SemEval-2024 Task 8: Multigenerator, Multidomain, and Multilingual Black-Box Machine-Generated Text Detection

Annanya Jain
University of Colorado Boulder
Boulder, Colorado
anja9719@colorado.edu

Devyani Srivastava
University of Colorado Boulder
Boulder, Colorado
desr8990@colorado.edu

Siva Naga Santosh Adabala
University of Colorado Boulder
Boulder, Colorado
siva.adabala@colorado.edu

ABSTRACT

This report delves into SemiEval Task 8, focusing on Subtask A: Binary Human-Written vs. Machine-Generated Text Classification. The task challenges participants to discern between human-authored and AI-generated texts, an increasingly crucial skill in the digital information age. We explore monolingual (English) tracks, reflecting the diverse linguistic landscape of today's global data. Our report begins with an overview of the task's significance, followed by a detailed examination of the technological evolution leading to the current state-of-the-art in both text generation and classification. We offer a comprehensive analysis of the methodologies adopted by various participants, highlighting the innovative approaches and techniques employed in this binary classification task. Our analysis extends to comparative assessments of performance across monolingual contexts, providing insights into the efficacy of different models and the unique challenges presented by linguistic diversity. Our analysis extends to comparative assessments of performance across monolingual contexts, providing insights into the efficacy of different models and the unique challenges presented by linguistic diversity. The report concludes with a look at future directions in the field, underscoring the growing significance of robust, linguistically diverse text classification systems in a world increasingly shaped by advanced AI text generation. Our findings are intended to guide future research and development in this vital area of computational linguistics, offering a beacon for navigating the complexities of human and machine-generated text classification.

1 INTRODUCTION

In the realm of computational linguistics and natural language processing, the SemEval-2024 Task 8 stands as a pivotal challenge, pushing the boundaries of automated text analysis. This task, titled "Multigenerator, Multidomain, and Multilingual Black-Box Machine-Generated Text Detection," addresses a critical and contemporary issue: the differentiation between human-authored and AI-generated text. The task is set against the backdrop of an ever-expanding digital universe where AI-driven text generation tools are becoming increasingly sophisticated, making the distinction between human and machine-generated content more challenging yet imperative. Subtask A of Task 8, which is our primary focus, presents a binary classification challenge where participants are required to determine whether a given text is human-written or machine-generated. This subtask is further divided into two tracks: monolingual, dealing exclusively with English texts, and multilingual, encompassing a broader linguistic scope. This distinction is crucial in understanding the complexity of the task, as it addresses the linguistic variability and the challenges it poses in text classification. But mostly out work is based on addressing the monolingual tracks. Our approach to Subtask A involves experimenting with various advanced nlp models and techniques. We delve into the utilization of state-of-the-art transformer-based models like DistilBERT, DeBERTa, RoBERTa, and ALBERT, each known for its unique strengths in processing and understanding natural language. The innovative aspect of our methodology lies in the hybrid approach, such as combining the DeBERTa tokenizer with the RoBERTa model, which is an exploration into how different components of these models can be synergized for enhanced performance. The rationale behind choosing these models stems from their proven efficiency in handling large-scale language data and their robustness in feature extraction and contextual understanding. Our objective is to harness their capabilities to accurately classify texts, navigating through the nuances and intricacies that differentiate human and machine-generated content. This endeavor is not just a technical challenge but also a venture into understanding the evolving landscape of AI-generated text and its implications. In this report, we will present a comprehensive analysis of our methodologies, the training and fine-tuning processes of these models, the evaluation metrics used, and the results obtained. Through this exploration, we aim to contribute valuable insights to the field of automated text classification and pave the way for future advancements in distinguishing between the nuanced writings of humans and machines.

2 BACKGROUND

The exploration and refinement of various models and tokenizers have become instrumental in advancing our understanding and capabilities in text processing. This report centers around the strategic application and nuanced tuning of a range of cutting-edge NLP models and tokenizers. These tools are fundamental in our quest to effectively discern and classify a diverse array of textual data, particularly distinguishing between human-authored content and text generated by artificial intelligence.

2.1 Overview of Tokenizers

At the forefront of our text analysis process are the tokenizers, each uniquely designed to complement specific NLP models. We employ the RoBERTa tokenizer, known for its byte-level Byte-Pair-Encoding capabilities, alongside the DeBERTa tokenizer which enhances positional understanding of words. ALBERT's tokenizer is leveraged for its efficiency, especially in large-scale data scenarios, while the DistilBERT tokenizer offers a balance between speed and comprehension. These tokenizers collectively lay the groundwork for effective text preprocessing, crucial for accurate model training and analysis.

