The Role of Structure in Building Adaptive Machine Learning

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Pranav Maneriker,

Graduate Program in Department of Computer Science and Engineering

The Ohio State University

2024

Dissertation Committee:

Dr. Srinivasan Parthasarathy, Advisor

Dr. Andrew Perrault

Dr. Micha Elsner

Dr. Amy Sheneman

© Copyright by
Pranav Maneriker
2024

Abstract

The success of neural networks and the advent of specialized hardware such as GPUs has led to larger models with increasingly large unstructured datasets in machine learning. Curating and assembling a large, high-quality dataset is a time-consuming process. Further, training models on these datasets requires expensive computing resources. Some of these issues are alleviated with the advent of paradigms such as self-supervised and transfer learning. However, when the data drift and change over time, models must be periodically retrained to keep up.

Graph structures, both implicit and explicit, are ubiquitous in Natural Language Processing. Implicit structures can be derived from language morphology, syntax, and semantics and expressed using attributed tree graphs. External structures capture world knowledge and semantics using knowledge graphs and ontologies. Additionally, textual data may have associated metadata in external graphs, such as network structure for social media interactions. In this dissertation, we posit that an abundance of associated structural information needs to be utilized for scaling and adaptation. The prevalence of these structures behooves us to utilize them to improvise, adapt, and overcome the challenges posed by scaling and drifts in data.

In our work, we focus on three broad directions for using these structures: augmenting existing text models with structure, exploring the role of structure in creating adversarial testing samples, and structured-enhanced monitoring of model performance over time. The

first direction that we explore is the impact of incorporating structure into text representation learning pipelines. In our first contribution, we study how the implicit structure of text data (here, URLs) can be used to design domain-specific losses and adversarial attacks to build a state-of-the-art system for phishing URL detection. This work comprehensively analyzes transformer models on the phishing URL detection task. We consider the standard masked language model and additional domain-specific pre-training tasks and compare these models to fine-tuned transformer models. Our model improves over the best baseline over a range of low false positive rates. Using a domain-informed attack scenario, we then demonstrate how these models can be more robust using adversaries constructed from benign URLs. In both fine-tuning and adversarial attacks, the underlying syntax of URLs serves as the structure that enables us to build a robust model.

Our second area of research is the role of intrinsic structure in visualizing and analyzing the fairness of machine learning models. Specifically, we study the syntax of commonly used fairness metrics. Our contribution improves the probabilistic guarantees for such grammars in an interactive and online setting. We construct a novel visualization mechanism that can be used to investigate the context of reported fairness violations and guide users toward meaningful and compliant fairness specifications. Our framework requires certain assumptions about the data-generating process at run time. Following this work, we investigate techniques that can help expand probabilistic guarantees under weaker assumptions. In particular, we are interested in a setting where dependencies between different data points are represented through a (predefined) network structure. We critically analyze the choices made and describe the trade-offs associated with existing work in this domain.

Finally, in our third broad direction, we study the problem of author identification. Our work demonstrates that it is possible to appropriately intermingle graph representation

learning with textual representations to utilize the orthogonal signals from each and improve author identification across time-disjoint task settings. We first develop a novel stylometry-based multitask learning approach for natural language and model interactions using graph embeddings to construct low-dimensional representations of short episodes of user activity for authorship attribution. We comprehensively evaluate our methods across four darkweb forums, demonstrating their efficacy over the state-of-the-art, with a lift of up to 2.5X on Mean Retrieval Rank and 2X on Recall@10. Next, we focus on the textual component of the author identification models. We demonstrate that it is possible to use models trained on large, clear web datasets to improve author identification on darkweb forums. We conclude this direction with a study of the limitations of text-based models in generalizing across time and demographics.

Our work has potential extensions for the latter two directions, which we discuss in a concluding chapter on future work. We empirically demonstrate that structure can improve author identification even with large-scale datasets. We provide concrete architectural suggestions that may be used to train models that utilize both the structure and content of large datasets in future work. Secondly, we discuss extensions of the ideas we discussed above in the work on fairness monitoring. We expand our work on our theoretical framework for conformal prediction on graphs to propose mechanisms for runtime fairness in graph-structured data. Finally, based on the observed limitations of author identification models, we propose extensions of ideas explored in our work on temporal robustness that may be used to provide bounds on the generalization capabilities of these models.

Dedicated to Meghana, Pranjali, my parents, teachers, and Sage.

Acknowledgments

The journey of completing this dissertation has been one of many ups and downs, and I am grateful to many people who have supported me along the way. My advisor, Dr. Srinivasan Parthasarathy, has given me the right impetus for taking the most important step at each stage—the next one. He has encouraged me to go beyond my comfort zone and explore new research directions. His guidance has helped me refine my ideas and thoughts through writing and presentations, which has made me more effective in communicating my research. Further, he has always encouraged me to pursue interesting external research opportunities through summer internship programs, which have helped shape my research interests and allowed me to apply them in practical settings. I am grateful for his mentorship and support throughout this process.

I would also like to acknowledge the members of my candidacy and dissertation committees, Dr. Andrew Perrault, Dr. Micha Elsner, Dr. Eric Fosler-Lussier, and Dr. Amy Sheneman, for their valuable feedback, which helped me better situate my research in the broader context of ideas in stylometry, information retrieval, and fairness in machine learning. In particular, Dr. Micha Elsner provided valuable feedback on the early drafts of my stylometry work, which provided the final nudge to help push it toward a high-quality publication. I would also like to thank my mentors before I started my Ph.D., Dr. Atanu Sinha and Dr. Subhajit Roy, for encouraging me to pursue a Ph.D. and providing me with the proper guidance to transition from a wide-eyed undergraduate student to industry and academia.

Members of the Data Mining Research Group at The Ohio State University have been frequent collaborators and confidantes through the process of my Ph.D. I have had the privilege of working with Dr. Nikhita Vedula, Dr. Bortik Bandyopadhyay, Dr. Saket Gurukar, Dr. Goonmeet Bajaj, Dr. Mark Susmann, Dr. Moniba Keymanesh, Yuntian He, Vedang Patel, Aditya Vadlamani, Sean Current, Anutam Srinivasan, Dominik Winecki, Ram Sai Ganesh, Saumya Sahai, Meghana Moorthy Bhat, Kuan-Chieh Lo, and Yue Zhang, who have been excellent collaborators and friends. I am grateful for the many discussions, brainstorming sessions, and collaborative projects that have helped me maintain steady progress in the face of challenges. Through external collaborations and internships, both inside and outside Ohio State, I have also been fortunate to work with Dr. Zhanlong Qiu, Dr. Lanfeng Pan, Dr. Byung-Doh Oh, Dr. Nanjiang Jiang, Dr. Jay Stokes, Dr. Octavian Udrea, Dr. William Groves, Dr. Nicholas Andrews, Dr. Marcus Bishop, Dr. Zachary Lubberts, Dr. Kevin Duh, AlFahad AlQadhi, Roy Siegelmann, Dr. Cristina Aggazzotti, Dr. Dana Haynie, and Dr. Scott Duxbury who have provided valuable feedback and guidance on my research.

I am also thankful to the National Science Foundation, which has funded my research through grants EAR-1520870, SES-1949037, CCF-2028944, and #2112471 (AI-EDGE). My research would not have been possible without the excellent computing resources supporting Ohio State Computer Science students, including the MRI cluster (also funded by the NSF through OAC-2018627) and the Ohio Supercomputer Center. Cisco's Responsible AI group has graciously provided funding to support my work on fairness auditing. Any opinions, findings, conclusions, or recommendations expressed in this material are mine and do not necessarily reflect the views of the National Science Foundation or Cisco.

