Language Processing II

MSc in IT & Cognition, 2023

Multi-Author writing style change detection: Encoder versus Prompt Engineering

Lecturers: Manex Agirrezabal and Patrizia Paggio

Department of Nordic Studies and Linguistics. August 17, 2024

Group members

Exam number	Contributions
Shiling Deng(56)	Fine-tuning and evaluating (BERT); Introduction (BERT); Related work (BERT); Methodology (BERT); Reviewing report
Yuwei Shen(55)	Abstract; Introduction(Prompt); Related work(prompt); Baseline model; Question 1
Qing Li(22)	Methodology, Experiments, Results and Discussion (prompting); Conclusion; Question 2

1 Abstract

Detecting writing style changes in multi-author documents is a critical task in author identification, providing solutions for plagiarism detection, author verification, and writing support technologies. This paper presents our experiment for the Multi-Author Writing Style Analysis 2024 task. We introduce an innovative approach utilizing large language models (LLMs), specifically ChatGPT, through Feature-Prompt, which leverages step-by-step stylometric explanation prompts to enhance interpretability and comprehensive analysis. Additionally, we fine-tuned BERT-based models, which achieved optimal performance by pre-training deep bidirectional representations, for the author style change detection task. Furthermore, we employed a baseline model using a Random Forest classifier with combined feature representations, including TF-IDF, POS tags, character-level n-grams, and text statistical features, to capture different aspects of writing style. Our results highlight the effectiveness of these methods in detecting changes in writing style and authorship attribution.

Contents

2 Introduction

Detecting author switches in multi-author documents is a challenging and interesting problem in the field of author identification. This field, known as Stylometry, involves the statistical examination of differences in literary style across various authors or genres. According to the Oxford Dictionary, it is described as "the statistical analysis of variations in literary style between one writer or genre and another." This practice has a long history, dating back to the early Renaissance.(Iyer & Vosoughi, 2020a). In forensic analysis, it can help verify the authenticity of documents and uncover cases of ghostwriting(Iqbal, Khan, Fung, & Debbabi, 2010), which is crucial for legal investigations and the validation of important documents. In academic settings, it aids in detecting instances of plagiarism(Stamatatos & Koppel, 2011). Detecting writing style changes in multi-author documents is crucial for security applications. It can help identify authorship inconsistencies in sensitive documents(Shu, Wang, Lee, & Liu, 2020), detect potential impersonation or fraud in digital communications, and uncover deceptive content in online platforms. Such capabilities are essential for maintaining the integrity and authenticity of information in various security contexts.

Since its inception in 2009, PAN has hosted numerous shared tasks addressing various computational challenges in stylometry and digital text forensics. These tasks have ranged from determining whether a document is single- or multi-authored, to identifying the number of authors within a document, detecting style changes between paragraphs, and even pinpointing style changes at the sentence level. The PAN 2024 Multi-Author Writing Style Analysis task continues the tradition of exploring the detection of writing style changes in multi-author documents. The primary goal is to identify the exact positions within a given text where the author's writing style changes, specifically at the paragraph level. The datasets are based on user posts from various subreddits on the Reddit platform. Participants must determine whether there is a style change between consecutive paragraphs in these documents. The output is a binary array indicating whether a style change occurs (1) or does not occur (0) between each pair of paragraphs. This setup facilitates a precise evaluation of the model's ability to detect nuanced changes in writing style.

In this study, we first formulated the task of detecting authorship change among a few paragraphs into an authorship change detection between two consecutive paragraphs. We utilized two main methods to detect writing style changes in multi-author documents: prompt engineering with GPT-based models and fine-tuning BERT models. Additionally, we implemented a baseline approach using traditional feature extraction including TF-IDF, POS tags, character-level n-grams, and text statistical features with a Random Forest classifier. Our hypothesise is we can extract features from lexical and syntactic aspects to detect authorship changes, and methods using GPT, BERT, TF-IDF, and POS feature extraction are effective in detecting authorship changes in multi-author documents. Prompt engineering refers to the systematic design and optimization of input prompts to guide the responses of LLMs, ensuring accuracy, relevance, and coherence in the generated output (Chen, Zhang, Langrené, & Zhu, 2024). This process is crucial in harnessing the full potential of these models, making them more accessible and applicable across diverse domains. Carefully designed prompts can more effectively guide models to generate more accurate and relevant outputs, enhancing the controllability and interpretability of the models. Prompt engineering techniques have demonstrated exceptional application potential across multiple domains, including education, content creation, healthcare, and finance. Previous research has demonstrated the potential of prompt engineering in authorship verification (AV) Hung et al(Hung, Hu, Hu, & Lee, 2023) proposed PromptAV, an innovative approach that guides large language models (LLMs) in authorship attribution analysis by providing step-by-step prompts on stylometric features. Simultaneously, Huang et al(Huang, Chen, & Shu, 2024) introduced

Language Indication Prompts (LIP), aiming to further enhance LLMs' ability to identify core stylistic and linguistic features relevant to authorship analysis. These research findings demonstrate that, with the aid of prompt-based learning paradigms, LLMs can not only accurately perform authorship verification in an unsupervised manner but also overcome the limitations of traditional methods, offering interpretable insights for complex language understanding tasks.

In this study, we will explore how to apply prompt engineering to the task of multi-text classification. Our approach leverages advanced large language models (LLMs), particularly ChatGPT, to address the complexities of multi-author writing style analysis.

BERT is a language representation model developed by Google AI Language. Unlike previous models that focused on unidirectional context, BERT aims to pre-train deep bidirectional representations by joint conditioning on both left and right context in all layers. This approach enables BERT to achieve optimal performance across various natural language processing tasks. The architecture of BERT is based on the Transformer model, which uses self-attention mechanisms to process input text(Devlin, Chang, Lee, & Toutanova, 2018a). The pre-trained BERT model can be fine-tuned with just one additional output layer to adapt to a variety of downstream tasks, making it a versatile and powerful tool in NLP. For example, in text classification, question-answering systems, natural language inference, and machine translation(Koroteev, 2021). In Vries' work (De Vries, Van Cranenburgh, & Nissim, 2020), they demonstrated that BERT works like a language processing pipeline. In lower layers, word representations contain information about POS tagging and named entity recognition. In higher layers, the learned representations contribute more to abstract information, e.g. coreference resolution and dependency parsing. This suggests the representations from the last layer attend on constituents and long-distance correlations, which can be used to detect authorship change.

