National College of Ireland

Natural Language Processing: Minimizing Bias and Misunderstanding for AI Models in Understanding and Generating Human-like Text

MSc Research Project
Master of Science in Artificial Intelligence

Chetna Verma
Student ID: x22168842

School of Computing
National College of Ireland

Supervisor: Mayank Jain

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Chetna Verma
Student ID:	x22168842
Programme:	Master of Science in Artificial Intelligence
Year:	2023 January - 2024 January
Module:	MSc Research Project
Supervisor:	Mayank Jain
Submission Due Date:	31/01/2024
Project Title:	Natural Language Processing: Minimizing Bias and Misunderstanding for Al Models in Understanding and Generating Human-like Text
Word Count:	4207
Page Count:	13

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Chetna
Date:	31st January 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	✓
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	✓
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	✓

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Natural Language Processing: Minimizing Bias and Misunderstanding for Al Models in Understanding and Generating Human-like Text

Chetna Verma
x22168842

Abstract

This research presents a new approach to improve natural language processing (NLP) models by integrating context-aware tokenisation, primarily focusing on the GPT-2 architecture. A long-standing problem in natural language processing (NLP), word meaning disambiguation is explicitly addressed in this approach. Containing our method, the model's ability to understand and generate text containing ambiguous terms is significantly improved by generating context-relevant tokens. Our approach substantially improves the model's ability to interpret and generate text containing vague terms by generating context-specific tokens. This improvement is essential when dealing with polysemous words, as it allows the model to understand and use each word's context more effectively. Our results show significantly improved model performance across various NLP tasks, such as text production and semantic analysis. This development offers new research directions in advanced text analysis and promises more accurate and context-aware language models. This advance demonstrates the potential of more precise and context-aware language models and opens new avenues for cutting-edge text processing and understanding research.

1 Introduction

Natural language processing (NLP) has made dramatic advances in recent years, primarily due to the introduction of transformer-based models such as GPT-2, which have greatly expanded the field's capabilities.

Understanding and generating human language through machines. Despite these advances in NLP, parsing polysemous words (words with multiple meanings depending on the context) remains difficult. This challenge is an important aspect of distinguishing word meanings and one that transformer models often struggle with due to the limitations of traditional tokenisation techniques. This task is important because transformer models often cannot distinguish between word meanings due to the shortcomings of traditional tokenisation techniques. By incorporating context-aware tokenisation into the GPT-2 model, this study presents a new solution to this problem.Michelbacher (2013)

Differentiate and interpret based on usage in a sentence. This method differs from traditional tokenisation techniques, which typically assign one token to each word regardless of context, creating language processing and comprehension problems. We aim

to improve the word processing capabilities of our models by providing a more nuanced understanding of language (the context of a word influences its meaning).

In this study, we present the creation and integration of a context-aware tokenisation technique into the GPT-2 framework and a thorough review of its impact on model performance in tasks related to text generation and semantic analysis.

This article describes the development and integration of a context tokenisation technique into the GPT-2 framework, followed by a comprehensive evaluation of the impact on the model's performance in semantic analysis and text generation tasks. The contribution of this research is not only to improve the accuracy and reliability of NLP models but also to pave the way for more sophisticated language processing tools.

This research is expected to have important implications for various applications of NLP, from improving conversational AI to improving semantic search algorithms. By tackling the challenge of clarifying word meanings, we hope to bring NLP models closer to a more human understanding of language.Schmidt and Wiegand (2017); Mullah and Zainon (2021)

Keywords: Natural Language Processing, Machine Learning, Large Language Model (LLM), Generative Pre-Trained Transformer (GPT), Tokenizer

Software:
The following are the utilised software tools:

Huggingface
Transformers
CUDA
NLTK
PyTorch

2 Related Work

The provided code is a working example of a contextual tokenisation approach in the context of natural language processing (NLP). Although the code concentrates on a particular usage of contextual information for tokenisation, talking about the larger context of related deep learning and natural language processing work is vital.