Finally, I am grateful for having had my wife, Meghana Gupta, who has suffered my (occasional) grumpiness through deadlines and late nights that come with the territory of

being a Ph.D. student. She and our cat Sage have been a constant source of joy and support and have helped me maintain a semblance of balance in my life. I am also grateful to my parents and sister, who supported my decision to leave a comfortable job in India to pursue a Ph.D. in the United States. Their encouragement has been invaluable in helping me navigate the challenges of graduate school.

Vita

July 2012 – June 2016	B. Tech, Department of Computer Science and Engineering, Minor: English Literature, Indian Institute of Technology, Kanpur, India.
May 2015 – July 2015	Research Intern, Adobe Research, Bengaluru, India.
Aug 2015 – Nov 2015	Teaching Assistant, Data Structures and Algorithms, Indian Institute of Technology, Kanpur, India.
Jan 2016 – Apr 2016	Tutor, Introduction to Programming, Indian Institute of Technology, Kanpur, India.
June 2016 – July 2018	Research Associate, Adobe Research, Bengaluru, India.
August 2018 – present	Ph.D. student, Department of Computer Science and Engineering, The Ohio State University, Columbus, USA.
August 2018 – present	Graduate Research Associate, Department of Computer Science and Engineering, The Ohio State University, Columbus, USA.
May 2019 – August 2019	Applied Scientist Intern, Amazon, Seattle, USA.
May 2020 – August 2020	Research Intern, Microsoft Research, Redmond, USA.
May 2021 – August 2021	Research Intern, Dataminr Inc., New York, USA.

June 2022 – August 2022 ..... Visiting Research Scholar,
Johns Hopkins University HLTCOE, Baltimore, USA.

Publications

Research Publications

Pranav Maneriker, Codi Burley, Srinivasan Parthasarathy, "Online Fairness Auditing trhough Iterative Refinement", In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2023.

Pranav Maneriker, Yuntian He, Scott Duxbury, Dana Haynie, and Srinivasan Parthasarathy. "Following the trail: Tracking user styles on clear and dark web forums", In *Cambridge Cybercrime Center: Sixth Annual Conference*, 2023.

Saket Gurukar, Priyesh Vijayan, Srinivasan Parthasarathy, Balaraman Ravindran, Aakash Srinivasan, Goonmeet Bajaj, Chen Cai, Moniba Keymanesh, Saravana Kumar, **Pranav Maneriker**, Anasua Mitra, Vedang Patel, "Benchmarking and Analyzing Unsupervised Network Representation Learning and the Illusion of Progress", In *Transactions of Machine Learning Research (TMLR)*, 2022.

Pranav Maneriker, Yuntian He, Srinivasan Parthasarathy, "SYSML: StYlometry with Structure and Multitask Learning: Implications for Darknet Forum Migrant Analysis", In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Pranav Maneriker, Jack W Stokes, Edir Garcia Lazo, Diana Carutasu, Farid Tajaddodianfar, Arun Gururajan, "URLTran: Improving Phishing URL Detection Using Transformers", In *Proceedings of IEEE Military Communications Conference (MILCOM)*, 2021.

Bortik Bandyopadhyay, **Pranav Maneriker**, Vedang Patel, Saumya Yashmohini Sahai, Ping Zhang, Srinivasan Parthasarathy, "DrugDBEmbed: Semantic Queries on Relational Database using Supervised Column Encodings", In *arXiv preprint arXiv:2007.02384*, 2020.

Nikhita Vedula, Nedim Lipka, **Pranav Maneriker**, Srinivasan Parthasarathy, "Open Intent Extraction from Natural Language Interactions", In *Proceedings of The Web Conference (WWW)*, 2020.

Goonmeet Bajaj, Bortik Bandyopadhyay, Daniel Schmidt, **Pranav Maneriker**, Christopher Myers, Srinivasan Parthasarathy, "Understanding Knowledge Gaps in Visual Question

Answering: Implications for Gap Identification and Testing", In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

Ritwick Chaudhry, Sumit Shekhar, Utkarsh Gupta, **Pranav Maneriker**, Prann Bansal, Ajay Joshi, "LEAF-QA: Locate, Encode & Attend for Figure Question Answering", In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020.

Byung-Doh Oh, **Pranav Maneriker**, Nanjiang Jiang, "THOMAS: The Hegemonic OSU Morphological Analyzer using Seq2seq", In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology (SIGMORPHON)*, 2019.

Pranav Maneriker, Nikhita Vedula, Hussein S Al-Olimat, Jiayong Liang, Omar El-Khoury, Ethan Kubatko, Desheng Liu, Krishnaprasad Thirunarayan, Valerie Shalin, Amit Sheth, Srinivasan Parthasarathy, "A Pipeline for Disaster Response and Relief Coordination", In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019.

Nikhita Vedula, **Pranav Maneriker**, Srinivasan Parthasarathy, "BOLT-K: Bootstrapping Ontology Learning via Transfer of Knowledge", In *Proceedings of The World Wide Web Conference (WWW)*, 2019.

Paridhi Maheshwari, Nitish Bansal, Surya Dwivedi, Rohan Kumar, **Pranav Maneriker**, Balaji Vasan Srinivasan, "Exemplar based Experience Transfer, In *Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI)*, 2019.

Ritwik Sinha, Dhruv Singal, **Pranav Maneriker**, Kushal Chawla, Yash Shrivastava, Deepak Pai, Atanu R Sinha, "Forecasting Granular Audience Size for Online Advertising", In *Proceedings of the AdKDD*, 2019.

Balaji Vasan Srinivasan, **Pranav Maneriker**, Kundan Krishna, Natwar Modani, "Corpus-based Content Construction", In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, 2018.

Gaurush Hiranandani, **Pranav Maneriker**, Harsh Jhamtani, "Generating Appealing Brand Names", In *Proceedings of the 18th International Conference of Computational Linguistics and Intelligent Text Processing (CICLing)*, 2018.

Atanu R. Sinha, Meghanath Macha, **Pranav Maneriker**, Sopan Khosla, Avani Samdariya, Navjot Singh, "Anti-Ad Blocking Strategy: Measuring Its True Impact", In *Proceedings of the AdKDD*, 2017.

Natwar Modani, **Pranav Maneriker**, Gaurush Hiranandani, Atanu R Sinha, Utpal, Vaishnavi Subramanian, Shivani Gupta, "Summarizing Multimedia Content", In *Web Information Systems Engineering (WISE)*, 2017.

Patents

Pranav Maneriker, Codi Burley, Srinivasan Parthasarathy, "Systems and Methods for Measuring and Auditing Fairness", *US Patent App. No. 18/419,130*, 2024.

Jack Wilson Stokes III, **Pranav Maneriker**, Arunkumar Gururajan, Diana Anca Carutasu, Edir Vinicio Garcia Lazo, "Phishing URL Detection using Transformers", *US Patent App. No. 17/246,352*, 2022.

Ritwik Sinha, Virgil-Artimon Palanciuc, **Pranav Maneriker**, Manish Dash, Tharun Mohandoss, Dhruv Singal, "Accurate and interpretable rules for user segmentation", *US Patent No. 11,200,501*, 2021.

Pranav Maneriker, Reshmi Sasidharan, Atanu R. Sinha, "Facilitating changes to online computing environment by assessing impacts of temporary interventions", *US Patent No. 11,038,785*, 2021.

Pranav Maneriker, Vishwa Vinay, Sopan Khosla, Niyati Himanshu Chhaya, Natwar Modani, Cedric Huesler, Balaji Vasan Srinivasan, Anandhavelu Natarajan, "Fact Replacement and Style Consistency Tool", *US Patent No. 11,194,958*, 2021.

