over the specified n-gram sizes (ns). For each n-gram size, it computes the n-grams from the sentences using the get\_ngrams function. Then these n-grams are normalized and used to create a similarity matrix Ks. The matrices for different n-gram sizes are averaged to form a final matrix K. Finally, The function returns a Vendi Score calculated from the averaged similarity matrix K using vendi.score\_K.

**BERT Vendi Score & SimCSE Vendi Score**

These two metrics are Vendi Scores based on embedding vectors [4]. The pseudo-code for calculating embedding Vendi Score is shown below:

**Algorithm 3** Embedding Vendi Score

def embedding\_vendi\_score(sents, model, tokenizer):

X = get\_embeddings(sents, model, tokenizer)

n, d = X.shape

if n < d:

return vendi.score\_X(X)

else:

return vendi.score\_dual(X)

In the algorithm above, sents is the list of input sentences, and model refers to the pre-trained embedding model. It first computes the embedding vector X of the sentences using the get\_embeddings function. Then, it gets the shape of X, that is the number of sentences n and the dimension of embedding d. If the number of sentences is less than the dimension of embedding, it calculates the Vendi Score by calling vendi.score\_X(X) directly. Otherwise, it calculate the Vendi Score using the matrix decomposition method by calling vendi.score\_dual. If the model used is BERT and Simple Contrastive Sentence Embedding (SimCSE) [13], we can get the BERT Vendi Score and SimCSE Vendi Score.

**Self-BLEU**

Self-BLEU is developed from the BLEU method. Unlike BLEU that require reference texts and candidate texts, self-BLEU measures the similarity of each sentence to the other sentences in the same set. It is also a metric used to evaluate the diversity of the generated sentences from LLMs [1]. The pseudo-code for calculating self-BLEU is shown below:

**Algorithm 3** self-BLEU

def calculate\_selfBleu(sents):

bleu\_socre = []

for s in sents:

create a copy of sents called sents\_copy

remove s from sents\_copy

bleu = get\_bleu\_score(s, sents\_copy)

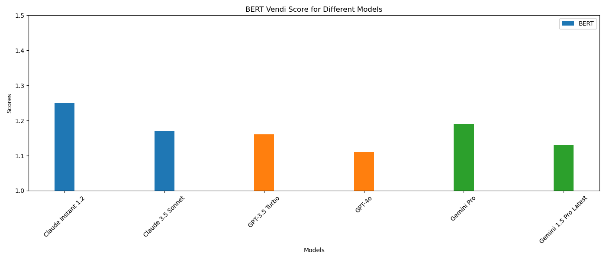
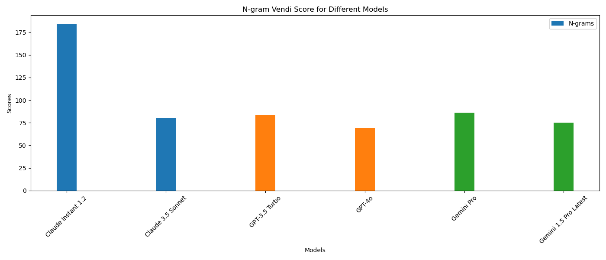
bleu\_score.append(bleu)

return average(bleu\_score)

In the algorithm above, sents is the list of the input sentences. It first creates an empty list bleu\_socre to store the BLEU scores. Then, it loops through each sentence in sents list. In the loop, it creates a copy of sentences list and then remove the current sentence from the copy. BLEU score is calculated using the current sentence as candidate and the copy containing sentences other than current sentence as reference. At the end of the loop, BLEU score calculated is appended to the bleu\_score list. After finishing the loop, it calculates the average of bleu\_score list as the final returned value, the self-BLEU score.

Noted that self-BLEU is inverse proportional to the diversity, a higher self-BLEU score means lower diversity of a model output.

The results of applying these four metrics to the generated texts of the selected LLMs are shown in figure 6 below.