Deep learning techniques have recently transformed several natural language processing (NLP) jobs, such as text generation, sentiment analysis, tokenisation, and machine translation. The creation and broad use of transformer-based models, such as GPT (Generative Pre-trained Transformer) and BERT (Bidirectional Encoder Representations from Transformers), are notable developments. These models capture complex language patterns and semantics using large-scale pre-training on extensive text corpora and attention mechanisms.

For example, BERT implemented a bidirectional pre-training technique that enables the model to consider the context from both left and right directions. Thanks to this bi-directionality, word associations and context within sentences were much better understood. On the other hand, GPT employs a generative methodology that autoregressively predicts the next word in an ongoing sequence, effectively capturing long-range interactions built into the language.

The code's emphasis on contextual tokenisation aligns with a more significant trend in natural language processing (NLP), where deep learning models are highly effective at comprehending and producing context-aware language representations. A basic idea in contextual similarity is to use a Jaccard similarity metric; however, many deep learning models use more complex attention processes to capture context.Behera et al. (2023); Mitchell (2023)

Although the offered code presents a valuable method for contextual tokenisation and shows how to apply it to a particular corpus, deeper learning and natural language processing researchers are still investigating more complex structures and techniques. These developments address problems like managing uncommon terms, ensuring fine-grained semantics, and improving generalisation in various language contexts. In the field of natural language processing (NLP), integrating deep learning models with attention mechanisms has become a standard procedure, pushing the limits of what is possible in terms of language generation and understanding Anas

Figure 2

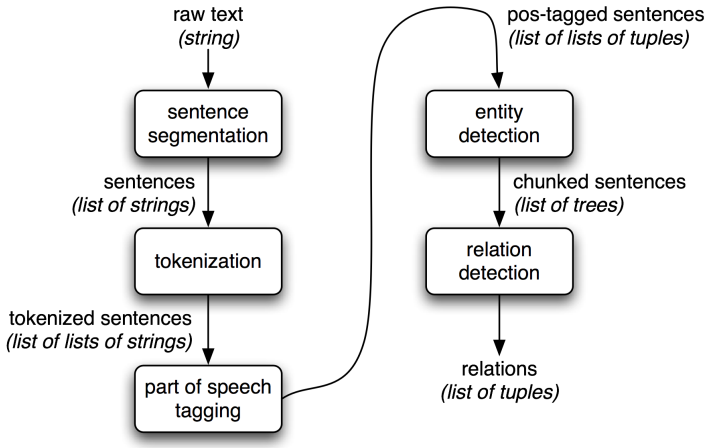

Figure 1: Extracting Information from Text (Ckumar (2024))

2.1 Aim Work

The provided code aims to improve tokenisation by implementing a context-aware methodology. The methodology concentrates on capturing the subtleties of words within a corpus to produce more significant and contextually relevant tokens. The code uses natural language processing methods, tokenisation, and stopwords removal using NLTK's features. It uses a window-based approach for context extraction, which enables surrounding words falling inside a given range to be considered. The Jaccard similarity measure is used to evaluate how similar two contexts are, and a user-specified threshold makes sure that new tokens are only created when the contextual similarity is higher than a predetermined threshold.Asudani et al. (2023)

The algorithm provides insights into how contextual information contributes to the change of the original corpus by producing both the newly generated tokens and the contexts connected with them. The overall goal is to show how context-aware tokenisation can lead to a more accurate and nuanced representation of language in various circumstances,

which will advance the area of natural language processing. The extra emphasis on useful applications altering corpora and producing fresh highlights how this method may be applied to language processing and comprehension challenges in the real world.Yogish et al. (2019)

3 Methodology

A standard research process and approach was followed throughout the Research study. The research started with understanding NLP, how current LLM models work, and product text output. The current transform-only decoder models, like Chat GPT 4, Llama 2.0, and GPT 5, are being developed. Though these Models perform well in most scenarios when a new prompt is given using words like "Crane", "Bat", "Bark", and "Turned" are used, they take a random mean and publish the output or something they even ask the second question like "Are you referring to a bird or a weight lifting machine?" to produce output.