Atanu R Sinha, Meghanath Macha Yadagiri, **Pranav Maneriker**, Sopan Khosla, Avani Samdariya, Navjot Singh, "Techniques to quantify effectiveness of site-wide actions", *US Patent No. 11,093,957*, 2021.

Balaji Vasan Srinivasan, **Pranav Maneriker**, Natwar Modani, Kundan Krishna, "Constructing Content based on Multi-sentence Compression of Source Content", *US Patent No. 10,949,452* 2021.

Ritwik Sinha, **Pranav Maneriker**, Dhruv Singal, Atanu R Sinha, "Identifying High Value Segments in Categorical Data", *US Patent No. 10,929,438*, 2021.

Balaji Vasan Srinivasan, Shiv Kumar Saini, Kundan Krishna, Anandhavelu Natarajan, Tanya Goyal, **Pranav Maneriker**, Cedric Huesler, "Bundling Online Content Fragments for Presentation based on Content-Specific Metrics and Inter-Content Constraints", *US Patent No. 10,891,667*, 2021.

Balaji Vasan Srinivasan, Surya S Dwivedi, Rohan Kumar,**Pranav Maneriker**, Paridhi Maheshwari, Nitish Bansal, "Techniques for Generating Templates from Reference Single Page Graphic Images", *US Patent App. No. 16/376,906*, 2021.

Natwar Modani, Vaishnavi Subramanian, Shivani Gupta, **Pranav Maneriker**, Gaurush Hiranandani, Atanu Sinha, "Multimedia Document Summarization", *US Patent No. 10,762,283*, 2021.

Pranav Maneriker, Anandhavelu Natarajan, Vivek Gupta, "Predicting Style Breaches within Textual Content", *US Patent No. 10,650,094*, 2021.

Natwar Modani, Vaishnavi Subramanian, Shivani Gupta, **Pranav Maneriker**, Gaurush Hiranandani, Atanu R Sinha, "Determining quality of a summary of multimedia content", *US Patent No. 9,454,524*, 2016

Fields of Study

Major Field: Computer Science and Engineering

Studies in:

Data Mining	Dr. Srinivasan Parthasarathy
Computational Linguistics	Dr. Micha Elsner
Optimization	Dr. Abhishek Gupta

Table of Contents

Page

List of Tables

Table **Page**

List of Figures

Figure **Page**

Chapter 1: Introduction

The rise of deep learning as the primary paradigm for machine learning has been precipitated by the progress in the ability to use *unstructured* data, development of special purpose hardware (GPUs), and availability of software frameworks for rapid prototyping. In particular, *unstructured* data is expected to account for a significant fraction (nearly 80%) of the stored data across enterprises by 2025 (Rydning et al., 2018). Neural networks, composed of multiple layers, input with *raw* or *unstructured* data such as speech, images, and text form the basis of most modern deep learning architectures (LeCun et al., 2015). Distributed representations store concepts within a network as patterns across a number of processing units and are a central concept in the connectionism movement in cognitive science (Hinton et al., 1986). These distributions have motivated representation learning for unstructured data (Goodfellow et al., 2016; LeCun et al., 2015) in neural networks. Deep learning has only had mixed success with structured data - tree-based methods continue to outperform deep learning on several tasks, including classification and regression (Shwartz-Ziv and Armon, 2022; Grinsztajn et al., 2022). The successes include relation extraction, cell filling, and question-answering tasks on tabular data (Deng et al., 2022). In a similar vein, neural networks for graph representation learning have seen successes (Welling and Kipf, 2016; Veličković et al., 2018) though there remain challenges in their application for unsupervised representation learning (Gurukar et al., 2022).

The success and failures of the aforementioned approaches prompt a natural question - are there hybrid approaches that can take advantage of the successes of machine learning on *unstructured* data while simultaneously capitalizing on any available associated structure? The central theme of this dissertation is to demonstrate that structures, either those present implicitly or derived explicitly, provide opportunities for building improved models over methods that rely on unstructured data alone. Similar perspectives have been explored, for example, for natural language processing problems (Wu et al., 2021) and computer vision (Johnson et al., 2018). In particular, the study of language involves the study of structures. The main subject of this dissertation is the utility of these structures in formal and natural languages. A formal language refers to a set of strings of symbols derived from a finite alphabet and specified by a set of rules that generate them (Scott, 2000). Formal languages are grouped into increasingly large classes that can be organized into the Chomsky hierarchy (Chomsky, 1956). These classes can be characterized by the nature of the rules that are used to generate strings and the complexity of the *formal machine* that recognizes the language. Further, the study of grammar and its innateness in formal languages has also played a crucial role in the study of machine learning on natural language, though not without critics (Pullum and Scholz, 2002; Linzen and Baroni, 2021).

The preceding text references both *implicit* and *explicit* structures. We use *implicit* structures to reference those that directly influence the derivation of strings/sentences in a language. For a formal language, these are necessarily externally specified (alphabet, tokens, syntax). On the other hand, for natural languages, multiple competing formulations exist for the same phenomenon. For example, the notion of a word/lexeme is not well-defined across languages (Martin, 2017), and therefore, it can be hard to separate morphological and syntactic differences. Different formulations exist for describing syntactic structures, such as

trees having recursive phrase structures (constituency grammars) or general graphs connecting dependent words (dependency grammars). By *explicit* structure, we refer to the locally ascribed or inferred semantics and a contextual structure that helps define meaning. For instance, we study conversations on online forums and how a post can be better understood in the presence of the context preceding and surrounding it. Following this discussion of the broad range of available structure choices, we next revisit the question of their utility. While there are claims of large language models as a panacea (Bommasani et al., 2021) for solving language-related tasks, through this dissertation, we aim to demonstrate scenarios where it is necessary to take advantage of structure to achieve *adaptive* machine learning systems. For instance, representations learned for language modeling the most likely next token for a given prefix (as is common in large language models) would correspond to predicting the mode for the next token. Thus, such predictions may not be appropriate for modeling author style, where we want representations to capture the diversity of responses possible rather than model the most likely one. Stylometric modeling is one of the scenarios explored comprehensively in this dissertation.

Machine learning systems in the real world must adapt to new concepts and deal with issues arising from shifts in the underlying data distribution (Huyen, 2022). The problem of continuing to learn new tasks without forgetting knowledge from previous tasks, commonly referred to as *continual learning* or *life-long learning*, has been a subject of study since at least the 1990s (Thrun, 1998). However, in general, one cannot assume anything about how a model trained on data from the past may perform in the future or another domain. We examine these issues contextualized with respect to the three stages of the life cycle of a machine learning model, viz. pre-deployment, runtime monitoring, and post-hoc analysis. With each stage, we can associate challenges with building *adaptable* ML. For the pre-deployment stage,

we study *adversarial testing*. In the runtime monitoring stage, we study *online monitoring* and *monitoring for structured data*. Finally, in the post-deployment stage, we study *domain generalization* and *temporal robustness*.

Adversarial Testing The standard paradigm for evaluating an ML model is using train-validation-test splits to estimate the performance of a model. Typical benchmarks include standardized splits for comparing the performance of models (Gorman and Bedrick, 2019). However, having standardized splits can lead to issues in quantifying performance; measurement on a standardized, held-out split only estimates actual performance. Some of these issues can be avoided using randomized (Gorman and Bedrick, 2019) or adversarial (Søgaard et al., 2021) splits. However, alternative splits of the data cannot model all behaviors that may be encountered by a model. One mechanism of measurement of the ability of a model to capture specific required properties is by validating its behavior against well-designed tests (Ribeiro et al., 2020). Further, in the *pre-deployment* stage, these tests can serve as a tool to build more robust systems. Coming up with domain-specific test suites for generating adversarial examples efficiently remains a challenging task.