BERT plays an indispensable role in multi-author text analysis. In the multi-author text analysis competitions of PAN[1], many participants have leveraged BERT and its various variants for text embedding and feature extraction. In this paper, We utilized Google's pre-trained BERT model to tokenize and embed each sentence in the documents, and explored fine-tuning the BERT model. We employed the representation of token [CLS] at the last layer as the the embedding of to consecutive paragraphs. Then, we applied a random forest classifier on these feature vectors to classify whether there is a change of authorship. Our code and data can be found on our Github repository [2].

3 Related Work

Science detecting whether two documents are written by the same person is vital in morden online communication, plenty of research have been done on this topic. We will illustrate a few of them and clarify our motivation for exploring the three methods.

3.1 Prompt-Engineering

Prompt engineering is a novel and increasingly popular approach in the field of natural language processing (NLP). This technique involves designing and utilizing specific prompts to guide large language models (LLMs) such as GPT-3 or ChatGPT to generate the desired output. As large language models continue to develop, prompt engineering is becoming more and more important. This approach not only enhances model performance but also guides the models to generate more accurate and relevant outputs through carefully designed prompts.

[1] https://pan.webis.de/clef24/pan24-web/style-change-detection.html
[2] https://github.com/avialofmeth/Multi-Author-Style-Change

In the 2021-2022 overview of Multi-Author Writing Style Analysis task at PAN(Bevendorff et al., 2021, 2022), we found that most approaches employed traditional machine learning techniques, as well as deep learning and neural network methods, to detect style changes. Only one paper uses prompts to solve this question, they Combine the DeBERTa model with contrastive learning, using automatically generated prompts to identify style changes. The application of ChatGPT in multi-text classification analysis is rarely explored. Traditional methods have primarily used LLMs for auxiliary tasks, such as feature extraction and data annotation, instead of fully utilizing their capabilities(Patel, Rao, Kothary, McKeown, & Callison-Burch, 2023). Unlike traditional models, LLMs, i.e. ChatGPT, can solve problems without fine-tuning and provide understandable natural language explanations for its predictions(Barlas & Stamatatos, 2020). In the article *Can Large Language Models Identify Authorship?* the authors comprehensively evaluated the use of LLMs for authorship verification and attribution, exploring the ability of LLM models to distinguish changes in authorship(Huang et al., 2024). Our paper builds on the PromptAV approach(Hung et al., 2023) by focusing more on the impact of linguistic features on multi-author identification, introducing the Feature-Prompt. This method employs step-by-step prompts and feature analysis, enabling the model to accurately capture and understand the stylistic differences between different authors. Specifically, the Feature-Prompt method guides the model to focus on linguistic features such as punctuation, vocabulary choice, syntactic structure, and text sentiment, systematically analyzing each pair of text passages to effectively detect changes in authorship.

3.2 Transformer Encoders

In Iyer and and Vosoughi's paper (Iyer & Vosoughi, 2020b), they tackled the task of identifying stylistic changes in documents to determine multiple authorship. They employ BERT embeddings to generate sentence-level vectors and use a Random Forest classifier for the classification tasks. They applied Logistic Regression, Decision Trees, Random Forest, Support Vector Machines and Naive Bayes (Multinomial and Gaussian) on these tasks, and found that Random Forest is the best. The method demonstrated strong performance with F1 scores of 0.86 for detecting style changes.

However, the lack of fine-tuning BERT is a limitation given the differences between their tasks and BERT's pretraining tasks. Specifically, BERT's pretraining includes tasks like determining if two sentences are sequentially related, focusing on topic and semantic coherence, which are not directly related to writing style. Writing style can span different topics and articles by the same author, indicating a need for fine-tuning. Additionally, using representations from the last four layers and summing them is computationally intensive. Utilizing the feature of [CLS] from only the final layer could simplify the approach.

In the PAN 2022 competition, Lin, etc. proposed a system using an ensemble of fine-tuned pre-trained transformers, including BERT, RoBERTa, and ALBERT, to address the Style Change Detection tasks (Lin, Chen, Tzeng, & Lee, 2022). Their method utilizes the [CLS] token for classification and combines predictions from individual models through majority voting, achieving a strong performance. However, the use of three models during inference results in lower computational efficiency. The authors highlight the potential improvements by focusing more on feature extraction and using fewer model layers for inference to enhance efficiency. The project underscores the efficiency and effectiveness of pre-trained models in writing style change detection while suggesting areas for future optimization.

Similar to the previous one, Ahmad Hashemi and Wei Shi also fine-tuned BERT, RoBERTa, and ALBERT for the tasks. Additionally, they employed data augmentation techniques. Their approach includes creating non-consecutive paragraph pairs to enhance training data and using an ensemble modeling strategy for improved accuracy. This method ranked first in two sub-tasks and second in another (Hashemi & Shi, 2023).

Overall, the Transformer architecture has demonstrated significant effectiveness in detecting authorship changes. Previous studies have shown that using the fine-tuned BERT representation alone is effective, as well as combining BERT-extracted features with a random forest classifier. However, no one has yet combined the fine-tuned BERT [CLS] representation with a random forest classifier. We hypothesize that combining the fine-tuned BERT with a random forest classifier can further enhance accuracy. Additionally, while prior research primarily reports experimental results, it lacks discussions on BERT attention distribution and feature extraction. Therefore, we visualized the final layer attention weights of BERT and use word clouds to show the attention distribution on our test set.

4 Methodology

4.1 Zero-shot Prompting

Inspired by a recent work (Hung et al., 2023), we designed two different prompts - *Chain of thought* (CoT) Prompt (Wei et al., 2023) and Feature-Prompt - to test the zero-shot prompting performance of LLMs on detecting multi-author writing style change.

CoT Prompt employs a series of intermediate reasoning steps to significantly improve the ability of LLMs in performing complex reasoning. It involves adding a simple prompt, "Let's think step-by-step," to guide the LLM through a logical reasoning process without requiring additional examples. In this task, we designed the CoT Prompt that asks the LLM for an overall assessment of the likelihood that the texts were written by the same author, considering linguistic features, without specifying which features to focus on.

Based on the CoT Prompt, we designed Feature-Prompt, asking the LLM to make the judgement based on five linguistic features: punctuation style, lexical and grammatical style, sentence structure and quantitative style, text and discourse style, and spelling and typographical errors. Before the overall assessment, it also involves asking the LLM for separate explanations and confidence scores of each feature, enabling a more structural and systematical judgment. The confidence scores of all the five linguistic features, along with the brief explanations behind scoring, provide more context and explainability to the assessment of whether the two given texts share a certain degree of similarity.

