Does CS1050's Chatham House Policy Protect Against LLM-Powered Stylometric Linkage De-Identification?

Kento Nishi
Harvard College
kentonishi@college.harvard.edu

Abstract

CS1050 operates under the Chatham House Rule, which permits sharing ideas from class discussions but prohibits attributing them to individuals by name. This norm assumes that removing explicit identifiers suffices for anonymity, yet recent advances in LLM-powered authorship attribution suggest that writing style itself can serve as an identifier. I tested whether anonymized Canvas discussion posts could be re-identified by matching them against external writing samples, using a fine-tuned DeBERTa classifier and Linguistically-Informed Prompting with Gemini 2.5 Flash. On 151 posts from eight consenting participants, the fine-tuned model achieved 24.5% top-1 accuracy while the prompting method achieved 20.5%, compared to a 12.5% random baseline. Both methods beat chance but neither achieved accuracy high enough for confident deanonymization, suggesting that the Chatham House Rule still protects privacy against current stylometric techniques. Sanitized source code is available at github.com/KentoNishi/CS1050-Stylometry.

1 Introduction

In CS1050, we operate under the Chatham House Rule: ideas shared in class may be discussed externally, but they may not be attributed to individual speakers by name. The rule assumes that stripping names is enough for anonymity, but class readings on re-identification attacks made me wonder whether that assumption still holds. Recent work by Huang et al. [2024] shows that modern transformer models can re-identify authors based solely on writing style, raising an uncomfortable question: does the Chatham House Rule actually protect us, or can an adversary with the right tools figure out who said what?

I set out to test this empirically by attempting to deanonymize Canvas discussion posts from our own class. Eight classmates volunteered external writing samples, and I tried two attribution methods—a fine-tuned DeBERTa classifier and Linguistically-Informed Prompting with Gemini 2.5 Flash—to see if I could match anonymous posts back to their authors. If re-identification accuracy turned out high, that would challenge the Chatham House assumption; if it stayed low, the rule might still offer meaningful protection against stylometric linkage attacks.

2 Background

The Chatham House Rule. The Royal Institute of International Affairs established the Chatham House Rule in 1927 to enable frank discussion without fear of attribution [Chatham House, 2002]. The modern formulation states that "when a meeting, or part thereof, is held under the Chatham House Rule, participants are free to use the information received, but neither the identity nor the affiliation of the speaker(s), nor that of any other participant, may be revealed." CS1050 adopted this rule so students could discuss surveillance practices and corporate data policies without worrying that their views might follow them into future job interviews. The implicit assumption is that stripping names is enough, and this project tests whether that assumption holds.

Stylometric Authorship Attribution. Stylometry—the statistical analysis of writing style—has been unmasking anonymous authors for decades. The foundational study by Mosteller and Wallace [1963] used Bayesian methods to settle a centuries-old debate about the Federalist Papers, concluding that James Madison likely wrote all twelve disputed essays based on word frequency patterns alone. In 1996, Vassar professor Donald Foster identified journalist Joe Klein as the author of the anonymously published novel *Primary Colors*, and in 2013 Patrick Juola confirmed that J.K. Rowling had written *The Cuckoo's Calling* under the pseudonym Robert Galbraith [Juola, 2013]. The lesson from these cases is clear: stylistic fingerprints can persist even when authors actively try to hide.

Modern transformers like DeBERTa learn dense representations that capture stylistic patterns without hand-engineered features. The BEDAA approach [Zahid et al., 2025] fine-tunes DeBERTa for authorship attribution using multiple loss functions to push same-author documents together in embedding space, while Huang et al. [2024] introduced Linguistically-Informed Prompting (LIP), which guides LLMs to focus on specific stylistic cues like function words and punctuation—achieving competitive results without any fine-tuning at all.

3 Data

Collection. Eight classmates agreed to participate after I explained the project through a Google Form. Each volunteer submitted external writing samples—essays and position papers from other courses—and gave permission for me to analyze their Canvas discussion posts from this semester. The Canvas corpus spans 19 discussion sessions from September 4 through November 18, totaling 151 posts. Topics ranged from surveillance and panopticism early on to differential privacy and algorithmic fairness toward the end, so any stylistic signatures the models detect must generalize across substantial thematic variation.

Participant	Canvas Posts	External Documents
A	19	6
B	19	5
C	19	4
D	18	5
E	19	6
F	19	5
G	19	6
H	19	4
Total	**151**	**41**

Table 1: Number of Canvas discussion posts and external writing samples per participant. Samples include essays and position papers from other courses submitted voluntarily via a Google Form.