Runtime Monitoring Once a model has been deployed and begins to influence decisions in the real world, monitoring it to understand whether it achieves the expected performance is essential. The distribution of data encountered by the model at training time may differ from when it is deployed. While the outputs of a model are used to make decisions, the true labels may be unavailable and expensive to obtain. Effective monitoring of an ML system at *runtime* requires building monitoring tools that can provide estimates with fewer examples (Ginart et al., 2022). Monitoring is also imperative for auditing the compliance of decision-making ML systems with regulatory requirements. It is difficult to create efficient monitoring tools that can establish when a deployed model cannot reach regulatory or performance targets.

Online guarantees using concentration-based arguments are feasible under certain statistical assumptions about the distribution of the runtime data. In graph-structured data, the edges denote potential dependencies between nodes. This structure can provide monitoring capabilities for models used for decision-making on graphs. At the same time, given that there are weaker assumptions on the data here, the guarantees for monitoring such models would be weaker. Carefully evaluating the tradeoffs between the assumptions and the guarantees is necessary to develop monitoring tools for graph-structured data.

Domain Generalization While there is some disparity in the notion of what constitutes a *domain*, one frequently used notion of a *domain* is to reference data that are sampled from a fixed, joint distribution of inputs and outputs (Wang et al., 2022). In the *post-deployment* phase, a model built for a specific domain may be adapted for use in other analogous settings, each having its joint distribution. This motivates the problem of *domain generalization*. Representative strategies for *domain generalization* include multi-task learning (Caruana, 1997) and transfer learning (Zhuang et al., 2020). The third challenge we study in this dissertation is setting up model architectures and training algorithms and understanding the robustness of domain generalization in specific contexts.

1.1 Thesis Statement

In this dissertation, we aim to establish structure-based mechanisms usable throughout the development and deployment stages of a machine learning model that can help them adapt to changing scenarios. Specifically, we posit that implicit and explicit structures present in language aid in designing more adaptable machine-learning systems. We focus on three specific challenges in this space: adversarial testing, runtime monitoring, and domain generalization. Our contributions describe approaches utilizing core ideas to address each

challenge. Specifically, we aim to answer the following questions: *Can we use syntactic structures to train more robust models? Are there suites of behavioral tests that can be designed to test and enhance the robustness of models prior to their deployment? Can we quantify whether a model achieves or violates a fairness guarantee following its deployment while minimizing the number of samples required? Can we build analogous guarantees in the presence of graph-structured data? Are there procedures that can be used to improve a model built for a specific domain such that it can be generalized to a new domain? Can we use information from multiple related domains to enhance a model built for a specific domain? Even within a single domain, what factors affect a model's robustness over time?* In the subsequent sections, we describe our contributions that help address these questions.

1.2 Our Contributions

1.2.1 Improving the Classification of Phishing URLs using Transformers

The number of URLs and known phishing pages has continued to increase at a rapid pace. Browsers have started to include one or more machine learning classifiers as part of their security services that aim to better protect end users from harm. Browsers typically evaluate every unknown URL using some classifier in order to quickly detect these phishing pages. In this contribution, we first perform a comprehensive analysis of transformer models on the phishing URL detection task. We consider standard masked language models and additional domain-specific pre-training tasks and compare these models to fine-tuned BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) models. We use insights from the design of denoising encoders (Lewis et al., 2020; Clark et al., 2020) for text and the *structure* of URLs to design training objectives that improve the robustness of phishing URL detection models. Combining the insights from these experiments, we propose URLTran, which uses transformers to significantly improve the performance of phishing URL detection over a

wide range of very low false positive rates (FPRs) compared to other deep learning based methods. Further, we design *adversarial tests* that help quantify the robustness of models to known phishing URL attack threat models. We consider additional fine tuning with these adversarial samples and demonstrate that URLTran can maintain low FPRs under these scenarios.

1.2.2 Auditing Fairness Online through Iterative Refinement

Machine learning algorithms are increasingly being deployed for high-stakes scenarios. A sizeable proportion of currently deployed models make their decisions in a black-box manner. Under these settings, at *runtime*, monitoring the performance of machine learning models is a challenging task. This problem is further compounded when aiming to quantify the fairness of decision-making models. In this contribution, we focus on user-specified accountability of decision-making processes of black box systems. Previous work has formulated this problem as run time fairness monitoring over decision functions. However, formulating appropriate specifications for situation-appropriate fairness metrics is challenging. We follow prior work (Albarghouthi and Vinitsky, 2019; Bastani et al., 2019) in defining *structured* grammars for a broad range of fairness metrics. We then construct AVOIR, an automated inference-based optimization system that improves bounds for and generalizes prior work across these fairness metrics. AVOIR uses the grammar to distribute uncertainty across different terms within the definition of a fairness metric to achieve more efficient bounds. It also offers an interactive and iterative process for exploring fairness violations aligned with governance and regulatory requirements. Our bounds improve over previous probabilistic guarantees for such fairness grammars in *online* settings. Further, we also construct a novel visualization mechanism that can be used to investigate the context of reported fairness violations and guide users toward meaningful and compliant fairness specifications. This

visualization makes use of the parse trees for fairness metrics that are generated from the grammar used to define them.

1.2.3 Conformal Prediction for Graph Structured Data

In AVOIR, we focused on online fairness auditing in decision-making systems where data was assumed to be independent and identically distributed (IID) at run time. While it provides strong probabilistic guarantees, it is not possible to use AVOIR for monitoring models that make decisions on graph-structured data, as the graph structure imposes dependencies between the behavior of nodes. The conformal prediction framework (Vovk et al., 2005) provides a mechanism to provide statistically sound guarantees for the behavior of a model at run time under a weaker exchangeability assumption. In this contribution, we study the tradeoffs associated with different approaches aimed at using conformal prediction for graph-structured data. We contrast the dataset and data splitting setups used by existing approaches for conformal prediction on graphs. Our findings indicate that careful implementation of randomization procedures can significantly improve the efficiency of existing conformal prediction methods for graph-structured data.

1.2.4 Stylometry using Structure and Multitask Learning for Darkweb forums

Darknet market forums are frequently used to exchange illegal goods and services between parties who use encryption to conceal their identities. The Tor network is used to host these markets, which guarantees additional anonymization from IP and location tracking, making it challenging to link across malicious users using multiple accounts (sybils). Additionally, users migrate to new forums when one is closed, further increasing the difficulty of linking users across multiple forums. Recent advances in forensic linguistic (Juola, 2006) strategies allowed author identification to be used on short texts on online social media users (Shrestha

et al., 2017; Andrews and Bishop, 2019). In our contribution, we use *domain generalization* strategies to adapt these approaches to darkweb forums. We utilize the *structure* of phpBB-based bulletin-board like forums[1] prevalent on the darkweb from 2013-2017 to enhance author representations. Our *multitask learning* approach for natural language and model interactions using graph embeddings helps construct low-dimensional representations of short episodes of user activity for authorship attribution. We provide a comprehensive evaluation of our methods across four different darknet forums demonstrating their efficacy.

1.2.5 Towards Robust Author Representations

Concurrent with our work with SYSML, the literature in the field of authorship attribution using text-based approaches has seen significant performance improvements with the use of large language models (Khan et al., 2021; Rivera-Soto et al., 2021). However, the robustness of these models across domains, demographics, and time needs to be better understood. In our final contribution, we conduct a comprehensive study of the robustness of these models across time and domains. First, we study whether the representations learned by these models can be used to improve author identification on darkweb forums. Our findings indicate the limitations on the transferability of these representations to the darkweb domain. Following this, we demonstrate an approach that combines data across the domains that can be used to improve the robustness of these models. Finally, we study how models trained within a domain generalize across temporal and demographic shifts. Our careful experiments indicate that the degradation in the performance of these models arises from temporal shifts in the underlying author style rather than from model estimation errors.