Both prompts require the LLM to provide the responses in a standard JSON format, enabling more efficient data extraction in later stages.

In this subtask, our goal is to delve into the LLM's capability of detecting the linguistic features of a text, whether the retrieved scores of the features can serve as a significant role in correctly predicting the commonalities and distinction between two texts, as well as which feature is more important than the others in the prediction when they are considered separately. In the evaluation stage, we first took the average scores of all five feature scores as the overall confidence scores for every paragraph pair, classified them under varying thresholds between 0.1 and 0.9 on the training set, and evaluated the validation set under the threshold with the highest F1 score. Second, we delved into the scores of each feature, separately, and got the evaluation results in the case where only one feature is considered at any given point. Finally, we utilized a simple logistic regression to train a classifier considering all five features and retrieved the optimized weights of the features.

CoT Prompt

Given the two texts:
Paragraph 1: paragraph1
Paragraph 2: paragraph2
On a scale of 0 to 1, with 0 indicating low confidence and 1 indicating high confidence, please provide a general assessment of the likelihood that the two texts were written by the same author. Your answer should reflect a moderate level of strictness in scoring, disregarding differences in topic and content. Focus on the linguistic features to determine if the texts are likely written by the same author.
In your analysis, give equal attention to identifying both the commonalities and distinctions between the texts to assess whether they share a similar writing style indicative of the same author.
Respond in a standard JSON format, with only a brief explanation (key ”explanation”) and a score (key ”score”).
Let's think step by step.

Feature-Prompt

Given the two texts:
Paragraph 1: paragraph1
Paragraph 2: paragraph2
On a scale of 0 to 1, with 0 indicating low confidence and 1 indicating high confidence, please provide a general assessment of the likelihood that the two texts were written by the same author. Your answer should reflect a moderate level of strictness in scoring, disregarding differences in topic and content. Focus on the following linguistic features to determine if the texts are likely written by the same author.
1.Punctuation Style: Hyphens, brackets, colons, commas, parentheses, quotation marks;
2.Lexical and Grammatical Features: Lexical variation and word choice; Grammatical categories and part of speech usage;
3.Sentence Structure and Quantitative Features: Sentence complexity, length, and arrangement; Coherence and cohesion; Word, clause, and sentence length, frequency, and distributions;
4.Text and Discourse Features: Narrative styles and speech events; Common expressions, idioms, tone and mood;
5.Spelling and Typographical Errors: Spelling mistakes and typographical errors.
In your analysis, give equal attention to identifying both the commonalities and distinctions between the texts to assess whether they share a similar writing style indicative of the same author.
First step: Understand the problem, Give the score of each feature. Then, carry out the plan and solve the problem step by step. Finally, show the overall confidence score.
Respond in a standard JSON format like: for each feature (name them feature1-5), there should be two pairs of keys and values (explanation and score), and finally, there should be an overall explanation and score.

4.2 BERT

We downloaded the BERT-base-cased [3] model from HuggingFace[4] and added a binary classification adaptor. For code implementation, we employed Transformers' "AutoModelForSequenceClassification" class for wrapping BERT, "Trainer" for the training process, and "Weights and Biases" for logging and hyperparameter searching.

After fine-tuning it on our constructed training set, we used the optimized BERT as a feature extractor. Specifically, we input the training set text into the fine-tuned BERT to extract the [CLS] token representations from the final layer, which served as feature vectors representing the writing styles of the texts. These feature vectors were then used to train a random forest classifier, which was validated on the development set. We conducted hyperparameter searches for both the fine-tuning of BERT and the training of the random forest classifier. Finally, we selected the optimal BERT and random forest models for testing.

Further, we visualized the attention weight distribution of the [CLS] token from the top layer of the BERT model, by averaging the attention weights from 12 attention heads. We use this distribution to analyze the information used by the model to distinguish writing style differences between paragraphs. Because some paragraphs are reused in the test set, we first calculated the sum of attention weights for all tokens, then we normalized these weights by dividing by the token frequency. Finally, these distributions are used to create word clouds. This visualization provided deeper insights into the information extracted by BERT.

4.3 Baseline Model: Random Forest with Combined Feature Representations

In our baseline approach, we employed a Random Forest classifier with a combination of feature representations, including TF-IDF, POS tags, character-level n-grams, and text statistical features. Each feature captures different aspects of writing style: TF-IDF highlights important words, POS tags provide syntactic information, character-level n-grams capture sub-word patterns and stylistic nuances, and text statistical features offer insights into the overall structure and complexity of the text. These features were vectorized and combined into a single feature matrix. Specifically, we used the TfidfVectorizer to transform each paragraph into a vector of TF-IDF scores, capturing the importance of words relative to the entire corpus. POS tags were extracted using NLTK's POS tagger and vectorized using DictVectorizer, which converted the POS tag counts for each paragraph into a numerical vector. Character-level n-grams (with n=2) were extracted from each paragraph and vectorized using DictVectorizer, converting the counts of each trigram into a numerical vector. Various statistical features, such as average word length, word count, unique word count, and lexical richness, were computed for each paragraph and vectorized using DictVectorizer. These combined features, along with the ground truth labels, were split into training and validation sets. The Random Forest classifier was then trained on the training set. To determine the hyperparameters for the Random Forest classifier, we referred to the parameters used in the study *Style Change Detection Using BERT Notebook for PAN at CLEF 2020*(Iyer & Vosoughi, 2020b). We selected 400 estimators based on their findings, which suggested this number to be effective for the task at hand. The model was subsequently evaluated on the validation set using the F1-score.

The performance of this baseline model serves as a comparison point for more advanced techniques, including fine-tuning BERT and employing prompt engineering.

[3] https://huggingface.co/google-bert/bert-base-cased
[4] https://huggingface.co/

5 Experiments

5.1 Dataset

PAN 24 provides three different datasets - Easy, Medium and Hard - all of which come from Reddit user posts and replies [5]. Our task considered the Medium level of the dataset, where the theme in a single document changes little (but still exists), forcing more attention to the writing style to effectively solve the detection task. The dataset is divided into two parts: training and validation set, including their ground truth data. Table 1 provides statistics of the dataset.