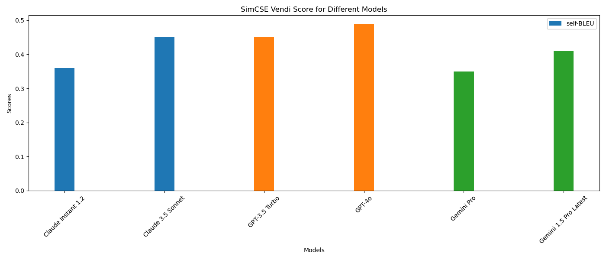
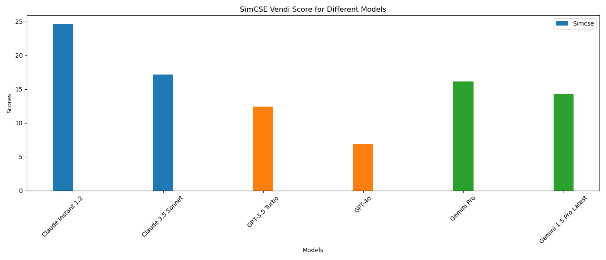


Figure 6: N-gram Vendi Score (top left), BERT Vendi Score (top right), SimCSE Vendi Score (bottom left) and self-BLEU Score (bottom right) of generated texts from LLMs: Claude Instant 1.2, Claude 3.5 Sonnet, GPT-3.5 Turbo, GPT-4o, Gemini Pro and Gemini 1.5 Pro (from left to right). For each model, 700 paragraphs generated texts are evaluated.

It is found that for each model, the earlier version has higher N-gram, BERT and SimCSE score and lower self-BLEU score, indicating that the earlier version has higher diversity than the later version. It also shows that under default parameters, Claude Instant 1.2 has the highest diversity, and GPT-4o has the lowest diversity.

When comparing different models, Claude has the highest diversity, and GPT has the lowest diversity overall. This phenomenon is especially evident when comparing SimCSE Vendi Scores, as the SimCSE embedding model published in 2021 has higher performance than the N-gram model and BERT embedding model [13].

For users who want to use website based LLMs (as website based LLMs are using default parameters) to do tasks that require low diversity such as translation, they are recommended to use GPT-4o as it has the lowest diversity. For users who want to use LLMs to help brainstorming, they are recommended to use Claude Instant 1.2 for its high diversity.

1. **Diversity of LLM Under Different Temperature Parameters**
   1. **Temperature**

In LLM, there are two parameters that may affect the diversity: temperature and top\_p, and here we are interested in how much temperature change can affect diversity. The LLM temperature is a key parameter that affects the balance between predictability and creativity of generated text. Lower temperatures prioritize exploiting learned patterns and produce more deterministic outputs, while higher temperatures encourage exploration and promote diversity and innovation [14].

In different versions of GPT and Gemini, the two widely used LLM, temperature is a parameter with range [0, 2] (except for Gemini-pro, which limits temperature to [0,1]). According to OpenAI, higher values of temperature like 0.8 will make the output more random, while lower values like 0.2 will make it more focused and deterministic. Values larger than 1 is considered really high temperature and are not recommended [15].

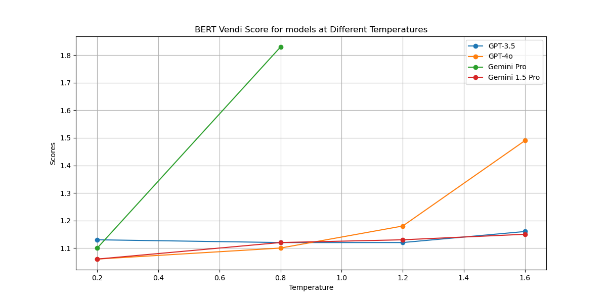
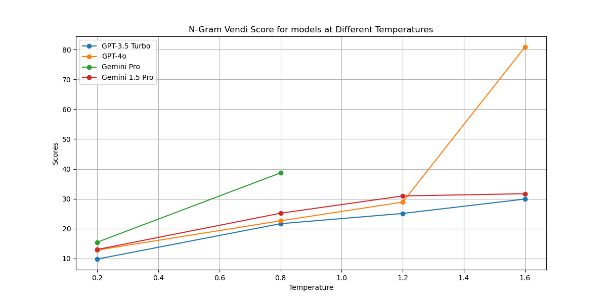
* 1. **Temperature and Diversity**