[1] https://www.phpbb.com/

1.3 Future Work

1.3.1 Large Scale Structure-aware Authorship Attribution

Continuing the theme of *domain adaptation* in our work on stylometry, in this extension, we propose a mechanism to study how different domains and their associated structure correlate with the stylometric identifiability of authors. In recent work, Barlas and Stamatatos (2020); Rivera-Soto et al. (2021) conducted large-scale studies of cross-domain transfer for authorship attribution. Their results show that training on some domains leads to models that transfer better to other domains. The thesis of their work was that the diversity of topics discussed and the number of distinct authors in one domain drive better transferability. A natural question arises regarding our contribution to stylometry using structure on the darkweb relevant to the scale of these datasets. The number of users and posts in the darkweb domain is limited. For example, the total number of forum posts across multiple years on large darkweb forums only lies in the 100,000-1,000,000 range. In comparison, there are over 70 million posts on Reddit over a single month (Andrews and Bishop, 2019) in an overlapping time period. This motivates our first direction for future work–the study of structure-based methods to improve authorship attribution models when the availability of text/authors and size of models is scaled up by multiple orders of magnitude.

1.3.2 Fairness through Conformal Prediction

In Chapter 3, we address the problem of fairness auditing using the structure of the monitored metrics. We utilize the adaptive Hoeffding concentration bound (Zhao et al., 2016) to quantify the uncertainty of the tracked fairness metric, which allows monitoring with arbitrary stopping mechanisms. However, using this inequality comes with certain assumptions of the data-generating process. The most restrictive assumption is that of a

fixed fairness specification, stationary data distribution, and independent and identically distributed (IID) data. A multitude of scenarios in the real world do not conform to these assumptions. In Chapter 4, we show that under the weaker assumption of exchangeability, it is possible to use the framework of conformal risk control to provide guarantees for the outputs of graph models. In this direction of future work, we propose a mechanism for using conformal prediction sets that can guarantee fairness properties for the outputs of an existing model at runtime.

1.3.3 Towards More Robust Authorship Attribution

In chapters 5 and 6, we study the robustness of authorship attribution models across time and domains. The methods we propose in chapter 5 require retraining a model with additional context from graph-based representations. As the results in chapter 6 indicate, the representations learned by models trained on one domain do not always generalize well to other domains. Further, even within the same domain, the robustness of these models across time and demographics varies. We find degradation across time that arises from temporal shifts in the underlying author style. However, no techniques exist to modify the outputs of these models to provide guarantees about their performance. Our success with conformal prediction in the aforementioned work motivates us to use it to provide such guarantees in the context of authorship attribution. In the final direction of the proposed future work, we discuss how recalibration and conformal prediction-based approaches may achieve more robust authorship attribution.

1.4 Organization

The remainder of the dissertation is organized as follows. First, we discuss a method to improve the classification of phishing URLs using transformers in chapter 2. We describe how,

in the *pre-deployment* stage, *adversarial tests* can be constructed and used to evaluate the robustness of machine learning models for this task. Following this, we discuss a mechanism for monitoring the *runtime* fairness properties associated with *deployed* machine learning models in chapter 3. Next, in chapter 4 we analyze the tradeoffs in the conformal prediction framework and its usability in the graph setting. The next part of this dissertation focuses on the *post-deployment* stage. In chapter 5, we describe how an authorship identification model trained on one domain can be *generalized* to work across domains. Further, in chapter 6 we study the robustness of authorship identification models across time and domains. Finally, in chapter 7, we first summarize our contributions and their limitations. We then describe directions of future work that may resolve some of these limitations. The first of these involves scaling authorship identification models to work across even more domains while utilizing the graphs associated with those domains. The second describes how fairness monitoring can be made to work across settings beyond those explored in chapter 3 and chapter 4. We conclude with a discussion of how recalibration and conformal prediction can be used to provide guarantees for the robustness of authorship identification models across time and domains.

Chapter 2: Detecting Phishing URLs using Transformers

In this chapter, we present our work on phishing URL (Uniform Resource Locator) detection (Maneriker et al., 2021b) conducted at Microsoft, contextualized within the broader theme of our thesis surrounding the use of structures to build *adversarial testing* for *pre-deployment* robustness. We study the problem of detecting phishing URLs using transformer models.

Browsers often include security features to detect phishing web pages. In the past, some browsers evaluated unknown URLs for inclusion in lists of known phishing pages. However, phishing URLs and websites have a very short life span (Garera et al., 2007; Chu et al., 2013). Therefore, models must be able to *adapt* to rapidly changing data distribution. As the number of URLs and known phishing pages has continued to increase rapidly, browsers have started to include one or more machine learning classifiers in their security services, which aim to better protect end users from harm.

Recent research has proposed using deep learning models for the phishing URL detection task (Sahoo et al., 2017; Yerima and Alzaylaee, 2020; Ren et al., 2019; Peng et al., 2019; Huang et al., 2019; Tajaddodianfar et al., 2020). Concurrently, text embedding research using transformers has led to state-of-the-art results in many natural language processing tasks. In this contribution, we first comprehensively analyze transformer models on the phishing URL detection task. We consider both pre-trained and end-to-end transformer models,

with standard masked language models and additional domain-specific pre-training tasks. We compare end-to-end training compare against fine-tuned BERT and RoBERTa models. Misclassifying a benign URL as malicious can be damaging for a phishing URL classification model.Therefore, phishing URL detection models are compared by measuring true positive rates at very low false positive rates. The insights our experiments help us propose URLTran, which uses transformers to significantly improve the performance of phishing URL detection over a wide range of very low false positive rates (FPRs) compared to other deep learning-based methods. For example, URLTran yields a true positive rate (TPR) of 86.80% compared to 71.20% for the next best baseline at an FPR of 0.01%, resulting in a relative improvement of over 21.9%. We use insights from the structure of URL grammar (Berners-Lee et al., 2005).

As mentioned previously, phishing URL attacks are carried out through short-lived and changing URL patterns. Therefore, the machine learning models must be retrained and redeployed at regular intervals. This procedure may lead to a *catastrophic forgetting* (McCloskey and Cohen, 1989) phenomenon, whereby models forget the old patterns and only adapt to new ones. In our second contribution to this work, we propose a threat model and construct an *adversarial testing* scenario to validate models against known threat patterns before deployment. We consider some classical adversarial black-box phishing attacks, such as those based on homoglyphs and compound word splits, to improve the robustness of URLTran. Inspired by the behavioral testing paradigm (Ribeiro et al., 2020), we provide algorithms to efficiently construct datasets that can help quantify the capabilities of trained models against known threat patterns.

2.1 Introduction

Phishing occurs when a malicious web page is created to mimic the legitimate login page used to access a popular online service for the purpose of harvesting the user's credentials or a web page whose purpose is to input credit card or other payment information. Typical phishing targets include online banking services, web-based email portals, and social media websites. Attackers use several methods to direct the victim to the phishing site in order to launch the attack. In some cases, they may send the user a phishing email containing the URL of a phishing page. Attackers may also use search engine optimization techniques to rank phishing pages high in a search result query. Modern email platforms use various machine learning models to detect phishing web page attacks. In this work, we propose a new deep learning model that analyzes URLs and is based on transformers which have shown state-of-the-art performance in many important natural language processing tasks.