Dataset	documents	Avg. paragraphs per document	samples	positive	negative
Training Set	4200	6.22	21913	12493	9420
Validation Set	900	6.10	4592	2603	1989

Table 1: Statistics of the Dataset: #samples refers to the number of paragraph pairs in each set, # positive and #negative refer to the label 1 (author change occurs) and 0 (no author change) of each pair

For each set, each document (problem) is separated into multiple paragraphs. In each document, each paragraph set is paired with its corresponding ground truth label. Figure 1 is a brief illustration of the preprocessing of the training set. The validation set is preprocessed in the same way.

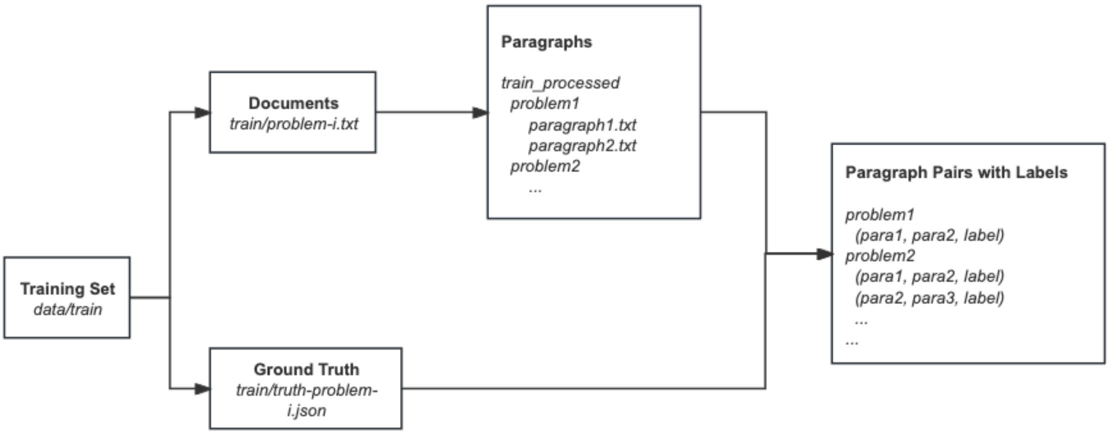


Figure 1: Preprocessing of the Training Set

The paragraph pairs in the original training set are then divided into a training set (80%, 17530) and a development set (20%, 4383), and the original validation set is used as a test set in the evaluation. Because of the input length limitation (512) of BERT, we assigned 256 to each paragraph and truncated them into this length limitation, during the data preprocessing stage.

5.2 Experiment Setup

5.2.1 Prompting

We chose gpt-3.5-turbo (API) as the LLM to test the zero-shot performance of our two prompts, with the temperature parameter set to 0.7, which strikes a balance between determinism and randomness,

[5]https://zenodo.org/records/10677876

which allows the model to capture subtle stylistic differences that could indicate an author change while still produce rather stable responses.

We loaded one paragraph pair to the prompt within each session, and subsequently iterated over all paragraph pairs within all problems (documents). Each generated response were stored separately under the corresponding problem, so that they were later paired with the ground truth labels of each problem.

5.2.2 BERT

First, we performed a full fine-tuning of our BERT model on a server, with two NVIDIA T4, Transformer 4.41.2, WandB 0.17.1, and Python 3.10.12. We primarily focused on three hyperparameters: batch size (ranging from 24 to 40), epoch (1, 2, and 3), and learning rate (ranging from $1 * 10^{-6}$ to $2 * 10^{-5}$). These search ranges were set based on the fine-tuning hyperparameters suggested in the original BERT paper (Devlin, Chang, Lee, & Toutanova, 2018b), considering both our computational resource limitations and the similarities between this task and BERT's pretraining tasks.

For training the random forest classifier, we used the optimally fine-tuned BERT model and kept its parameters fixed. We then initialized the hyperparameter search range for the random forest based on the methodology outlined in (Iyer & Vosoughi, 2020b). The number of estimators is selected from $[100, 200, 300, 400, 500, 600]$, and the random state is sampled from a uniform distribution in the span of $[2, 50]$.

During the testing and analysis phase, we conducted our work on a local computer, with a NVIDIA GTX 3060. We used the Huggingface Transformers 4.39.1 library to extract attention weights and the Wordcloud 1.9.3 [6] library to create the word clouds.

6 Results and Discussion

Task	Prompt-engineering	BERT	Fine-tuned BERT	Linguistic method (baseline)
Medium	0.594	0.795	0.806	0.811

Table 2: F1 scores on our test set of the Medium hard task

In Table 2, we show the F1-scores of our methods and the baseline model. The linguistic method (baseline) achieved the best performance.

6.1 Responses from Prompts

Table 3 shows an example of the generated responses from zero-shot CoT Prompt and Feature-Prompt, with the two paragraphs in the first document in the dataset. The original responses were in standard JSON format, but they are re-formatted here for better display, without modifying the content. Compared to the response from CoT Prompt, the response from Feature-Prompt provides a more structural context in determining the confidence score.

In this example, the generated response from CoT Prompt briefly includes some linguistic features such as tone and vocabulary choice (feature4 and feature2 in Feature-Prompt), without us specifying which specific features to consider. In the response from Feature-Prompt, the explanation of feature4 (narrative style and tone), however, addresses the differences in contents (even if we have required

[6] https://pypi.org/project/wordcloud/

the model not to focus on the contents in the prompt) and gives a below moderate score despite the similarity in tone, indicating a certain degree of unreliability of the provided scores. The overall confidence score in this response is exactly the average of the five feature scores, but it is not always the case since LLMs essentially lack the capability of numerical computation.

Prompts	**Generated Responses**
CoT Prompt	After analyzing the linguistic features of the two texts, it appears that they share some commonalities such as the use of complex sentence structures, including the employment of subordinate clauses and longer sentences. Both texts also exhibit a formal tone and employ relatively advanced vocabulary. However, there are notable distinctions in terms of content and subject matter, as well as slight differences in the level of formality and specific vocabulary choices. These differences suggest that while the writing style shows some similarities, the texts may have been written by different authors. **score: 0.6**
Feature-Prompt	**feature1:** Both texts exhibit similar use of punctuation marks such as commas, parentheses, and quotation marks. However, there are differences in the frequency and placement of these punctuation marks. **score: 0.7** **feature2:** The lexical and grammatical features show some similarities in terms of word choice and grammatical structures. However, there are noticeable differences in the vocabulary and sentence structures used in the two texts. **score: 0.5** **feature3:** While both texts display varied sentence lengths and some coherence in the narrative, there are distinct differences in the sentence complexity and overall organization of ideas. **score: 0.6** **feature4:** In terms of text and discourse features, both texts convey a serious tone and discuss political matters. However, the specific content and focus differ significantly between the two texts. **score: 0.4** **feature5:** There are no noticeable spelling or typographical errors in either text, indicating a certain level of attention to detail in writing. **score: 0.9** **overall:** Considering the analysis of the five features, the likelihood that the two texts were written by the same author is moderate, with some shared writing style features but also significant differences in language use and content. **score: 0.62**