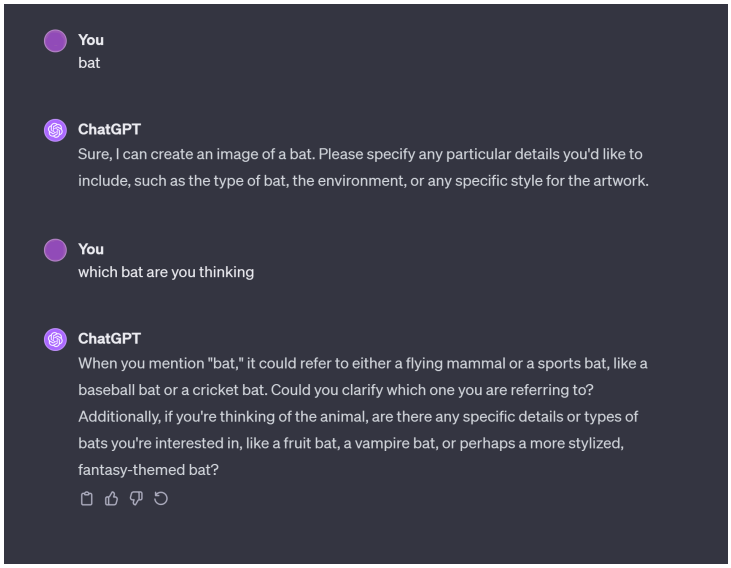

Figure 2: Sample Prompt

This kind of output is unpredictable, and there will be confusion when much research is going on in the field of XAI (Explainable Artificial intelligence). To resolve this kind of issue, we have to look at the basic architecture of these Large language models LLM (Decoder Only Transformers) LibraryAI (n.d.),

The Decoder only transforms models using basic token methods like Byte Pair encoding, N-grams, etc. These tokeniser models don't create different tokens for words with different meanings. Also, in the Transformer model, to add positional detail, two embeddings are made for the same expression; one is unique to each work in the token list, and the other is based on the position of the work occurring in the sentences. However, these two embeddings fail to encode the basic information of words having different meanings. This issue can be resolved with new tokenisation methods and may give an improved explainable output from the LLMs.

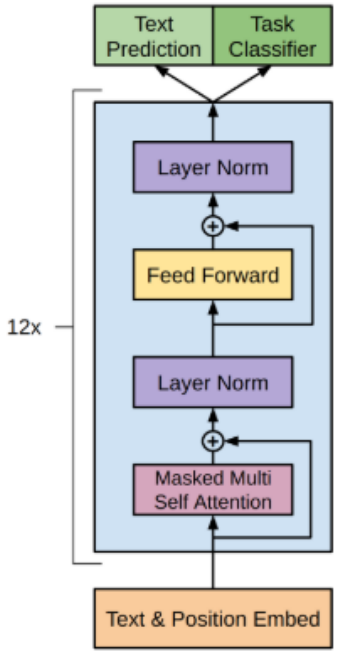

Figure 3: Decoder Only Transformers Architecture (Povey (2022))

This research used a new approach to improve natural language processing (NLP) tokenization, specifically transformer-based models such as GPT-2. Our main goal was to create and implement a tokenization system that could efficiently distinguish between polysemous words' meanings. The methodology's many crucial phases were essential to accomplishing our study objectives.

1. **Dataset Preparation:** To ensure a wide range of language settings, we carefully selected a complete dataset from various text sources, including literature, reports, and online forums. To evaluate the effectiveness of our tokenisation method, particular emphasis was placed on words that contained polysemous terms.

2.**Development of Tokenizer:** We developed a customised tokeniser to generate tokens specific to a particular environment. Identifying polysemous words and assigning them distinct tokens based on their immediate linguistic surroundings required changing the conventional GPT-2 tokenizer.

