# Source Code Authorship Attribution and Verification under Adversarial Conditions

*Project Description and Clarification*

Your Sponsor

Sponsor logo (if any)

**Your Team Name & Team Logo (pick something reasonable)**

Elang Sisson

Andrew Varkey

**Note**: Recall that this writing assignment says:

Length = 2+ pages text  plus appendices as needed.  Cover page, table of contents, glossary, references, pictures, tables, images, diagrams do not count for the 2+ pages text.

Posted as a single self-contained file (no links to outside resources.)

Posted as a PDF file.

Typed single-spaced.

Typed with black text.

Typed with #11 font size.

Typed using Arial font.

Typed with one inch margins on sides, top and bottom.

**Please erase this page in your final document.**

# I. Introduction

Stylometry, the study of linguistic style, has traditionally been applied to natural language but is being increasingly used in programming languages to identify authorship in source code. This is relevant to cybersecurity, where authorship can help in finding malicious scripts, identifying threats, and detecting plagiarism as well. In today's landscape, the rise of large language models in software development and the growing risk of attacks create significant challenges for code attribution.

Attackers can deliberately obscure or mimic stylistic traits, making it difficult to tell the difference between genuine authorship from adversarial changes which threatens how accurate traditional stylometry methods are. This project aims to explore how machine learning can be applied to verify authorship and identify when adversarial authorship occurs. By studying the impacts of these attacks and experimenting with approaches to improve robustness, our work seeks to support the development of more resilient tools for source code verification.

# II. Background and Related Work

The domain of our project lies in "source code stylometry for authorship attribution and verification with a focus on adversarial scenarios". This domain combines elements of machine learning, software security, and digital forensics. Stylometry has historically been applied to natural language, but its expansion to source code has allowed researchers to find certain authorship patterns even when multiple developers share a similar style or guideline. The research landscape can be divided into two areas: traditional code authorship attribution methods and adversarial stylometry research.

Traditional Stylometry research in source code has relied on statistical and structural features such as whitespace patterns, naming conventions, comment density, and even Abstract Syntax Trees. Work by Caliskan-Islam et al. demonstrated that programmers could be de-anonymized through these kinds of markers with over a 90% accuracy under certain conditions [1]. Other methods like contrastive learning with embeddings, have aimed to reduce the amount of training data required while maintaining strong attribution performance [2]. These works established code stylometry as a valuable tool for applications like plagiarism detection, insider threat analysis, and software provenance tracking.

Adversarial stylometry, studies how attribution can be manipulated or avoided. Certain obfuscation techniques such as variable renaming and comment removal significantly reduce model accuracy, while mimicry allows attackers to intentionally code in another developer's style. Data poisoning presents a challenge as well, as adversarial injected samples can corrupt training datasets and even mislead the attribution models. Recent approaches like RoPGen propose style transformations and adversarial training to better secure authorship models against these attacks [3].

While prior research has proven that authorship attribution is possible and adversarial attacks are effective, not much work has focused on detecting when adversarial authorship has actually occurred. We aim to build on existing attribution methods while introducing adversarial evaluation, with the goal of contributing toward stronger adversarial systems. To do so we will

build our skills in machine learning, adversarial ML, as well as developing our familiarity with Python ML libraries and visualization tools.

## III. Project Overview

Software supply chains, insider threat investigations, and academic integrity reviews are all based on knowing who wrote a piece of code. Code stylometry tackles this by measuring author specific habits embedded in source files: how identifiers are named, how control flow is shaped, preferred commenting conventions, whitespace and layout choices, and recurring syntax patterns. In practice however, these signals are fragile. Teams often follow common style guides, automated formatters erase surface cues, and an adversary can intentionally rewrite code to hide or imitate style. This project will design, implement, and stress test a code stylometry pipeline that performs both authorship attribution and authorship verification, while explicitly measuring the robustness under realistic attacks such as obfuscation, formatting wipes, identifier renaming, and style mimicry.

The main objective is to create a system that works on everyday code and remains useful when style is intentionally altered. We will separate sources so that near duplicates and templates do not leak across training and testing. This keeps the results honest and repeatable.

The approach draws on features that past work has shown to be effective. These include naming patterns, whitespace and commenting habits, and signals from syntax trees. We will build a clear baseline and then improve it only when tests show a real gain. The system will handle two tasks. The first is attribution, which selects the most likely author from a set. The second is verification, which decides if two samples are from the same author.

Robustness is part of the plan from the start. We will evaluate the pipeline after program preserving edits such as formatter runs, removal or alteration of comments and whitespace, and identifier renaming. We will also consider style mimicry where an attacker pushes a file toward another author’s look. In addition, we will study poisoning attacks in which misleading examples are injected into training data to reduce accuracy.

For attribution we want high accuracy on clean code and a limited drop after edits. For verification we want a low error rate and a stable and defensible threshold. For each edit or attack we will show how much it changes results so strengths and weaknesses are easy to see.

Intended users are cybersecurity researchers and analysts. The expected impact is a more resilient approach to authorship attribution and verification in the presence of adversarial behavior. If results are strong we may prepare a research paper on the project.

## III. Client and Stakeholder Identification and Preferences

Our primary client is PhD candidate Tashi Stirewalt from Washington State University. They prefer biweekly online meetings through Microsoft teams. The expected deliverable is a codebase for an automated tool with documentation. Python is the preferred technology and dataset will be provided.

Intended users are cybersecurity researchers and analysts. They can use the tool for authorship attribution and authorship verification on source code, with attention to adversarial conditions.

The project aims to make these analyses more resilient and may lead to a research paper submission depending on results.

Client expectations and preferences are clear. They prefer students with interests in machine learning and cybersecurity. Preferred skills include artificial intelligence, machine learning, and project management. Background on machine learning and starter code are provided and there are no data privacy, ethics, or legal requirements mentioned by the client

## IV. Glossary

**Stylometry** The statistical analysis of variations in literary style between one writer or genre and another.

## V. References


[1] A. Caliskan-Islam, R. Harang, A. Liu, A. Narayanan, C. Voss, F. Yamaguchi, and R. Greenstadt, “De-anonymizing programmers via code stylometry,” in *Proceedings of the 24th USENIX Security Symposium*, Washington, D.C., USA, Aug. 2015, pp. 255–270. Available: sec15-paper-caliskan-islam.pdf

[2] D. Álvarez-Fidalgo and F. Ortín, “Efficient source code authorship attribution using code stylometry embeddings,” in *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025)*, SCITEPRESS, 2025, pp. 342–353.

[3] Z. Li, Y. Chen, and L. Wang, “RoPGen: Towards robust code authorship attribution via automatic coding style transformation,” *arXiv preprint arXiv:2202.06043*, 2022. [Online]. Available: https://arxiv.org/abs/2202.06043