**4.2.1 Generation of Datasets**

Temperature cannot be change when using the website based GPT and Gemini, but is modifiable when using API calls. We used API calls to generate texts with different temperatures. 100 paragraphs of texts were generated for each model when temperature is 0.2, 0.8, 1.2, 1.5 respectively for evaluation.

**4.2.2 Automatic Metrics Evaluation**

Similar to the work in 3.5, we used N-gram Vendi Score, BERT Vendi Score and self-BLEU as metrics to evaluate the diversity in this part. The results are shown in figure 7 below.



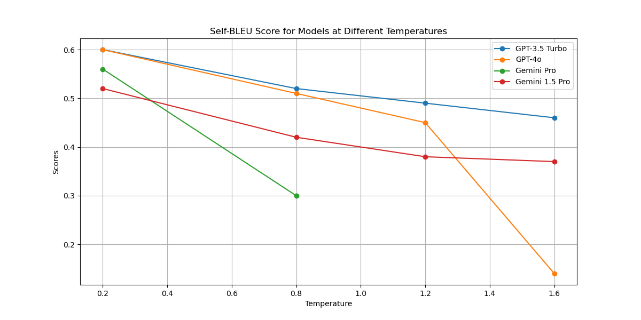


Figure 7: N-gram Vendi Score (top left), BERT Vendi Score (top left) and self-BLEU Score of generated texts from LLMs: GPT-3.5 Turbo (blue), GPT-4o (yellow), Gemini Pro (green) and Gemini 1.5 Pro (red) at different temperatures. For each model, 100 paragraphs generated texts are evaluated at each temperature.

It is found that for all models, N-gram Vendi Score and Bert Vendi Score increase when temperature increase and self-BLEU Score decrease when temperature increase, proving that diversity increase with temperature. For GPT models, at low and medium temperature, GPT-3.5-turbo and GPT-4o has similar value and gentle growth rate which is also similar. However, for temperature larger than 1.2, while GPT-3.5-turbo continues its gentle growth, GPT-4o’s value has a significant increase. For Gemini models, although Gemini-pro does not support temperature greater than 1, it shows a high diversity at 1 already. And Gemini-1.5-pro shows a slow but steady increase throughout, with data and growth rate similar to gpt-3.5-turbo.

For users who want to balance tasks that require low diversity, such as problem solving, and tasks that require high diversity, such as creative writing, GPT-3.5-turbo and Gemini-1.5-Pro are good choices. There is a nearly linear relationship between the diversity and temperature of these two models, allowing users to easily change the diversity of the model by adjusting the temperature to values they want. In addition, we do not recommend using GPT-4o with temperature greater than 1.2 as the sharp increase in diversity makes the model's output very unpredictable.

**4.2.2 Text Readability and Quality**

Some additional findings related to text readability and text quality at different temperatures are listed below.

1. At low temperatures, the paragraphs are very repetitive. The paragraphs tend to have a consistent structure, only differ by several key words. Some sample texts from GPT-3.5-turbo at 0.2 temperature is illustrated below:

In the heart of a bustling city, where skyscrapers pierce the sky and the hum of life creates a constant symphony, …

In the heart of a bustling city, where skyscrapers pierce the sky and the hum of life never ceases, lies a hidden garden…

In the heart of a bustling city, where the cacophony of honking cars and chattering pedestrians creates a symphony of urban life, lies a hidden garden…

The overall structure of these paragraphs is very similar, with only a few key words changing the meaning.