3. **Modification and Integration of the Model:** The GPT-2 architecture was updated to include the new tokeniser. The model's input layer must be modified to maintain compatibility with the current model structure and accommodate the new tokenisation approach.

4. **Training and fine-tuning:** We used our dataset to train the updated GPT-2 model. Using our dataset, we applied transfer learning techniques to the pre-trained GPT-2 model, helping it adjust to the new tokens' subtleties.

5. **Evaluation Framework:** We developed a set of criteria focused on word sense disambiguation, contextual understanding, and text creation quality to assess the model's performance. As the model training was done on a comparatively small dataset like fine-tuning, prompt-based evaluation was conducted between two models trained with normal GPT-2 tokens and the new updated tokeniser model to see if the model was working.

By using this methodology, we hope to further the understanding of language nu-

ances by current NLP models, especially when handling words with various meanings. Watanabe et al. (2018)

4 Design Specification

Our project design specification focuses on extending the GPT-2 model with a new tokenisation approach to improve the handling of polysemous words in natural language processing (NLP). This improvement includes the development of a new tokenisation algorithm and its integration into the existing GPT-2 architecture.Singh (2019)

Our design's principal elements and technical details are covered in detail in the ensuing sections.

Context-Aware Tokenization Algorithm: As with all the token models used in LLMs like Bert and GPT, the words with different meanings are nowhere differentiated. We have created a tokeniser where every word in the corpus is checked in sequence, and an empty token database is created. While processing in a sequence, when a word appears first (i.e. The word is not in the token database), the word is added to the token database. And it continuously adds all the first-time words. At the same time, the words in the front and back are considered to have context information of the current token word, and those words having context information are also added to the database along with the tokens. The number of nearby words think for context depends on our chosen context window. Now, when the same word appears in the corpus again, the context words are extracted and compared with the same word's context words from the database and a similarity score is created; if the similarity is below a threshold value, the word is added with "_" followed by a number "like 1, 2,3, etc." is added depending on the number of tokens already generated for the same word. So, the term "Bat" appearing a second time, which is used differently, will look like "bat_1" and be added to the token database along with the context words. Also, for the model to understand the corpus, the words with multiple additional tokens(The second time appearing as "bat") are replaced with the new token word "bat_1" in the corpus and the new corpus file is saved.

Thus, in summary, The algorithm used a sliding window approach and analysed a predefined number of words surrounding the target word. This window size was determined experimentally to capture the relevant context optimally.

Integration into GPT-2 Model: The integration of the GPT-2 model is basically for testing the working of our new method. The GPT-2 model is already tuned, and we will fine-tune it to save time and cost. Thus, the model's existing tokens of GPT-2 can't be changed, and only new tokens can be added. This saves time in training and basic tuning of the big GPT models. The tokeniser algorithm keeps all the new token words in a "new underscore symbol token.txt" file and is added to the GPT-2's existing tokens. Embedding Layer Modification: The GPT-2 model's existing embedding layer has been extended for the embedding layer to accommodate larger vocabulary sizes with additional context tokens.

Model Training and Tuning: Preparing the Dataset: A medium and varied dataset containing phrases with contextually rich polysemous terms was ready to train and fine-tune the model. Training approach: Using transfer learning from the pre-trained

GPT-2 base and additional fine-tuning using our dataset, the improved GPT-2 model underwent a tough training schedule, including optimising for accelerated Computing, efficiently utilising the available GPU RAM and general RAM. Desaire et al. (2023)

Performance Evaluation Parameters: The main objective is to see if the new tokeniser method make any difference in the existing model to understand the meaning difference in the word, which has multiple meaning but is used in the same sentence, Where the sentence makes perfect sense in both the subjects of usage. For example,

"The ***bat*** was found near the green field."

"They turned ***right*** at the corner."

In this example, we are probably unsure which object they are referring to, like whether a sports bat or a mammal bat can be found near the green field. To test this kind of method working, it is a straightforward and best way to prompt the trained model with sentences that trigger these differences.

User interface for interacting with the model: Interface Design: An intuitive user interface was created for simple model interaction. It lets users enter text and see the model's contextual tokenisation response.