In order to prevent users from inadvertently uploading personal information to the attackers, web browsers provide additional security services to identify and block or warn a user from visiting a known phishing page. For example, Google's Chrome browser utilizes their Safe Browsing technology (Google, 2007) and Microsoft's Edge browser includes Windows Defender SmartScreen (Microsoft, 2023). In a related attack which is also addressed by these services, malicious URLs may point to a web page hosted by a misconfigured or unpatched server with the goal of exploiting browser vulnerabilities in order to infect the user's computer with malware (i.e., malicious software). Successful phishing web page detection includes a number of significant challenges. First, there is a huge class imbalance associated with this problem. The number of phishing pages on the internet is very small compared to the total number of web pages available to users. Second, phishing campaigns are often short-lived. In order to avoid detection, attackers may move the login page from one site to another

multiple times per day. Third, phishing attacks continue to be a persistent problem. The number of known phishing sites continues to increase over time. Therefore, blocking phishing attacks only using a continuously growing list of known phishing sites often fails to protect users in practice.

Popular web browsers may render hundreds of millions or even billions of web pages each day. In order to be effective, any phishing or malicious web page detection must be fast. For this reason, several researchers (Blum et al., 2010; Le et al., 2018; Tajaddodianfar et al., 2020) have proposed detecting both phishing and malicious web pages based solely on analyzing the URL itself. With the proliferation and ease of access to phishing kits sold on the black market as well as the phishing as a service offering, it has become easy for attackers with little expertise to deploy phishing sites and initiate such attacks. Consequently, phishing is currently on the rise and costing over $57 million from more than 114,000 victims in the US according to a recent FBI report (2019). The number of phishing attacks rose in Q3 of 2019 to a high level not seen since late 2016 (HelpNetSecurity, 2019). As phishing is proving to be more and more fruitful, the attacks have become increasingly sophisticated. At the same time, the lifespan of phishing URLs has continued to drop dramatically – from 10+ hours to minutes (Zvelo, 2020).

Given the significant repercussions of visiting a phishing or malicious web page, the detection of these URLs has been an active area of research (Sahoo et al., 2017). Researchers have proposed the use of extracted feature-based natural language processing methods to detect malicious URLs (Blum et al., 2010). Recent efforts have also begun to use deep learning models to detect these URLs (Le et al., 2018; Tajaddodianfar et al., 2020). Concurrently, semi-supervised machine learning methods have been used to create text embeddings that offer state-of-the-art results in many natural language processing tasks. The key idea in these

approaches is the inclusion of a transformer model (Vaswani et al., 2017). BERT (Devlin et al., 2019; Rogers et al., 2020) utilizes transformers to offer significant improvements in several natural language processing (NLP) tasks. The GPT (Ratdford et al., 2018; Radford et al., 2019; Brown et al., 2020) series of models have also followed a similar approach. The semantics and syntax of natural language are more complex than URLs, which must follow a syntax specification (Berners-Lee et al., 2005) in the finite-state automaton/regex level of the Chomsky hierarchy (Chomsky, 1956). However, recent work using transformers has also demonstrated that these models can be applied to tasks involving data with more strict syntactic structures. These include tabular data (Yin et al., 2020), python source code (Kanade et al., 2020) and SQL queries (Wang et al., 2020a). The success of these approaches further motivates us to apply transformers on URLs.

In this paper, we compare two settings: 1) we pre-train and fine-tune an existing transformer architecture using only URL data, and 2) we fine-tune publicly available pre-trained transformer models. In the first approach, we apply the commonly used Cloze-style masked language modeling objective (Taylor, 1953) on the BERT architecture. In the second approach, we fine-tune BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) on the URL classification task. Each of these systems forms an example of a URLTran model. URLTran$_B$ is the best performing model obtained from these approaches. Finally, we simulate two common black-box phishing attacks by perturbing URLs in our data using unicode-based homoglyph substitutions (Woodbridge et al., 2018) and inserting '-' characters between sub-words in a compound URL (e.g., 'bankofamerica.com' $\rightarrow$ 'bank-of-america.com'), along with a perturbation scenario under which the URL parameters are reordered, and the URL label remains unchanged to improve the robustness of URLTran.

Results on a large corpus of phishing and benign URLs show that transformers are able to significantly outperform recent state-of-the-art phishing URL detection models (URLNet, Texception) over a wide range of low false positive rates where such a phishing URL detector must operate. At a false positive rate of 0.01%, URLTran increases the true positive rate from 71.20% for the next best baseline (URLNet) to 86.80% (21.9% relative increase). Thus, browser safety services, such Google's Safe Browsing and Microsoft's SmartScreen, may potentially benefit using the proposed URLTran system for the detection of phishing web pages. Further, we use the *implicit* structure of URLs and common threat models to devise an *adversarial testing* setup that facilitate development of more robust models.

We summarize the contributions of our work. First, borrowing from recent advances in many natural language processing tasks, we propose the use of transformers to improve the detection of phishing URLs. Second, We build URLTran, a large-scale system with production data and labels and demonstrate that transformers do offer a significant performance improvement compared to previous recent deep learning solutions over a wide range of very low false positive rates. Third, we analyze the impact of various design choices in terms of hyperparameters, pre-training tasks, and tokenizers to contribute to an improved model. Finally, we analyze the adversarially generated URLs from the system to understand the limitations of URLTran, forming a benchmark that can also be used to evaluate the *adaptation* of phishing URL detection models.

2.2 Related Work

The URLTran system is most closely related to phishing and malicious URL detection models which have been previously proposed in the literature. In this section, we first provide a short summary of the related work for deep learning-based text embeddings in general.

Following this, we describe some examples of adversarial attack models commonly used for testing the robustness of text embedding models. We then review related work in phishing and malicious web page detection using a webpage URL which builds upon the previous text embedding models proposed in the NLP domain. In particular, we focus on two recent, deep learning URL detection models, URLNet and Texception, which helped to inspire this work.

Text Embeddings. Deep learning models for text embeddings have been an active area of research recently. One family of models - character-level CNNs[2] learn a text embedding from individual characters, and these embeddings are then processed using a sequential CNN and one or more dense layers depending on the task. Recent examples of character-level CNNs include work by Conneau et al. (2017); Zhang et al. (2015). In particular, Conneau et al. (2017) investigated very deep architectures for the purpose of classifying natural language text. Typically, these models are trained in an end-to-end fashion instead of from manually engineered features. Different formulations of recurrent Neural Networks for machine translation have also been widely used (Graves, 2013; Hochreiter and Schmidhuber, 1997; Cho et al., 2014; Bahdanau et al., 2015) for producing text embedding for text processing tasks. Transformers were introduced by Vaswani et al. (2017) in the context of neural machine translation. A number of models use variations of the original transformer architecture for other natural language processing tasks including BERT (Devlin et al., 2019; Rogers et al., 2020).RoBERTa (Liu et al., 2019) used careful optimization of the BERT parameters and training methodology to offer further improvements. Transformer-based models have been adopted for a wide number of tasks (Bommasani et al., 2021) beyond natural language processing.

[2]Convolutional Neural Networks

Adversarial Attacks on Text. Adversarial example generation has been a focus of some recent work on understanding the robustness of various text classification tasks. The examples generated using these approaches aim to impose certain semantic constraints without modifying the label of the underlying text. White-box attacks (e.g., Hotflip (Ebrahimi et al., 2018a)) require access to the internals of the classification model used, such as the gradient on specific examples. The attack framework proposed in our work is more in line with black-box attack frameworks such as DeepWordBug (Gao et al., 2018) and TextAttack (Morris et al., 2020) where the construction of adversarial data is motivated by a threat model but independent of the classifier used. Validating behavior against well-designed tests is an important mechanism to measure whether language models capture specific linguistic properties (Ribeiro et al., 2020). We specialize these schemes for the URL context.