2.2 Insights into Models

Our investigative focus extends to several transformative NLP models. The RoBERTa model stands out for its robust optimization, while DistilBERT offers a streamlined, efficient alternative to traditional models. DeBERTa distinguishes itself with a unique disentangled attention mechanism, and ALBERT is tailored for scalability and reduced computational demands. Each model has been meticulously fine-tuned and adapted to our specific dataset, ensuring optimal performance and accuracy in text classification.

2.3 Experimentation and Model Synergy

The harmonious integration of the DeBERTa tokenizer with the RoBERTa model, for instance, has shown exceptional promise. Our approach is not just limited to pairing but extends to fine-tuning various model parameters and architectures, reflecting a deep engagement with the intricacies of NLP modeling.

The synthesis of these advanced NLP tools and techniques forms the crux of our project. It represents a comprehensive effort to harness the potential of modern text analysis methods, aiming to deliver insightful, accurate, and efficient classification of text origins.

Throughout the experimentation phase, a thorough examination of tokenizer and model combinations was conducted, featuring notable pairings like BERT-based tokenization with RoBERTa models and DistillBERT tokenizer with corresponding models. Ultimately, the DeBERTa tokenizer paired with the RoBERTa model emerged as the most effective configuration. Further refinement through hyperparameter tuning was pursued, with specific adjustments to critical parameters. The resulting judiciously chosen model configuration, emphasizing synergy between the RoBERTa model and DeBERTa tokenizer, proved optimal for the task, demonstrating a nuanced understanding of the model's behavior and a commitment to efficacy based on comprehensive experimentation results.

3 METHODOLOGY

Our system for SemEval-2024 Task 8 employs a suite of advanced transformer-based models, each selected for its unique capabilities in language processing and text classification. The core models include:

3.1 Adapter Layer

In our efforts to enhance the performance of our model, we sought to incorporate adapter layers, specifically leveraging the PEFT technique alongside pre-trained models like BERT. The intention was to achieve task-specific fine-tuning without compromising the efficiency gained during pre-training, thanks to the minimal additional parameters introduced by adapters.

However, our attempts were met with a significant challenge—compatibility issues with the latest versions of transformer models. The evolving nature of transformer architectures posed a hurdle in seamlessly integrating adapters into our model. Despite the proven benefits of adapters in terms of parameter efficiency and comparable performance to traditional fine-tuning, the evolving transformer structures presented obstacles that hindered their straightforward incorporation.

The compatibility issues encountered necessitated careful consideration and potential modifications to adapt our existing adapter modules to the latest transformer versions. This challenge prompted us to explore alternative strategies for achieving task-specific adaptation while navigating the evolving landscape of transformer model architectures. While faced with this setback, our commitment to improving model performance remains steadfast, and we continue to explore other innovative solutions.

3.2 Tokenizers

3.2.1 RoBERTa. The RoBERTa tokenizer, derived from the GPT-2 tokenizer, is a byte-level Byte-Pair-Encoding tokenizer. Here we treat spaces as part of tokens, so a word is encoded differently at the start of a sentence (without space), also we replaced the multiple spaces with a single space and also removed the HTML tags as the text might be scraped from different sources .. The tokenizer includes special tokens like '<s+>' which is a pattern to match one or more whitespace characters, '<.*?>' a pattern to match HTML tags, and </d+> to match numbers. The tokenizer can handle sequences and sequence pairs, adding special tokens as needed.

3.2.2 DeBERTa. The DeBERTa tokenizer, developed specifically for the DeBERTa model, enhances BERT's word position understanding through its unique disentangled attention mechanism. This advanced attention mechanism allows DeBERTa to better capture the nuances in word relationships, improving the overall performance in complex text analysis tasks. The tokenizer includes special tokens like '<s+>' which is a pattern to match one or more whitespace characters, '<.*?>' a pattern to match HTML tags, and </d+> to match numbers.

3.2.3 ALBERT's. tokenizer, designed for the ALBERT model, emphasizes efficiency and scalability. It maintains the core functionalities of BERT's tokenizer but is optimized to reduce model size and increase processing speed, making it particularly suitable for handling large-scale datasets. The tokenizer includes special tokens like '<s+>' which is a pattern to match one or more whitespace characters, '<.*?>' a pattern to match HTML tags, and </d+> to match numbers.