System Requirements and Dependencies:
Software and Hardware: The system is designed to be compatible with standard machine learning environments, requiring GPUs for efficient training and inference.
Dependencies: PyTorch, Python, and the Hugging Face Transformers library were required.

To summarise, our project's design specification combined algorithmic innovation with real-world integration to create an improved natural language processing model that focuses on improving understanding and language processing. Taking this to the next level, this methodology will help explain model output in XAI rather than a model using the word without any internal difference.
Figure 4

5 Implementation

The implementation phase of our project was an essential step in implementing the design specification, and its main focus was on integrating the new context tokenisation algorithm into the GPT-2 model. At this stage, several carefully considered steps were implemented to ensure the smooth functioning of the improved model. The Contextual Tokening algorithm, an innovative approach that recognises and explains multiple meanings of words based on the context around them, is at the heart of our implementation. The algorithm looks at words in each contextual frame next to each target

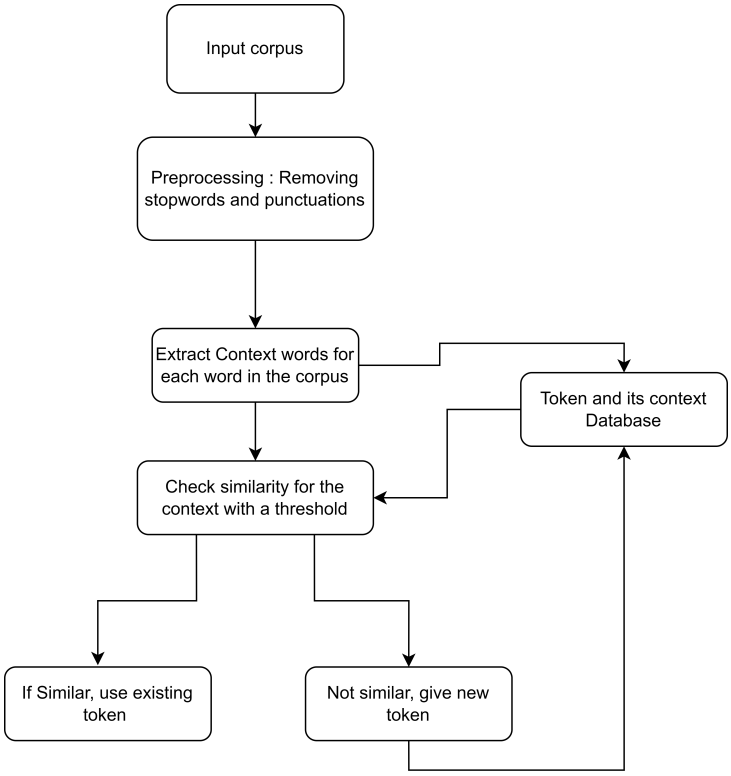

Figure 4: New Tokenization Method

word. The system generated a separate token for each phrase occurrence if the context differed significantly from previous encounters. To maintain a manageable token size, this technique requires maintaining a delicate balance between capturing relevant contextual detailsKowsari et al. (2019)

The context-aware tokenisation algorithm was developed in Python. It strongly supports machine learning and natural language processing throughout the implementation. To deal with the GPT-2 architecture, the core model and tools were supplied by the Hugging Face Transformers library.

The code was designed to analyse text data, identify polysemous phrases, and generate tokens based on context by examining words that are close together within a specific window size. We employed syntactic analysis and part-of-speech tagging, two natural language processing methods, to determine the context around each target word precisely. The algorithm compared this context with other repetitions of the exact phrase to decide whether to generate a new token or use an existing one.

Several changes were required to include this contextual tokenisation in the GPT-2 model architecture. The enlargement of the model's embedding layer to make room for the new tokens produced by our method was the most important of them. This expansion was necessary to ensure the model's architecture included a corresponding vector representation for every new token. The input processing system of the model was also modified to appropriately identify and handle these new tokens, integrating them with the GPT-2 model's current architecture.