URL-Based Phishing and Malicious Web Page Detection. Previous related work on the detection of phishing and malicious web pages based on their URL has progressed in parallel. We next review some important systems in chronological order.

Early phishing page detection based on URLs followed conventional deep learning approaches. A summary of these methods is provided by Sahoo et al. (2017). Blum et al. (2010) proposed using confidence-weighted online learning on a set of lexical features extracted from the URL. To extract these features, the URL is first split using the following delimiters: '?', '=', '/', '.', and ' '. Next, individual features are set based on the path, domain, and protocol. Le et al. (2018) proposed the URLNet model to detect URLs which are references to malicious web pages. URLNet processes a URL using a character-level CNN and a word-level CNN. Inspired by the Xception deep object recognition model for images, Texception (Tajaddodianfar et al., 2020) also uses separate character-level and word-level CNNs like URLNet. However, the CNN kernels in Texception form different sized-text windows for both the

character and word levels. In addition, Texception utilizes contextual word embeddings in the form of either FastText (Joulin et al., 2017) or Word2Vec (Mikolov et al., 2013b) to convert the URL into the input embedding vector. Another CNN-based phishing detection model was proposed by Yerima and Alzaylaee (2020), who create a 31-dimensional feature vector using the contents of web pages and train a CNN based on this feature vector. will be much slower for inference. Other work has proposed using LSTMs (i.e., recurrent sequential models) for phishing and malicious URL detection including Ren et al. (2019); Peng et al. (2019). Processing LSTMs is expensive in terms of computation and memory for long URLs which makes them impractical for large-scale production. Huang et al. (2019) also investigate using capsule networks for detecting phishing URLs.

2.3 Dataset Description

The datasets used for training, validation and testing were collected from Microsoft's Edge and Internet Explorer production browsing telemetry during the summer of 2019. The schema for all three datasets is similar and consists of the browsing URL and a boolean determination of whether the URL has been identified as phishing or benign. Six weeks of historical data were collected overall out of which four weeks of data were used for the training set, one week for the validation and one week for the test set. Due to the highly unbalanced nature of the datasets (roughly 1 in 50 thousand URLs is a phishing URL), we down-sampled the benign set and the resultant dataset consisted of a 1:20 ratio (phishing versus benign) for both the training and validation sets. The corresponding total sizes were 1,039,413 records for training and 259,854 thousand for validation, respectively. The test set used for evaluating the models consists of 1,784,155 records, of which 8,742 are phishing URLs and the remaining 1,775,413 are benign.

The labels included in this dataset correspond to those used to train production classifiers for Microsoft Smartscreen (Microsoft, 2023). Phishing URLs are manually confirmed by analysts including those which have been reported as suspicious by end user feedback. Other manually confirmed URLs are also labeled as phishing when they are included and manually verified in known phishing URL lists including Phishtank. [3] Benign URLs correspond to web pages which are known to not be involved with a phishing attack. In this case, these sites have been manually verified by manual analysis. In some cases, benign URLs can be confirmed by thorough (i.e., production grade) off-line automated analysis which is not an option for real-time detection required by the browser. None of the benign URLs have been included in known phishing lists or have been reported as phishing pages by users and later verified by analysts. It is important to note that all URLs labeled as benign correspond to web pages that have been validated. They are not simply a collection of unknown URLs, i.e., ones which have not been previously detected as phishing sites.

2.4 Methodology

URLTran seeks to use recent advances in natural language processing to improve the task of detecting phishing URLs. Building URLTran employs a two-pronged approach towards adapting transformers for the task of phishing URL detection. First, state-of-the-art transformer models, BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), are fine-tuned, starting from publicly available vocabularies and weights and across different hyperparameter settings and resulting in URLTran$_B$ and URLTran$_R$, respectively. Second, domain-specific vocabularies are built using different tokenization approaches, and a domain specific transformer (URLTran$_C$) is first pre-trained and then fine-tuned on the task.

[3]At the time of this study, the total of $73,705$ valid phishing URLs was significantly larger than the number of phishing URLs reported by competitors such as Phishtank (`http://phishtank.org/stats.php`).

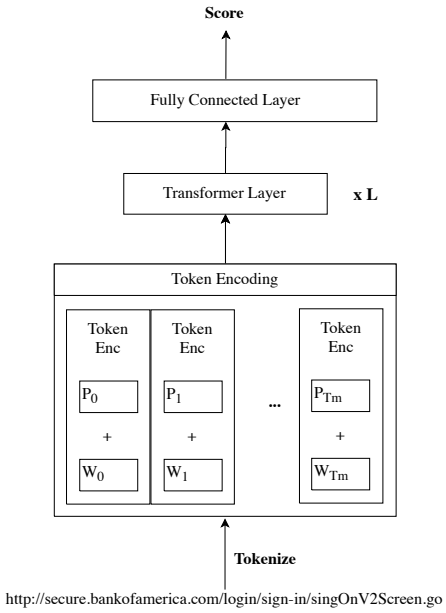


Figure 2.1: URLTran phishing URL detection model.

The general architecture of all the explored models takes a three stage approach for inference shown in Figure 2.1. It first uses a subword tokenizer to extract tokens from a URL. Next, a transformer model generates an embedding vector for the unknown URL. Finally, a classifier predicts a score indicating whether or not the unknown URL corresponds to a phishing web page. In the following sections, we first provide briefly summarize the transformer model architecture, followed by the training tasks used to train the model, and end with a description of the adversarial settings under which the best URLTran model is evaluated and then trained with adversarial examples to improve its robustness.

2.4.1 Architecture

We describe the tokenization schemes and overall architecture for classification in this section, skipping a detailed description of transformer models for brevity. Interested readers can

review the transformer (Vaswani et al., 2017), BERT (Devlin et al., 2019), or RoBERTa (Liu et al., 2019) papers for details of the internal structure of transformer layers.

2.4.1.1 Tokenization

The raw input to the URLTran model is the URL, which can be viewed as a text sequence. The first step in the phishing URL detection task involves converting this input URL into a numerical vector which can be further processed by a classic machine learning or deep learning model. Previous URL detection models (Blum et al., 2010) extracted lexical features by first splitting the URL with a set of important delimiters (e.g., '=', '/', '?', '.', ' ') and then creating a sparse binary features based on these tokens. Recent deep learning-based URL detection models (Le et al., 2018; Tajaddodianfar et al., 2020) instead include separate word-level and character-level CNNs where the character-level CNNs span different lengths of character subsequences.

Instead of these approaches, we experiment with multiple subword tokenization schemes in URLTran. Subword models have seen increased adoption in different tasks in NLP, including machine translation (Sennrich et al., 2016), word analogy (Bojanowski et al., 2017), and question answering (Zhang et al., 2019b). Using subword models helps balance between the tradeoffs of using full-length words for each token (leading to fewer tokens required per input but a large token vocabulary) and character-based models (more tokens required per input but a smaller token vocabulary). For example, a full-length model would consider 'bankofamerica' and 'bankofcanada' as completely unrelated tokens, whereas a subword model can recognize the shared subword 'bank' to correlate URLs belonging to the two banks. Frequently occurring character substrings tend to correspond to subwords; common prefixes and suffixes can also provide relevant information. In particular, for fine-tuned URLTran$_B$ and URLTran$_R$, we use the corresponding existing word piece (Wu et al., 2016;

URL (u_m)	secure.bankofamerica.com/login/sign-in/signOnV2Screen.go
Token Sequence ($\mathbf{TOK_m}$)	{ 'secure', '.', 'bank', '##of', '##ame', '##rica', '.', 'com', '/', 'log', '##in', '/', 'sign', '-', 'in', '/', 'sign', '##on', '##v', '##2', '##screen', '.', 'go' }

Table 2.1: Example of the wordpiece token sequence extraction from a popular banking web page.