Pretraining Corpus. With only 4–6 documents per participant, I needed an intermediate training stage before specializing to the eight target authors. I assembled a pretraining corpus from the Blog Authorship Corpus [Schler et al., 2006] by selecting ten authors at random, cleaning their posts (whitespace normalization, HTML removal), and truncating each to 2,000 characters. The resulting 1,536 examples share the informal register of Canvas discussions without containing any text from actual participants. I applied consistent preprocessing everywhere to avoid leaking identity through metadata: Unicode normalization, whitespace collapsing, and stripping quoted reply text and @-mentions from Canvas posts. Stylistic signals like punctuation patterns and capitalization habits remained intact.

4 Methods

Fine-tuned Classifier. The first method adapts DeBERTa-base for authorship attribution using the BEDAA multi-loss training approach [Zahid et al., 2025], passing its [CLS] token representation through a linear classification head that outputs a probability distribution over the eight candidate authors. I implemented the training pipeline in PyTorch using the HuggingFace Transformers library,

with a custom `LossManager` class that computes a weighted combination of four loss terms following the BEDAA approach:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + 0.5\,\mathcal{L}_{\text{focal}} + 0.5\,\mathcal{L}_{\text{SCE}} + 0.2\,\mathcal{L}_{\text{contrastive}}$$

where cross-entropy provides the primary classification signal, focal loss down-weights well-classified examples to focus on difficult cases, symmetric cross-entropy adds a reverse KL term for robustness to label noise, and supervised contrastive loss encourages same-author documents to cluster in the representation space.

I trained the model in two stages: Stage 1 fine-tunes the full model on the blog corpus for 6 epochs with learning rate 2×10^{-5}, adapting DeBERTa's representations to authorship discrimination generally, while Stage 2 freezes the transformer backbone and trains only the classification head on the volunteer corpus for 16 epochs with learning rate 6×10^{-5} for the head. To augment the limited training data in Stage 2, I implemented a `RandomWindowDataset` class that samples 32 random contiguous windows per document per epoch, each containing at least 32 tokens, rather than always using the same truncated prefix. Both stages use the AdamW optimizer with batch size 8, weight decay 0.01, 10% linear warmup, gradient clipping at norm 1.0, and automatic mixed precision via PyTorch's `GradScaler`. A `WeightedRandomSampler` balances the training batches by inverse class frequency to handle the uneven distribution of documents across authors. At inference time, the model estimates uncertainty via Monte Carlo Dropout by running $T = 5$ forward passes with dropout active and averaging the resulting softmax distributions.

Linguistically-Informed Prompting. The second method uses Linguistically-Informed Prompting (LIP) with Gemini 2.5 Flash, following Huang et al. [2024]. I implemented a retrieval-augmented pipeline that, for each Canvas post to be attributed, computes TF-IDF vectors (unigrams and bigrams, up to 20,000 features) over the external document corpus and retrieves the 10 most similar candidate documents by cosine similarity. The pipeline serializes these candidates as a JSON object mapping integer keys to document text, then constructs a prompt instructing Gemini to analyze the query post's writing style while attending to phrasal verbs, modal verbs, punctuation, rare words, affixes, quantities, humor, sarcasm, typographical errors, and misspellings. The prompt explicitly instructs the model to disregard topic and content differences and focus only on linguistic features. I configured the Gemini API to return structured JSON with an "analysis" field containing the model's reasoning and an "answer" field containing the predicted author index, using temperature 0.0, top-p 1.0, and maximum output length 1,024 tokens to minimize randomness. Each external document was truncated to 2,000 characters to fit within context limits while preserving stylistic signal.

5 Results

Overall Accuracy. Table 2 summarizes attribution accuracy across all 151 Canvas posts; with eight candidate authors, random guessing achieves 12.5% top-1 accuracy, 25.0% top-2, and 62.5% top-5.

Method	Top-1	Top-2	Top-5
Random baseline	12.5%	25.0%	62.5%
Gemini LIP	20.5%	—	—
Fine-tuned DeBERTa	24.5%	43.7%	73.5%

Table 2: Attribution accuracy on 151 Canvas discussion posts. Top-k accuracy measures whether the true author appears among the model's k highest-probability predictions. The random baseline reflects uniform guessing over eight candidates. LIP returns only a single prediction, so top-k metrics are unavailable.