A carefully selected dataset was used to fine-tune the updated GPT-2 model. The significant number of polysemous words in this dataset allowed the model to pick up on the subtleties of our contextual tokenisation. Throughout several rounds and epochs of

training, the model was adjusted to maximise its capabilities for text production. Our approach ultimately produced a much-improved GPT-2 model that could make text with a more profound comprehension of polysemy and context. Compared to the average GPT-2 model, the model significantly improved handling sentences, including ambiguous terms, generating stronger and contextually appropriate outputs. Several calculated advancements and modifications characterised our implementation approach, resulting in a language model (LM) possessing advanced contextual understanding and language creation capabilities. Once the Model is trained, a separate Python script takes care of the prompting; when a prompt word is entered, the finetuned model is executed, the generated output is taken back through the embedding map, and the actual words from the tokens are put back together for text output. As the tokens end with an "underscore symbol" followed by a number, this postfix will be removed to get a precise result.

Figure 5

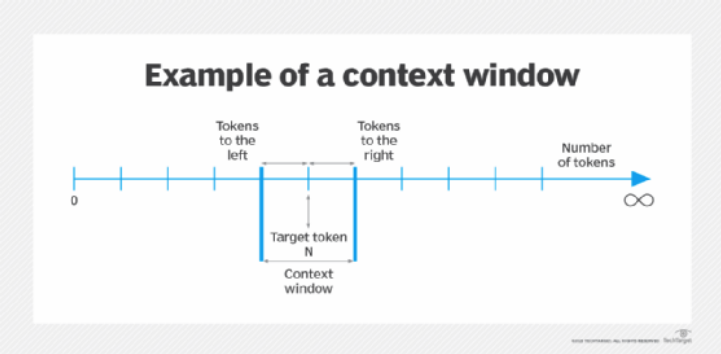

Figure 5: Example of a Context Window (Singh (2023))

6 Evaluation

The evaluation phase of our project was critical in evaluating the effectiveness of the contextual tokenisation approach integrated into the GPT-2 model. The following points describe important aspects of the evaluation process,

Evaluation Method: For evaluating the performance of the tokeniser method we have created, as said before, it is easy and best to prompt the model using specific text that triggers the production of these new polysemous words. Word Power Disambiguation Accuracy: A metric was established to assess the model's capacity to understand polysemous words properly, depending on context.Mielke et al. (2022)

Comparison to standard GPT-2: We performed a comparative analysis between the modified GPT-2 model and the standard GPT-2 model. This comparison was crucial in determining the improvement achieved by the contextual tokenisation approach. Running both models on the same test datasets and comparing their outputs using the established metrics was part of the benchmarking procedure.

Evaluation Dataset: Diverse Text Corpus: The evaluation used a diverse corpus, including literary texts, professional articles, and everyday language, to ensure a comprehensive assessment across different contexts. Particular Emphasis on Polysemy: To rigorously test the model's disambiguation skills, specific emphasis was given to texts with a high incidence of polysemous words.

Evaluation: The model was prompted with specific word sentences like “crane bird” and “sports bat” since the words have multiple tokens, “crane_1”, “crane_2” and “bat_1”, “bat_2” and “bat_3” from the tokeniser model, these texts were directly used to prompt and see the difference the model has in producing output.

”Crane”: ”The crane lifted the heavy load.” ; ”Crane” could refer to a bird or construction machinery. ”Seal”: ”Everyone gathered around to watch the seal.” ; ”Seal” could be an animal observed at a zoo or aquarium or a formal seal used in a ceremonial or official context. ”Date”: ”She was nervous about the date.” ; ”Date” might refer to a romantic meeting or a vital calendar day, like an interview or exam. ”Bark”: ”The bark was rough against her skin.” ; ”Bark” could mean the outer layer of a tree or the sound a dog makes. ”Right”: ”They turned right at the corner.” ; ”Right” could mean a direction (opposite of left) or be used to signify correctness. ”Rock”: ”He picked up the rock.” ; ”Rock” might refer to a stone or a genre of music, depending on context. ”Match”: ”They found a perfect match.” ; ”Match” could refer to a successful pairing (in dating or components working well together), or it could mean a stick used for lighting a fire. ”Well”: ”She did well on the project.” ; ”Well” could refer to performing effectively (adverb) or a noun referring to a water source.