Table 3: A sample of generated responses from both prompts

Figure 2 is a visualization of the F1 scores obtained from classifying the overall scores in the training set [7] using different thresholds, with both CoT Prompt and Feature-Prompt respectively. Here the overall scores with Feature-Prompt are calculated as the average of five feature scores within every paragraph pair. CoT Prompt is more effective at lower to mid thresholds, while Feature-Prompt performs significantly better at higher thresholds, probably due to a tendency to detect more similarities than distinctions of (a) certain feature(s) between two texts. The highest F1 scores with their corresponding threshold are listed in Table 4. There is no significant improvement in the performance with Feature-Prompt. One possible reason could be that GPT-3.5-Turbo is not efficient enough at extracting and identifying particularly specified linguistic features, and therefore not able to provide a superior overall assessment given the context of separately assessing each feature, than when it is not required to make the assessment based on specific features and instead just vaguely prompted to pay attention to the linguistic features.

With the best thresholds, the F1 scores on the validation set are 0.518 and 0.581, for scores from CoT Prompt and Feature-Prompt respectively.

[7]In this stage, we didn't consider splitting the original training set into a new training set and a development set, as essentially we cannot train GPT through constantly prompting it in a new session. So the training set was used to find the best thresholds, which were later used in evaluating the validation set

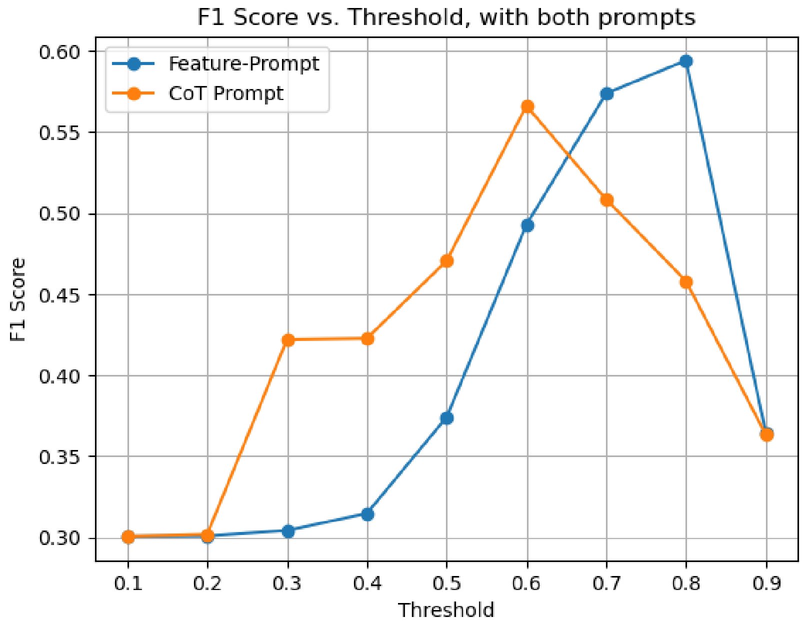


Figure 2: F1 scores across different thresholds evaluated on the training set, with both prompts

	Best threshold	Highest F1 score
CoT Prompt	0.6	0.566
Feature-Prompt	0.8	0.594

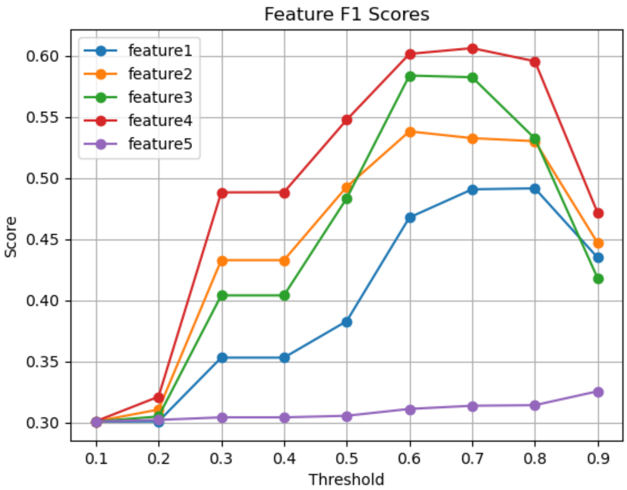
Table 4: Highest F1 scores obtained from both prompts, with corresponding thresholds

Figure 3 shows the F1 scores for scores of each of the five features across various thresholds, which is equivalent to the results when each feature is considered as the sole feature used in determining author change between two texts. The scores at threshold=0.8 are (0.491, 0.530, 0.532, 0.595, 0.314). In this case, narrative tone (feature4) seems to most significantly contribute to the prediction, while misspellings and typos (feature5) the least. The other three features share a similar significance at a moderate level. We also observed that the scores of feature5 across all data are generally higher than the scores of other features, with a large proportion being 0.8 or higher. This is either because the texts in this dataset vary less in the amount of spelling and typographical errors (all at a low level) or because GPT-3.5-Turbo is weak at detecting them.

Another point to note here is that, the analysis of separate feature scores is based on our assumption that the feature scores provided by the responses prompted by Feature-Prompt, given the entire context of the assessment of all the five features, are sufficiently close to the scores obtained by including only one feature in a prompt at a single time. In other words, we assumed that the assessment of each feature is largely independent.

We then trained a simple logistic regression model to further levarage the feature scores given in the responses and delve into the differences of the impacts of the features on the prediction [8]. We again iterated through different thresholds on the development set, and got 0.5 as the threshold that gives the best performance. The F1 score evaluated on the test set is 0.594, approximately the same as using the average scores of all five features as the overall scores. Table 5 shows the optimized feature weights after training. We can draw similar conclusions as the previous analysis of the F1 scores of the five features, that sentence structure and narrative tone seem to serve the most signif-

[8]In this stage, the original training set is split as stated in 5.1



feature1	Punctuation style
feature2	Word choice and grammar
feature3	Sentence structure
feature4	Narrative tone
feature5	Misspellings and typos

Figure 3: F1 scores of each linguistic feature when considered alone, across different thresholds, on the training set

icant roles in assessing the similarity level between two texts. However, we can also see from here that the punctuation style displays a slight negative effect, given that its optimized weight is below 0.