The fine-tuned model correctly identifies the author about one in four times, meaning it still errs three times out of four. LIP trails by 4 percentage points at 20.5%. Both methods beat random guessing, but neither approaches the accuracy needed for confident deanonymization. Accuracy varied considerably across the 19 discussion sessions, ranging from 0% to 50%, though with only eight posts per session this likely reflects sampling noise rather than anything systematic about topic.

Variation Across Participants. Table 3 breaks down accuracy by participant, revealing substantial variance. The fine-tuned model correctly identified Participant G 58% of the time but never once identified Participants A or B. The two methods also disagreed about who was identifiable: Participant C was identified at 37% accuracy with fine-tuning but 68% with LIP, while Participant H showed the reverse pattern.

Participant	Posts	Fine-tuned	LIP
G	19	57.9%	31.6%
H	19	47.4%	5.3%
F	19	36.8%	10.5%
C	19	36.8%	68.4%
E	19	10.5%	26.3%
D	18	5.6%	0.0%
B	19	0.0%	10.5%
A	19	0.0%	10.5%

Table 3: Top-1 attribution accuracy by participant, sorted by fine-tuned model performance. The two methods exhibit low correlation in which participants they identify correctly: Participant C is highly identifiable by LIP (68.4%) but only moderately by fine-tuning (36.8%), while Participant H shows the reverse pattern.

6 Discussion

Neither method achieved accuracy high enough for confident deanonymization. At 24.5% and 20.5%, both approaches beat random guessing but would still get it wrong three to four times out of five. An adversary using these methods would misattribute the vast majority of posts. This is reassuring: writing style does carry some identifying signal (both methods beat chance), but that signal is not strong enough to break anonymity given only 4–6 external documents per participant.

I was surprised by the variance across participants. Some people apparently write more distinctively than others, but even for the most identifiable participant (G at 58%), the model still erred more often than it succeeded. For most participants, the risk of accurate attribution remained low. The disagreement between methods was also unexpected—Participant C was highly identifiable to Gemini but only moderately so to the fine-tuned model, while Participant H showed the opposite pattern. Whatever stylistic features each method latches onto, they are clearly not the same.

Out of curiosity, I also tried to manually attribute some posts myself. I found it was nearly impossible, mainly because the Canvas posts discuss CS1050 topics—i.e., Warren and Brandeis, targeted advertising, differential privacy—while the external reference samples were on completely unrelated subjects like Chinese suicide rates or environmental pollution. The topic domains were so different that I could not see past the content to notice any stylistic similarities. This probably explains some of the difficulty for the models as well.

Two outcomes were possible: high accuracy would mean the Chatham House Rule no longer protects privacy, while low accuracy would suggest writing style is not easily linkable across contexts. The results support the second outcome. Stylometric methods can do better than guessing, but they cannot confidently deanonymize posts in this setting.

7 Limitations

Eight participants is a small cohort, and the variance I observed may not hold in a larger population. The external reference corpus was thin—only 4–6 documents per participant—and more auxiliary data might well improve attribution accuracy. I also only covered one semester of writing, so style drift over time could change the picture. Finally, everything here concerns English text; results might differ in multilingual settings.

8 Ethical Considerations

All participants completed an informed consent form explaining the project's purpose and risks. This report presents only aggregate statistics and anonymized per-participant breakdowns using single-letter identifiers; no participant was identified against their will. I have not released the trained models or code to prevent misuse.

9 Conclusion

This project asked whether CS1050's Chatham House anonymity holds up against LLM-powered stylometric attacks, and based on experiments with eight volunteers and two attribution methods, the answer is: yes, for now. The fine-tuned DeBERTa model achieved 24.5% accuracy and LIP prompting achieved 20.5%—both better than random guessing but far from confident deanonymization. Writing style does carry identifying signal, but that signal is not strong enough to break anonymity given limited auxiliary data.

The Chatham House Rule still protects privacy for our class discussions. That protection may erode as language models improve or if adversaries obtain substantially more reference material, but for now the threat level remains modest.

References

Chatham House. Chatham house rule. The Royal Institute of International Affairs, 2002. https://www.chathamhouse.org/about-us/chatham-house-rule.

Baixiang Huang, Canyu Chen, and Kai Shu. Can large language models identify authorship? *arXiv preprint arXiv:2403.08213*, 2024. Accepted to EMNLP 2024 Findings.