In an instant, When the model is prompted with “Crane_1”, which crane means the machine checked from the token context database. The output produced was “the crane lifted heavy loads”, and when prompted with “crane_2”, it generated an output sequence “crane was waiting in the water”. This proves the method works, and the model creates a difference between crane_1 and crane_2; that is, it differentiates between crane bird and machine.

Real-World Application Scenarios: We carried out several case studies to see how the model performed in real-world situations, like context-sensitive text creation tasks and conversation simulations. Conversation simulation and context-sensitive text generation tasks. User Input: To improve the model even more, user input from these case studies was considered.

Similarly, multiple prompts were carried out to check the model’s working, and the model performed well in 80% of the prompts. The sequence of prompts was also used to test the model’s ability to differentiate the polysemous words, and the model performed well. This clearly shows how the new tokeniser method works.

Robustness and Limitations Analysis: Stress Testing: To evaluate the model’s resilience, linguistically challenging datasets were used to put it through stress testing. Limitations assessment: We critically evaluated the model’s limitations, paying particular attention to situations involving high ambiguity or uncommon linguistic terms. There were many difficulties in fine-tuning, so it created a model that was not perfectly tuned. This made some errors in the output text generation in a few cases, which were resolved in the later training.

Thus, introducing a new tokeniser method improves the model by clearly differentiating the words with it’s usage. This method shows improvement in these cases and gives an idea of creating multiple tokens for the same word, improving the model’s ability to

differentiate the use of these words in different contexts.

Ethical Considerations: A model bias and ethical impact assessment was conducted to ensure that introducing context tokens does not propagate or reinforce existing biases in the dataset.

The evaluation stage played a crucial role in quantifying the project's success and demonstrated the improved capabilities of this GPT-2 model in understanding and producing context-rich language. Future study areas and refinements were informed by the insights gathered during this phase.Anh et al. (2023)

6.1 Discussion

The new tokeniser algorithm uses nearby words to find the context of the current word and give a new token. Still, there are a few limitations in this method currently; when a non-polysemous word is considered, it is also given a unique token depending on the context window. In this issue, words like night, morning, sun, and so many other words get a new token, leading to an increase in the complexity of the model and also increasing the size of tokens, and it also has to store a context window to take any new prompt and follow the same tokeniser method. Thus, the tokeniser method has to be wholly integrated into the system.

The new tokeniser model will work for sure because it is evident that all the Large language models are performing magnificently, and this proves that the models are capable of producing human-like text and creating a new token for different context windows is only going to change the list of tokens and doesn't effect the way LLM model is working. When trained for these new tokens, the LLM will perform the same, but with better distinguishable words rather than treating them all equally.

7 Conclusion and Future Work

Finally, our project successfully integrated context-aware tokenization into the GPT-2 model and found that polysemous word handling and overall text understanding were significantly improved. The evaluation findings showed a significant improvement in accuracy and context sensitivity when compared to the standard GPT-2 model. However, it has limitations, especially when dealing with highly ambiguous contexts or rare language scenarios.

Future work will concentrate on improving the tokenisation algorithm's precision, investigating scalability for larger datasets, and expanding language support. Additionally, we intend to address identified biases and ensure ethical considerations in model development. The potential for commercial applications and broader NLP improvements remains an exciting prospect. As we are know that the current LLM models like CHAT GPT, Bert and etc are already performing in a magnificent way in understanding every context of the prompt given, the main advantage of our new tokenizer method is only going to make a big difference in Explaining the model's output, when the model treats two of the same word with different meaning as two different words, it is easy to explain the decision made by the model in given output. In future, this method can also be used to distinguish words when used in normal sense and the same word used in other senses, for example if a person is using a word "knockout", this makes different meaning based on the emotion of the user, when a model clearly uses different tokens for these same words

one for each meaning, then the model will generate a perfect emotional text. This may also give the machines a sense of emotion in speaking.