Feature	Punctuation	Words and Grammar	Sentence	Tone	Spelling
Weight	-0.125	0.054	1.387	1.760	0.174

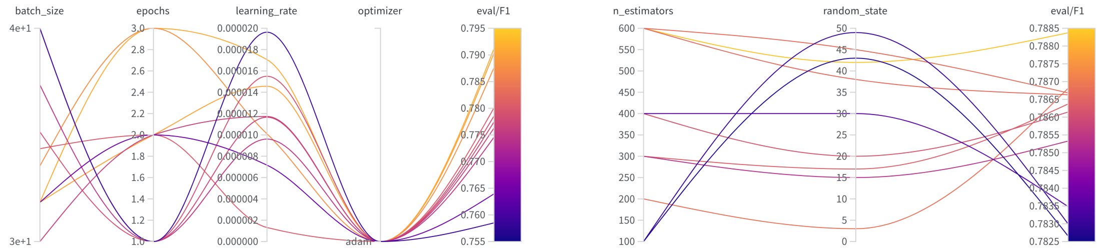
Table 5: Optimized weights of five features, after training the classifier

Achieving explainability in GPT-3.5-Turbo's responses on detecting author writing styles and rate the similarities between two given texts based on linguistic features is notoriously difficult, but Feature-Prompt can address to this challenge to a certain extent, by providing detailed, structural and interpretable solutions. Feature-Prompt offers a comprehensive comparison of linguistic features, while CoT Prompt delivers a more global, vague analysis.

However, it is also important to note that the explanations generated by GPT-3.5-Turbo and generally other LLMs, which appear to demonstrate reasoning processes, are intrinsically based on statistics instead of actual comprehension or logic processing. Even though GPT-3.5-Turbo is able to generate explainable responses by providing step-to-step explanations when prompted by Feature-Prompt, the scores it assigned can still be largely prone to randomness. Considering this, prompt engineering may not be the optimal approach to a task like this.

6.2 BERT

We employed Weights and Biases for hyperparameter searching. It is a powerful tool for managing training logs and visualization. Figure 4a shows that the number of epochs is the biggest factor for the final F1-score, the model is less sensitive to the other two hyperparameters, i.e. the learning rate and the batch size. The best hyperparameter set is batch size 27, epoch 3 and learning rate 0.00001707. Based on the best fine-tuned BERT, we calculated the embeddings of paragraph pairs. Then, we fed those embeddings to a random forest classifier for training and testing. Figure 4b shows that the classifier heavily depends on the number estimators. The optimized classifier has 600 estimators and 42 as the random seed. The combination of the fine-tuned BERT and the random forest classifier reaches a 0.806 F1 score on the test set. This promising result suggests that BERT is a good feature extractor.



(a)

(b)

Figure 4: The hyper-parameter sweep of (a) BERT-base-cased model and (b) the random forest classifier.

Compared to the zero-shot BERT, our model has a 1.1% improvement, which may indicate the weak effect of fine-tuning on this task. There are many possible reasons for that. The model may have learned new features that are related to writing styles. it can also be that the model reaches its performance bottleneck which is mentioned in Iyer's work (Iyer & Vosoughi, 2020b). However, it is clear that the BERT features learned from its pre-training tasks are highly effective for writing style change detection, which is significantly important for applications. People can apply pre-trained BERT without fine-tuning it.

To further understand the result, we calculated the average attention score of tokens from the validation set (our test set), shown in Figure 5. Figure 5a is the word cloud of identified authorship-inconsistency samples; Figure 5c shows the attention distribution of identified authorship-consistency samples; Figure 5d is the word cloud for paragraph pairs without authorship change but being classified as authorship-inconsistency; Figure 5b shows the result of misclassified samples with authorship-change.

Overall, BERT attends to a wide range of words and features in paragraphs, e.g. lexical and content-specific information. Specifically, BERT prefers casual expressions, formal expressions, and topic-related words (names, keywords, concrete nouns, etc.). Cross the four word clouds, we can see informal expressions, such as "Hi", "Hello", "Hopefully", "Yep", etc.; more formal expressions, such as "occupants", "ballots", "defeated", etc.; names, such as "Steven", "Stuttgart", "Santos", etc.; concrete nouns, such as "outlets", "pot", "Picture", etc; and keywords, such as "##PP", "##ot", "##dit", etc. All of these features are combined to determine whether there is an authorship change. The language register is also a powerful and common way for humans to tell the difference between content from different authors. The language register itself is highly related to vocabulary richness that results in different writing styles.

Another pattern we observed is that for the samples classified as having authorship change, BERT focuses more on keywords, language register and topic-related words. However, for those classified as authorship-consistent samples, BERT pays more attention to keywords and topic-related words. this may suggest that keywords and topic-related words are powerful features for telling that the two paragraphs are not written by the same author. However, it also confuses the model when the contents are actually from the same author. Whereas, the language register is a more concrete indicator of writing style and contributes more when there is an authorship change.

(a) (b)

(c) (d)

Figure 5: The attention-weighted word cloud: (a) correctly and (b) misclassified authorship changes; (c) correctly and (d) misclassified authorship consistency.

6.3 Baseline

In our study, the baseline model using a Random Forest classifier with a combination of various feature representations, including TF-IDF, POS tags, character-level n-grams, and text statistical features, achieved surprisingly high performance. This suggests that our feature selection and hyperparameter tuning for the baseline model were particularly effective. Specifically, the combination of lexical, syntactic, and statistical features provided a comprehensive representation of the writing styles, enabling the Random Forest classifier to perform exceptionally well.

The surprisingly outstanding performance of the baseline model may be attributed to the following reasons:

- Feature engineering: The selected features (TF-IDF, POS tags, character-level n-grams, and text statistical features) are highly effective in capturing the unique writing styles of different authors. TF-IDF features capture the lexical choices and topic distribution of each paragraph, helping us highlight words that differ in frequency among authors. POS tag features reflect the grammatical structure and sentence patterns of each paragraph, effectively distinguishing the word usage habits of different authors. Character-level n-gram features capture fine-grained writing style clues such as affixes and roots. We chose n=2, which helps us capture more of the writing style and spelling patterns within paragraphs. Textual statistical features, such as average sentence length and lexical richness, reflect the structure and complexity of each paragraph. These features provide insights into the overall structure and complexity of the text, aiding in distinguishing different writing styles. The fusion of these features provides the Random Forest classifier with rich data points for learning.