Patrick Juola. Rowling and "Galbraith": An authorial analysis. *Language Log*, July 2013. https://languagelog.ldc.upenn.edu/nll/?p=5315.

Frederick Mosteller and David L Wallace. Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed Federalist papers. *Journal of the American Statistical Association*, 58(302):275–309, 1963.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and gender on blogging. AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs, 2006.

Iqra Zahid, Youcheng Sun, and Riza Batista-Navarro. BEDAA: Bayesian enhanced DeBERTa for uncertainty-aware authorship attribution. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 17952–17966, Vienna, Austria, 2025. Association for Computational Linguistics.

A Implementation Details

Data Preprocessing. The `parse_canvas.py` script converts raw Canvas discussion exports into structured JSON. It extracts the discussion topic and date from header lines, parses "Reply from *Name*" markers to segment individual posts, and applies name canonicalization to handle variations in how Canvas displays author names (e.g., removing parenthetical notes, normalizing whitespace). The output is one JSON file per discussion session containing the date, topic, list of questioners, and an array of comment objects with `author_name` and `post_text` fields. The `prepare_blog_corpus.py` script preprocesses the Kaggle Blog Authorship Corpus for Stage 1 pretraining. It selects the top k most prolific authors (default 10), applies text cleaning (collapsing whitespace, removing hyphenated line breaks), filters posts to those with 100–2,000 words, and writes each post as a separate `.txt` file under an author-specific directory. This structure mirrors the volunteer corpus layout, allowing the same data loading code to be reused across training stages.

Loss Functions. The `bedaa/losses.py` module implements the multi-loss training objective through a `LossManager` class that accepts a list of loss names and weights at initialization and computes their weighted sum during training. Cross-entropy provides the primary classification signal via `nn.CrossEntropyLoss`, while focal loss down-weights well-classified examples by a factor of $(1 - p_t)^\gamma$ to focus the model on difficult cases. Symmetric cross-entropy adds a reverse KL term that measures divergence from predictions to labels, providing robustness to label noise, and supervised contrastive loss pulls same-author embeddings together in the [CLS] representation space using temperature-scaled cosine similarity.

Training Pipeline. The `train_bedaa.py` script implements the two-stage training procedure. The `TextDataset` class provides standard tokenization with truncation to `max_length`, while the `RandomWindowDataset` class augments limited training data by sampling multiple random contiguous windows from each document per epoch, each at least `min_tokens` long, multiplying effective training examples without repeating identical truncations. The `compute_class_balance` function returns inverse-frequency weights for each class, which `build_sampling_weights` uses to construct per-sample weights for PyTorch's `WeightedRandomSampler` to balance batches across authors with varying numbers of documents. The `get_base_module` function introspects the model to identify the transformer backbone for selective freezing during Stage 2. The training loop uses automatic mixed precision via PyTorch's `GradScaler`, gradient clipping at norm 1.0, and a linear warmup schedule over the first 10% of training steps.

Evaluation Scripts. The `bedaa_canvas_eval.py` script loads a trained model and evaluates it on the Canvas corpus, using `mc_dropout_predict` from `bedaa/utils.py` to run T forward passes with dropout active and average the softmax outputs for calibrated probability estimates; the script computes top-k accuracy for $k \in \{1, 2, 5\}$ and outputs a JSON report with per-comment predictions. The `prompt_canvas_eval.py` script implements the LIP pipeline by computing TF-IDF vectors over all external documents for each Canvas post, retrieving the top-k most similar documents by cosine similarity, serializing candidates as a JSON mapping, rendering the LIP prompt template, querying the Gemini API with structured JSON output, and parsing the model's response to extract the predicted author index.

Utility Functions. The `bedaa/utils.py` module provides inference and evaluation utilities. The `mc_dropout_predict` function runs Monte Carlo Dropout inference by performing multiple forward passes with dropout active and returning both mean probabilities and predictive entropy, while `topk_accuracy` computes whether the true label appears among the top-k predictions.

Code Sanitization. The public repository at github.com/KentoNishi/CS1050-Stylometry contains a sanitized version of the codebase with all participant data removed. A GitHub Actions workflow automatically compiles this report from the private repository, removes the `data/` directory containing raw Canvas exports and volunteer writing samples, strips the report submodule and workflow files, and pushes the result to the public repository. The trained model weights are also excluded to prevent misuse for deanonymization. The sanitized repository contains only the training and evaluation scripts, loss function implementations, and this compiled PDF.