3.2.4 DistilBERT. The DistilBERT tokenizer mirrors BERT's tokenizer with a focus on compactness and speed. While retaining the essential features of BERT, it is streamlined for faster performance, making it a preferred choice in scenarios where model efficiency is a priority without significant compromise in language understanding capabilities. The tokenizer includes special tokens like '<s+>' which is a pattern to match one or more whitespace characters, '<.*?>' a pattern to match HTML tags, and </d+> to match numbers.

3.3 Models

3.3.1 RoBERTa. (Robustly Optimized BERT Pretraining Accuracy) model (Liu et al., 2019) is a more robust version of BERT that is trained with a lot more data. It is based on fine-tuning the hyperparameters that has improved the results and performance of the model significantly. To boost the performance of BERT, RoBERTa also modified its training procedure and architecture. These modifications include removing the next sentence prediction and dynamically changing the masking pattern during pre-training.

3.3.2 DistilBERT. (Sanh et al., 2019) is a BERT-based Transformer model that is compact, quick, afford- able, and light. To shrink a BERT model by 40% during the pre-training stage, knowledge distillation is used. The authors offer a triple loss that combines language modeling, distillation, and cosine-distance losses to take advantage of the inductive biases that larger models acquire during pre-training.

3.3.3 DeBERTa. (Decoding-enhanced BERT with Disentangled Attention), introduced by He et al., innovates by disentangling the attention mechanism. This model enhances the BERT and RoBERTa models by incorporating a disentangled attention mechanism, where the relationship between words in a sentence is better understood, leading to improvements in natural language understanding tasks.

3.3.4 ALBERT. (A Lite BERT) is a model developed by Lan et al., focusing on reducing the size of BERT while maintaining its efficiency. It achieves this through parameter-reduction techniques, which help in scaling the model across larger datasets. ALBERT's design allows for increased training speed and improved performance on downstream tasks, making it a valuable model for processing large-scale language data.

3.4 Layers

The RoBERTa model has been used for the project in addition to its default setup, which can be found in https://huggingface.co. The initial RoBERTa configuration makes use of the GELU activation function and has a vocabulary size of 50,265; it also has a hidden size of 768, 12 hidden layers, and 12 attention heads. You broke from these conventions in your trials, concentrating on fine-tuning the model's structure to the particular task at hand. Changes included experimenting with the number of attention heads (10 to 14), changing the number of hidden layers (11 to 15), and changing the hidden sizes (500, 1000, 780, and 896). You also changed dropout probabilities and tested other activation functions, such as sigmoid, GELU, and ReLU. The purpose of these modifications was to improve the model's capacity to appropriately categorize texts, investigating how such variations might impact the model's performance in detecting human-written versus machine-generated text.

3.5 Optimizers

In our project, we utilized the Adafactor optimizer, an advanced and efficient alternative to traditional optimizers like Adam. Adafactor is particularly suitable for training large-scale models like RoBERTa due to its lower memory requirement and adaptive learning rate mechanisms. This optimizer dynamically adjusts learning rates based on the model's parameters, providing a more nuanced approach to training complex neural networks. The choice of Adafactor was strategic for our experiments, where we modified various configurations of the RoBERTa model. Its ability to scale parameter updates and efficiently handle large sets of parameters made it an ideal fit for our project's needs. This optimizer facilitated effective training and fine-tuning of the models, ensuring optimal performance

3.6 Learning Rate (scheduler)

For the project, the Adafactor learning rate (LR) scheduler was an integral component, working in tandem with the Adafactor optimizer. This scheduler is tailored for transformer models, providing efficient learning rate adjustments during training. It offers a balanced approach, enhancing stability and aiding in faster convergence, crucial for the complex architectures of models like RoBERTa. The LR scheduler's effectiveness lies in its dynamic adjustment of learning rates based on training progress, ensuring optimal model adaptation and performance. This feature was particularly beneficial in handling the varied configurations of RoBERTa.

3.7 Evaluation Metric

"F1" typically refers to the F1 score, which is a metric used to evaluate the performance of a classification model. The F1 score is a metric that takes into account both precision and recall. In the context of text classification tasks, precision measures the accuracy of the positive predictions, while recall measures the ability of the model to capture all the relevant instances of a particular class. In text classification, it's often important to balance between making accurate positive predictions (precision) and capturing as many true positives as possible (recall). F1 score provides a harmonic mean of precision and recall, offering a single metric that considers both aspects. Depending on the specific goals of the text classification task, there might be a need to prioritize either precision or recall. F1 score allows practitioners to find a balanced trade-off between these two metrics.F1 score condenses the evaluation of a classifier's performance into a single value, simplifying the comparison between different models or configurations. This makes it a convenient metric for decision-making in text classification scenarios.