- Hyperparameter tuning: Based on prior research, we carefully tuned the hyperparameters, ensuring that the model was well-optimized for this particular task.
- The dataset: The dataset comprises multi-author texts from the social media platform Reddit, where the authors' writing styles may share certain common "internet tones." The statistical text features used in the baseline model, such as punctuation frequency and sentence length distribution, may be more sensitive to this type of data.
- Task objective: The objective of this task is to determine whether there is a change in authorship between paragraph pairs, which can be viewed as a binary classification problem. Traditional classifiers like Random Forests tend to excel at handling such problems, while BERT and LLMs are more adept at sequence modeling and generation tasks, potentially underperforming in classification tasks.

However, despite the baseline model's efficiency and practicality, it may struggle to scale up to larger datasets or more complex texts. In contrast, advanced models like BERT and GPT, although currently underperforming, still hold the potential to surpass the baseline model's performance through further optimization and expansion of the dataset.

7 Conclusion

In this paper, we discussed three approaches to solving the multi-author style change detection task: the Baseline approach (utilizing traditional features such as TF-IDF, POS tags, and character-level n-grams with a Random Forest classifier), the BERT approach (involving fine-tuning), and the prompt engineering approach (using GPT-3.5-Turbo).

Surprisingly, the baseline model achieved a slightly higher F1 score than the fine-tuned BERT and prompt-based methods. The baseline approach offered a more straightforward and resource-efficient alternative, while the BERT-based model showed a remarkable capability in detecting writing style changes. Future research could explore hybrid models that combine the strengths of both approaches, leveraging BERT's deep contextual understanding alongside the baseline model's feature-based insights. This could further enhance performance and applicability in diverse text analysis scenarios.

On the other hand, through our experiments of the prompt engineering approach, we have been faced with a limited capability of GPT-3.5-Turbo in accurately capturing the nuances of linguistic features in texts, and subsequently, in accurately detecting author change, compared to the BERT and Baseline approaches. With the use of Feature-Prompt, we are able to delve into a more structural reasoning process that leads to determining the confidence scores compared to using CoT Prompt. However, the evaluation results are not significantly superior, due to various limitations with both the power of GPT-3.5-turbo and our computational resource. Possible approaches to enhance the performance of using prompt engineering on this task include:

- Consider few-shot prompting besides zero-shot prompting.
- Consider other LLMs with larger capacity, such as GPT-4.
- Consider more linguistic features, such as special character style, or acronyms and abbreviations.

References

Barlas, G., & Stamatatos, E. (2020, June). Cross-Domain Authorship Attribution Using Pre-trained Language Models. In I. Maglogiannis, L. Iliadis, & E. Pimenidis (Eds.), *16th IFIP International Conference on Artificial Intelligence Applications and Innovations (AIAI)* (Vol. AICT-583, p. 255-266). Neos Marmaras, Greece: Springer International Publishing. Retrieved from https://inria.hal.science/hal-04050592 (Part 5: Neural Network Modeling) doi: 10.1007/978-3-030-49161-1_22

Beltagy, I., Lo, K., & Cohan, A. (2019). *Scibert: A pretrained language model for scientific text.*

Bevendorff, J., Chulvi, B., Fersini, E., Heini, A., Kestemont, M., Kredens, K., ... Zangerle, E. (2022, 08). Overview of pan 2022: Authorship verification, profiling irony and stereotype spreaders, and style change detection. In (p. 382-394). doi: 10.1007/978-3-031-13643-6_24

Bevendorff, J., Chulvi, B., la Peña Sarracén, G. L. D., Kestemont, M., Manjavacas, E., Markov, I., ... Zangerle, E. (2021). Overview of pan 2021: Authorship verification, profiling hate speech spreaders on twitter, and style change detection - extended abstract. In *European conference on information retrieval.* Retrieved from https://api.semanticscholar.org/CorpusID:232423066

Chen, B., Zhang, Z., Langrené, N., & Zhu, S. (2024). *Unleashing the potential of prompt engineering: a comprehensive review.*

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018a). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, *abs/1810.04805*. Retrieved from http://arxiv.org/abs/1810.04805

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018b). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

De Vries, W., Van Cranenburgh, A., & Nissim, M. (2020). What's so special about bert's layers? a closer look at the nlp pipeline in monolingual and multilingual models. *arXiv preprint arXiv:2004.06499*.

Hasan, T., Bhattacharjee, A., Samin, K., Hasan, M., Basak, M., Rahman, M. S., & Shahriyar, R. (2020, November). Not low-resource anymore: Aligner ensembling, batch filtering, and new datasets for Bengali-English machine translation. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)* (pp. 2612–2623). Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.emnlp-main.207 doi: 10.18653/v1/2020.emnlp-main.207

Hashemi, A., & Shi, W. (2023). Enhancing writing style change detection using transformer-based models and data augmentation. *Working Notes of CLEF*.

Huang, B., Chen, C., & Shu, K. (2024). *Can large language models identify authorship?*

Hung, C.-Y., Hu, Z., Hu, Y., & Lee, R. K.-W. (2023). *Who wrote it and why? prompting large-language models for authorship verification.*

Iqbal, F., Khan, L. A., Fung, B. C. M., & Debbabi, M. (2010). e-mail authorship verification for forensic investigation. In *Proceedings of the 2010 acm symposium on applied computing* (p. 1591–1598). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/1774088.1774428 doi: 10.1145/1774088.1774428

Iyer, A., & Vosoughi, S. (2020a). Style change detection using bert. In *Conference and labs of the evaluation forum.* Retrieved from https://api.semanticscholar.org/CorpusID:225073958

Iyer, A., & Vosoughi, S. (2020b). Style change detection using bert. *CLEF (Working Notes)*, *93*, 106.

Koroteev, M. (2021, 03). *Bert: A review of applications in natural language processing and understanding.* doi: 10.48550/arXiv.2103.11943

Lin, T.-M., Chen, C.-Y., Tzeng, Y.-W., & Lee, L.-H. (2022). Ensemble pre-trained transformer models for writing style change detection. In *Clef (working notes)* (pp. 2565–2573).