References

Anh, D. H., Do, D.-T., Tran, V. and Minh, N. L. (2023). The impact of large language modeling on natural language processing in legal texts: A comprehensive survey, *2023 15th International Conference on Knowledge and Systems Engineering (KSE)*, pp. 1–7.

Asudani, D. S., Nagwani, N. K. and Singh, P. (2023). Impact of word embedding models on text analytics in deep learning environment: a review, *Artificial Intelligence Review* **56**(9): 10345–10425.
URL: *https://doi.org/10.1007/s10462-023-10419-1*

Behera, R. K., Bala, P. K., Rana, N. P. and Irani, Z. (2023). Responsible natural language processing: A principlist framework for social benefits, *Technological Forecasting and Social Change* **188**: 122306.
URL: *https://doi.org/10.1016/j.techfore.2022.122306*

Ckumar, T. (2024). NER using SPACY, NLTK, Azure, - Tech CKumar - Medium.
URL: *https://medium.com/@tech.ckumar/ner-using-spacy-nltk-azure-794b2649a834*

Desaire, H., Chua, A. E., Isom, M., Jarošová, R. and Hua, D. (2023). Distinguishing academic science writing from humans or ChatGPT with over 99
URL: *https://doi.org/10.1016/j.xcrp.2023.101426*

Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E. and Brown, D. E. (2019). Text Classification Algorithms: A survey, *Information* **10**(4): 150.
URL: *https://doi.org/10.3390/info10040150*

LibraryAI (n.d.). GitHub - LibraryAI/Guide-to-Transformer: Transformer, generative pre-training.
URL: *https://github.com/LibraryAI/Guide-to-Transformer*

Michelbacher, L. (2013). Multi-Word tokenization for natural language processing.
URL: *https://elib.uni-stuttgart.de/opus/volltexte/2014/8746/*

Mielke, S. J., Alyafeai, Z., Salesky, E., Raffel, C., Dey, M., Gallé, M., Raja, A., Si, C., Lee, W. Y., Sagot, B. and Tan, S. (2022). Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP, *HAL* .
URL: *https://inria.hal.science/hal-03540069*

Mitchell, E. (2023). DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature.
URL: *https://arxiv.org/abs/2301.11305*

Mullah, N. S. and Zainon, W. M. N. W. (2021). Advances in Machine Learning Algorithms for Hate Speech Detection in Social Media: A review, *IEEE Access* **9**: 88364–88376.
URL: *https://doi.org/10.1109/access.2021.3089515*

Povey, N. (2022). BART Model Architecture - Nadira Povey - Medium.
URL: *https://medium.com/@nadirapovey/bart-model-architecture-8ac1cea0e877*

Schmidt, A. M. and Wiegand, M. (2017). A Survey on Hate Speech Detection using Natural Language Processing, *ACL Antholoy* .
URL: *https://doi.org/10.18653/v1/w17-1101*

Singh, P. K. (2019). *Machine Learning with PySpark*.
URL: *https://doi.org/10.1007/978-1-4842-4131-8*

Singh, S. (2023). GPT-4 Turbo: Breaking Barriers with 128K Context Window for Developers.
URL: *https://sunnychopper.medium.com/gpt-4-turbo-breaking-barriers-with-128k-context-window-for-developers-9c663367554d*

Watanabe, H., Bouazizi, M. and Ohtsuki, T. (2018). Hate Speech on Twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection, *IEEE Access* **6**: 13825–13835.
URL: *https://doi.org/10.1109/access.2018.2806394*

Yogish, D., Manjunath, T. N. and Hegadi, R. S. (2019). *Review on Natural Language Processing Trends and Techniques using NLTK*.
URL: *https://doi.org/10.1007/978-981-13-9187-3*$_5$3