4 EXPERIMENTAL SETUP

4.1 Dataset

The dataset we're working with is quite extensive, consisting of 119,757 rows, which suggests a rich and diverse range of textual data. The dataset is structured into five columns, likely encompassing the textual content for analysis, labels for supervised learning, and additional metadata that could include identifiers and source information. The presence of both human and machine-generated texts, as indicated by the labels, allows for comparative analysis and model training to distinguish between these two types of content. This balanced representation of human and AI-authored text is particularly beneficial for tasks like text classification, sentiment analysis, or authorship attribution, providing a comprehensive foundation for robust model training and evaluation.

4.2 Preprocessing

Preprocessing of text data plays a pivotal role in preparing it for machine learning models. This process involves several crucial steps:

4.2.1 Text Cleaning. The primary aim of text cleaning is to strip the raw data of all superfluous elements that do not contribute to or might even hinder the understanding of the text's core content. This includes the removal of HTML tags, which are common in texts

Table 1: Model Performance Metrics

Model	Tokenizer	Accuracy	Precision	Recall	F1-Score
bert-base-case	roberta	0.565	0.5729	0.565	0.5528
roberta-base	roberta-base	0.835	0.8577	0.835	0.8323
distillbert	distillbert	0.74	0.7637	0.74	0.7340
dberta	roberta	0.835	0.8537	0.835	0.8327
roberta-base-dberta	dberta	0.845	0.8817	0.845	0.8411
albert	albert	0.875	0.8836	0.875	0.8743

Table 2: Model Hyperparameters and Performance

Hidden Size	Hidden Layer	Num_attention_heads	Activation Layer	Train F1-Score	Test F1-Score
500	11	10	relu	0.5	0.333
500	15	10	relu	0.5	0.333
1000	15	10	relu	0.5	0.3333
780	15	12	gelu	0.5	0.35517
896	15	14	gelu	0.49	0.654246
896	15	14	sigmoid	0.5	0.71717

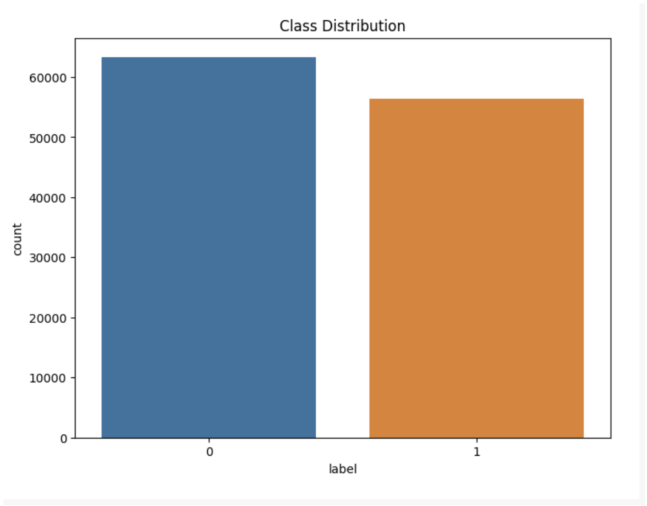

Figure 1: Distribution of Training Data

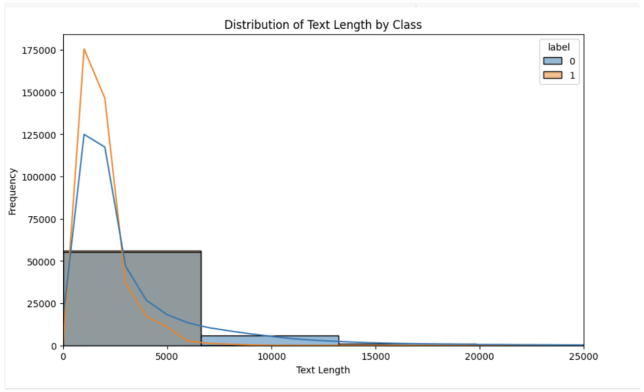

Figure 2: Distribution of Training Data

sourced from the web but are irrelevant for text analysis. HTML tags, if not removed, can introduce noise into the data, leading to inaccuracies in model training and prediction.