Patel, A., Rao, D., Kothary, A., McKeown, K., & Callison-Burch, C. (2023). *Learning interpretable style embeddings via prompting llms.*

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, *21*(140), 1–67. Retrieved from http://jmlr.org/papers/v21/20-074.html

Shu, K., Wang, S., Lee, D., & Liu, H. (2020). *Mining disinformation and fake news: Concepts, methods, and recent advancements.*

Stamatatos, E., & Koppel, M. (2011, 03). Plagiarism and authorship analysis: Introduction to the special issue. *Language Resources and Evaluation*, *45*, 1-4. doi: 10.1007/s10579-011-9136-1

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., … Zhou, D. (2023). *Chain-of-thought prompting elicits reasoning in large language models.*

Zhang, Z., Ren, S., Liu, S., Wang, J., Chen, P., Li, M., … Chen, E. (2018). *Style transfer as unsupervised machine translation.*

A Questions

A.1 Question 1

Q1. Give a brief account of constituent and dependency structure and discuss in what way features deriving from syntactic analysis can be applied to style detection and/or authorship attribution.

The explanation of constituent and dependency structure referenced the book *Daniel Jurafsky, James H. Martin (2023). Speech and Language Processing. Draft of Jan, 2023.*

Constituents are the building blocks of sentences. They are groups of words that function together as a single unit within a sentence's hierarchical structure. These components include phrases such as noun phrases (NP), verb phrases (VP), prepositional phrases (PP), etc. For example, in the sentence "The dog ate a meatball," the constituent structure analysis would group "The dog" as a noun phrase (NP), "ate" as a verb phrase (VP), and "a meatball" as another noun phrase (NP). The sentence can be represented as: [S [NP The dog] [VP ate [NP a meatball]]]. Dependency structure focuses on the grammatical relationships between words in a sentence. Unlike constituent parsing, which groups words into larger phrase units, dependency parsing represents syntactic structure through binary relationships between words, with each relationship directed from a headword to a dependent word. For example, in the sentence "The dog ate a meatball," the dependency structure shows "ate" as the root of the sentence, serving as the main verb. "Dog" is the subject of "ate," and "meatball" is the object of "ate." Additionally, "the" modifies "dog," and "a" modifies "meatball." This structure intuitively displays the relationships between words, clarifying the function and connection of each word within the sentence.

By using NLP tools and libraries such as NLTK, spaCy, and Stanford CoreNLP, we can extract various features from sentences. For example:

- Word Structure Preferences: This includes the distribution and frequency of parts of speech such as nouns, verbs, and adjectives. Features extracted from constituent structures, such as the frequency and distribution of noun phrases (NP), verb phrases (VP), etc.
- Syntactic Preferences: Analyzing the syntactic roles of words in a sentence (e.g., subject, object, modifier) reveals the author's preferences in sentence construction.
- Sentence Complexity: Analyzing the depth of syntactic and phrase structure nesting reflects the complexity of sentences.
- Dependency Relations Features: Extracting features from dependency structures, such as the frequency and distribution of subject-verb relations, verb-object relations, etc.

After extracting the relevant syntactic features from the text, we can vectorize these features, similar to what we did in our baseline. The vectorized syntactic features can be used as inputs for various supervised machine learning classifiers, such as Random Forests, Support Vector Machines, and Naive Bayes classifiers. We can train these machine learning models to detect writing styles or perform authorship attribution. Additionally, we can leverage the capabilities of deep learning models, such as Recurrent Neural Networks (RNN) or Transformers, to automatically learn representations from syntactic features and perform style or author prediction. For instance, RNNs can capture sequential information and contextual relationships in sentences through their recursive structure, while Transformer models can effectively model long-range dependencies and syntactic complexity through self-attention mechanisms. These models can capture the complex patterns and long-range dependencies present in the feature space, thereby enhancing the accuracy and effectiveness of classification.

A.2 Question 2

Q2. Imagine you were asked to perform the same task for an online forum, such as StackExchange, where the language style is different to the one from the current task. You are given a small data set, which has less than one-third of the current data instances. How would you proceed to solve this problem? Provide a brief description of the method you would use, give a justification for it, and explain why you expect the method would work (based on literature and/or your own justification).

To solve the same task for an online forum such as StackExchange, given a smaller dataset, we would consider using a combination of a data augmentation strategy and transfer learning with a BERT model that is fine-tuned specifically for this task.

Data augmentation can be used to artificially increase the amount of training data and improve the model's robustness, given the small size of the dataset. Traditional techniques of data augmentation, such as synonym replacement and back-translation, can however alter the original writing style, which is counterproductive when our ultimate goal is to detect changes in the writing style. In this case, we can adjust the strategy to generate valuable augmented data that preserves the authors' unique styles. We can consider the following style-preserving data augmentation techniques:

- Paraphrasing with contextual similarity: Use models like T5 and GPT with prompts tailored to preserve the writing style; implement paraphrasing constraints to ensure the generated text retains the original style and context as much as possible (Raffel et al., 2020).
- Contextual augmentation: Create new text by interpolating between existing segments written by the same author, which can help generate diverse yet style-consistent examples (Hasan et al., 2020).

For the transfer learning, we would consider taking the BERT model that was pre-trained on the current task and fine-tuning it on the contents on StackExchange. An alternative strategy is to consider domain-specific BERT models like SciBERT (Beltagy, Lo, & Cohan, 2019), which was pre-trained on data more similar to the contents on StackExchange, while the texts provided in our current task were collected from Reddit user posts and replies and the content may varies in terms of topics and/or lack scientific authority. Given the small size of the dataset, we would employ the following techniques during fine-tuning:

- Cross-validation: Use k-fold cross-validation to make the most out of the available data and to ensure the model's performance is reliable.
- Early stopping: Monitor the validation loss and stop training when it starts to increase, to prevent overfitting.
- Regularization: Use dropout layers and L2 regularization to further prevent overfitting.

To maximize the benefits from both data augmentation and fine-tuning the BERT model, our strategy is to use the writing style embeddings learned from the original texts by the pre-trained model to guide the generation of new texts (Zhang et al., 2018), and fine-tune the model on the combined dataset of both the original and augmented data. We should ensure that the fine-tuning process emphasizes maintaining the integrity of the author's writing style. Besides, we should iteratively refine the augmentation and fine-tuning process based on evaluation results.

With this adjusted approach, we expect an improved performance of the fine-tuned BERT model, since it will be trained on a more extensive dataset that captures the subtle differences and consistencies in author styles. We also expect that the augmented data will help the model generalize better to new, unseen data, making it more robust in detecting writing style changes across different contexts and authors.