1. For GPT-4o under high temperature (larger than 1.5), the output texts may have low readability and quality. Although some of the paragraphs keep their completeness, texts similar to below may be generated:



Figure 8: Meaningless sentences generated by GPT-4o at temperature equals to 1.5

Thus, we do not recommend any kind of use of GPT-4o with temperature larger than 1.5.

1. Under high temperature, paragraphs generated by GPT-3.5-turbo and Gemini-1.5-pro still remain high readability. However, However, there is one Chinese paragraph mixed in with all 99 English paragraphs for GPT-3.5-turbo.
2. **Future Direction**

Our research focus on random generation of texts from LLMs. However, this is done by continuous API calls to LLMs, which the randomness of texts may be affected by the contextual ability and continuous output capacity of different LLMs. We may try different datasets generation strategy to eliminate these terms. We may also further evaluate the diversity of LLMs under certain conditions, for example the LLM diversity when completing specific tasks.

**Acknowledgement**

I would like to acknowledge my supervisor, Prof. Farzan Farnia for his patient guidance and support over the research period. I have learnt a lot in his creativity in research.

‌

**References**

[1] “Automated metrics for evaluating the quality of text generation,” *Paperspace Blog*, Jul. 05, 2022. <https://blog.paperspace.com/automated-metrics-for-evaluating-generated-text/>

[2] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “BLEU: a Method for Automatic Evaluation of Machine Translation.” Accessed: Aug. 09, 2024. [Online]. Available: <https://aclanthology.org/P02-1040.pdf?ref=blog.paperspace.com>

[3] T. Zhang, V. Kishore, F. Wu, K. Weinberger, and Y. Artzi, “BERTSCORE: EVALUATING TEXT GENERATION WITH BERT.” Available: <https://arxiv.org/pdf/1904.09675>

[4] D. Friedman and A. Bousso Dieng, “The Vendi Score: A Diversity Evaluation Metric for Machine Learning,” 2023. Accessed: Aug. 09, 2024. [Online]. Available: <https://arxiv.org/pdf/2210.02410>

[5] O. Roy and M. Vetterli, "The effective rank: A measure of effective dimensionality," *2007 15th European Signal Processing Conference*, Poznan, Poland, 2007, pp. 606-610.

[6] OpenAI, “OpenAI API,” *platform.openai.com*, 2024.

<https://platform.openai.com/docs/models>

[7] OpenAI, “Hello GPT-4o” *openai.com*, 2024.

<https://openai.com/index/hello-gpt-4o/>

[8] “Gemini - Google DeepMind,” *deepmind.google*. <https://deepmind.google/technologies/gemini/>

[9] “Releasing Claude Instant 1.2,” *www.anthropic.com*. <https://www.anthropic.com/news/releasing-claude-instant-1-2> (accessed Jul. 05, 2024).

[10] “Introducing Claude 3.5 Sonnet,” www.anthropic.com. <https://www.anthropic.com/news/claude-3-5-sonnet>

[11] Google, “What is Clustering?  | Clustering in Machine Learning,” *Google Developers*, 2019. <https://developers.google.com/machine-learning/clustering/overview>

[12] Wikipedia Contributors, “t-distributed stochastic neighbor embedding,” *Wikipedia*, Jul. 04, 2019. <https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding>

[13] T. Gao, X. Yao, and D. Chen, “SimCSE: Simple Contrastive Learning of Sentence Embeddings.” Available: <https://arxiv.org/pdf/2104.08821>

[14] “What is LLM Temperature? - Hopsworks,” *www.hopsworks.ai*. <https://www.hopsworks.ai/dictionary/llm-temperature>

[15] “OpenAI API,” *platform.openai.com*. <https://platform.openai.com/docs/api-reference/chat>