Another significant aspect of text cleaning involves dealing with numbers. While numbers can provide context in certain datasets, in many cases, they add unnecessary complexity without contributing meaningful information. The presence of numerical data can skew the analysis, as these elements often do not carry the same semantic weight as words in a sentence. Removing these elements helps streamline the dataset, ensuring that the machine learning models focus solely on the textual content that is most indicative of the text's origin - human or machine. This meticulous process of cleaning enhances the overall quality and reliability of the dataset, thereby improving the performance of text classification models.

4.2.2 Tokenization. This process is vital for transforming unstructured text into a structured form that machine learning models can understand and analyze. Tokens typically represent words or phrases, but they can also be individual characters or subwords, depending on the granularity required for the specific task.

The choice of tokenization method can significantly impact the performance of a model. For instance, in languages with complex morphology or in cases where the text includes a mix of languages, more sophisticated tokenization techniques might be necessary to accurately capture the linguistic nuances. Tokenization not only aids in simplifying the text for analysis but also helps in embedding layers where each token can be represented by a vector in a high-dimensional space, encapsulating its semantic properties. Effective tokenization is essential for ensuring that the subsequent steps in the text processing pipeline, such as embedding and model training, are based on a solid foundation of well-structured and meaningful data.

4.2.3 Text Normalization. This process includes transforming all characters to lowercase, which helps in reducing the complexity of the data by treating words like "Hello," "hello," and "HELLO" as the same. It also involves removing accents and special characters, making the text more consistent and easier for algorithms to process. Normalization aids in diminishing data sparsity and improves model performance by focusing on the essence of the language rather than its stylistic variations. By standardizing the text, models can more effectively learn patterns and make accurate predictions, particularly in tasks like classification, where consistency in data representation is key to distinguishing between different classes of text.

4.2.4 Padding and Truncation. Here in our project, we use Padding and Truncation. Padding involves adding a specific type of token to shorter text sequences to match the length of the longest sequence in a dataset. This ensures uniformity in input size, which is essential for models that require fixed-length inputs. Conversely, truncation is used to reduce the length of texts that exceed a predetermined maximum length. It involves cutting off parts of the text and focusing on retaining the most significant content. Both padding and truncation are crucial for efficient and effective model training, as they standardize input lengths, enabling the model to process batches of data efficiently.

4.2.5 Limiting Cleaning. Preprocessing is not necessary when utilizing pre-trained language representation models like BERT, as they leverage advanced mechanisms to capture information from sentences comprehensively. BERT employs a multi-head self-attention mechanism that considers all aspects of a sentence, including punctuation and stop-words, from diverse perspectives. The model's strength lies in its ability to learn contextual representations, where the meaning of a word can vary based on its context within a sentence.

The removal of stopwords and punctuation, often considered in traditional NLP preprocessing, may not enhance results when working with BERT. BERT's contextual learning encompasses stopwords and punctuation, which can significantly influence the meaning of a sentence. Removing these elements would inadvertently discard valuable context that BERT relies on to achieve superior results.

The tokenization process in BERT involves a word-piece vocabulary learned during training, making manual removal of tokens impractical. While identifying low-frequency tokens is feasible, BERT's vocabulary intentionally includes almost 1000 unused tokens, providing flexibility to remove specific tokens if desired. However, it's crucial to note that such removal may not yield discernible improvements, as BERT's design ensures efficient utilization of a broad vocabulary for comprehensive language understanding. In essence, the unique features of BERT, including its contextual learning and extensive vocabulary, render traditional preprocessing steps unnecessary for optimal model performance.

Similarly, other BERT-like models or transformer-based models designed for language understanding tasks, such as RoBERTa, ALBERT, and DistilBERT, share the fundamental concept of capturing contextual information bi-directionally.

4.3 Final Model

Our final model adopts a sophisticated approach to text classification, leveraging the latest advancements in natural language processing. We have chosen to implement the following models:

RoBERTa: A robustly optimized version of BERT, fine-tuned for our specific task. The RoBERTa model is known for its improved accuracy and efficiency. It's been trained with a batch size of 8 over a few epochs.

DeBERTa: This model enhances BERT and RoBERTa with a disentangled attention mechanism. For our project, DeBERTa is fine-tuned to better understand the intricate relationships in our dataset, employing a tailored training approach to optimize its performance.

We utilize the Hugging Face Transformers library, which offers pre-trained models and tokenizers for a variety of architectures. The primary model chosen for this task is 'roberta-base,' a widely used transformer-based model known for its robust performance in natural language understanding tasks. The training and validation datasets are prepared from the provided pandas DataFrames. These data frames are then converted to Hugging Face Datasets for compatibility with the library's training pipeline. We initialize the RoBERTa model and its corresponding tokenizer from Hugging Face's model hub. The chosen model architecture is 'microsoft/deberta-base,' which has demonstrated effectiveness in sequence classification tasks. The tokenizer is responsible for converting input text into a format suitable for model input. The training process is configured using the Hugging Face Trainer class. We specify the output directory for saving checkpoints, the learning rate, batch sizes for training and evaluation, the number of training epochs, weight decay, and evaluation and saving strategies. We employ the Adafactor optimizer, a variant of Adam, and a learning rate scheduler specific to Adafactor. The training and validation datasets are tokenized using the chosen tokenizer, and padding is applied to ensure uniform sequence lengths. The model is fine-tuned using the Trainer class, which handles the training loop, optimization, and evaluation. The best model is saved based on the evaluation results. For testing, the RoBERTa model and tokenizer are loaded from the saved checkpoints. The testing dataset is tokenized using the same preprocessing function used during training. Softmax activation is applied to obtain probability predictions, and the class with the highest probability is selected as the final prediction.

Due to computation limitation, we were able to run our code for 4 epochs only which gave us the F1 0.5, Validation Loss of 0.693307, and Training loss of 0.693200.

We then limited our training data to 20,000 data points, validation and test data points to 4000 and 2000 respectively. Which was run for 5 epochs to give a training loss of 0.69340, validation loss of 0.693982, and F1 of 0.5000. The precision, recall, and f1-score on testing data were 0.25,0.5 and 0.333.

5 CONCLUSION

In Conclusion, this experimentation process underscores the importance of a systematic and iterative approach to model development. The initial exploration of diverse tokenizer and model combinations not only demonstrated the versatility of transformer architectures but also revealed the nuanced dynamics between different components. The meticulous scrutiny of hyperparameters

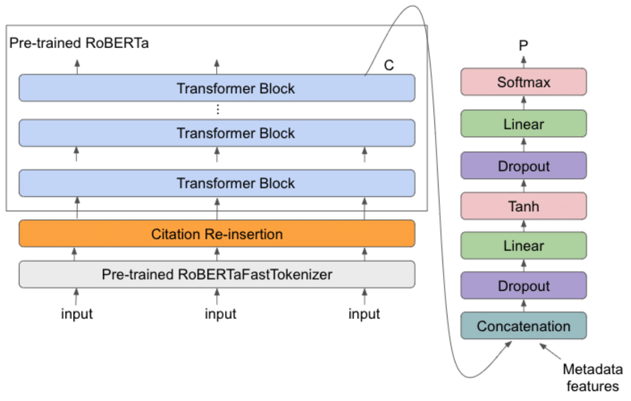

Figure 3: Generic Architecture of Roberta Model without Deberta Tokeniser

further elucidated the model's sensitivity to specific configurations. While hyperparameter tuning is a powerful tool for refining models, the acknowledgment that the initial configuration, featuring the RoBERTa model and DeBERTa tokenizer, already represented an optimal solution is significant. This decision showcases a deep understanding of the interplay between architecture and task requirements, emphasizing the importance of domain-specific knowledge in guiding model development. In essence, this study exemplifies a comprehensive and thoughtful strategy for navigating the complexities of model selection, experimentation, and fine-tuning within the realm of natural language processing.

6 GITLINK

https://github.com/devyanisri/Semeval2024Task8a

ACKNOWLEDGMENTS

We express our heartfelt thanks to Prof. Dr. James Martin for his unwavering support and guidance, which significantly contributed to our successful implementation of learned knowledge in completing this task. His mentorship has been invaluable to our academic journey.

REFERENCES

[1] https://www.semanticscholar.org/reader/077f8329a7b6fa3b7c877a57b81eb6c18b5f87de
[2] https://aclanthology.org/2021.emnlp-main.20.pdf
[3] https://arxiv.org/pdf/1910.01108.pdf
[4] https://www.semanticscholar.org/reader/7a064df1aeada7e69e5173f7d4c8606f4470365b
[5] https://openreview.net/pdf?id=XPZIaotutsD
[6] https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf
[7] https://ar5iv.labs.arxiv.org/html/1804.04235
[8] https://arxiv.org/pdf/2306.16842.pdf
[9] https://www.semanticscholar.org/reader/0de580957d23dd65e31b6c95e6bc5d15bc15c57d
[10] https://www.researchgate.net/publication/370262546_Semantic_Tokenizer_for_Enhanced_Natural_Language_Processing
[11] https://aclanthology.org/C92-4173.pdf