Devlin et al., 2019) and Byte Pair Encoding (BPE) models (Radford et al., 2019; Liu et al., 2019), respectively. In addition to these, we create custom character-level and byte-level BPE vocabularies using the training URL data to have a domain specific vocabulary for URLTran$_C$. We test two different vocabulary sizes, 1K and 10K. We tested both sentence piece and BPE tokenization schemes in our models.

The BPE models first break the m^{th} URL, u_m, into a sequence of text tokens, $\mathbf{TOK}_m$, where the individual tokens may represent entire words or subwords (Schuster and Nakajima, 2012; Sennrich et al., 2016; Wu et al., 2016). Following the notation in (Devlin et al., 2019), the token sequence is formed as:

$$\mathbf{TOK}_m = \text{Tokenizer}(u_m) \tag{2.1}$$

where $\mathbf{TOK}_m$ is of length T_m positions and consists of individual tokens Tok_m^t at each position index t. For example, the BERT wordpiece token sequence generated from the URL of a popular banking login page,

u_m = secure.bankofamerica.com/login/sign-in/signOnV2Screen.go

is shown in Table 2.1. The wordpiece model includes special text tokens specified by (##) which build upon the previous token in the sequence. In the example in Table 2.1, '##of' refers to the continuation from s token ('bank'), and helps distinguish from the more common separate token 'of'.

2.4.1.2 Classifier

The final encoding produced by the transformer model can be used for a variety of downstream NLP tasks such as language understanding, language inference, and question answering, and text classification. We use the transformer embeddings for two tasks: pre-training masked language models, and fine-tuning for classification of phishing URLs. Both of these tasks require a final representation layer which can be applied to multiple tokens for masked token prediction, and a pooled representation for classification. The transformer models that we train use a single, dense two-class classification layer, which is applied to a special pooled token ('`[CLS]`') for classification. A dense layer having an output size equal to the size of the vocabulary of the tokenizer (`vocab_size`) classes is used for predicting the masked token for the masked language modeling task during pre-training:

$$s_m = \sigma(\mathbf{W}_p \mathbf{x}_m^0 + \mathbf{b}_p) \tag{2.2}$$

$$\mathbf{s}_m^t = \boldsymbol{\sigma}(\mathbf{W}_v \mathbf{x}_m^t + \mathbf{b}_v) \tag{2.3}$$

In (2.2), $\mathbf{W}_p$ and $\mathbf{b}$ are the weight matrix, bias vector respectively, for the final dense linear layer for predicting the phising label. σ is the softmax function and s_m is the score which predicts if the URL $\mathbf{u}_m$ corresponds to a phishing web page when performing classification. Similarly, in (2.3), $\mathbf{s}_m^t, t \in [n]$ are the sequence of masked token probability score vectors when performing masked language modeling for input token $\mathbf{x}_m^t$ computed using the softmax function $\boldsymbol{\sigma}$, with weight matrix $\mathbf{W}_v$ and bias vector $\mathbf{b}_v$. We now describe how input tokens are modified for masked language modeling and fine tuning.

2.4.2 Training

2.4.2.1 Masked Language Modeling (MLM)

The MLM task is commonly used to perform pre-training for transformers (Devlin et al., 2019; Liu et al., 2019). In this task, a random subset of tokens is replaced by a special '`[MASK]`' token. The training objective for the task is the cross-entropy loss corresponding to predicting the correct tokens at masked positions. The intuition for using this task for URLs is that specific query parameters and paths are generally associated with non-phishing URLs and therefore predicting masked tokens would help to uncover these associations. Similar intuitions derived from the cloze task (Taylor, 1953) motivate the usage of MLMs for pre-training natural language models. Following the MLM hyperparameter settings for BERT, we selected 15% of the tokens sampled uniformly for masking, of which 80% are replaced, 10% were left unchanged, and 10% were replaced by a random vocabulary token at each iteration. We used dynamic masking (Liu et al., 2019), i.e., different tokens masked from the same sequence across iterations. Only the training subset of the full dataset was used for pre-training to prevent any data leakage.

2.4.2.2 Fine Tuning

For URLTran$_B$ and URLTran$_R$, we initialized all the parameters using the pretrained weights provided for BERT by Devlin et al. (2019) and RoBERTa by Liu et al. (2019) respectively. For URLTran$_C$, we first pretrain each model using the MLM pretraining task and use the learned weights as initialization values. Next, each URLTran model, is trained using a second "fine-tuning" task using the error backpropagated from the URL classification task. We used the Adam (Kingma and Ba, 2014) optimizer with cross-entropy losses to train each model.

2.4.3 Adversarial Attacks and Data Augmentation

Threat Model The approach we use for generating data for an adversarial attack includes generating separate augmented training, validation and test datasets based on their original dataset.For each URL processed in these datasets, we generate a random number. If it is less than 0.5, we augment the URL, or otherwise, we include it in its original form. For URLs which are to be augmented, we modify it using either a homoglyph attack, a compound attack or parameter reordering with equal probability. If a URL has been augmented, we also include the original URL in the augmented dataset.

The threat model for URLTran allows for the attacker to create any phishing URL including those which employ domain squatting techniques. In its current form, URLTran is protected against homoglyph and compound word attacks through dataset augmentation. However, any domain squatting attacks can also be simulated and included in the augmented adversarial training, validation, and test sets. In addition, a larger number of adversarial training examples can be directed at more popular domains such as https://www.bankofamerica.com that may be a target of attackers. We assume that inference can be executed by the countermeasure system prior to the user visiting the unknown page. This can be done by the email system at scale by evaluating multiple URLs in parallel. In our evaluation, we found that URLTran requires 0.36096 milliseconds per URL on average which is a reasonable amount of latency.

Adversarial Testing Data augmentation using invariants, contextual replacement, and reward-based learning has been used to improve classifiers in the text domain (Kobayashi, 2018; Hu et al., 2019). These can be extended to augment data in the URL domain. Phishing URL attacks can occur on short-lived domains and URLs which have small differences from existing, legitimate domains. We simulate two attack scenarios by constructing examples of

such adversaries based on modifying benign URLs. Note that these generated domains do not actually exist in the pre-existing training and testing data, but are based upon frequently observed phishing attack patterns. We also utilized a reordering-based augmentation, which is used is used to generate benign perturbations for evaluating robustness of adversarially trained models.

2.4.3.1 Homoglyph Attack

We generated domains that appear nearly identical to legitimate URLs by substituting characters with other unicode characters that are similar in appearance. This attack strategy is commonly referred to as a *homoglyph attack* (Gao et al., 2018; Yerima and Alzaylaee, 2020), and we implement this strategy using the `homoglyphs`[4] python library. In particular, given a URL, we first extract the domain. For a randomly selected character in the domain, we check for one unicode (utf-8) Latin or Cyrillic character that is a homoglyph for it. For each perturbed URL, we selected a random character to perturb and generated the associated URL, labeled as a phishing URL. We replaced the character by its homoglyph to construct a new URL.

2.4.3.2 Compound Attack

An alternative way to construct new phishing URLs is by splitting domains into sub-words (restricted to English) and then concatenating the sub-words with an intermediate hyphen. For example, 'bankofamerica.com' → 'bank-of-america.com'. To implement this, we leveraged a popular dictionary used by multiple spell check programs, the `enchant` dictionary[5]. Consider a URL with domain d having $|d| = n$ characters. Let $\mathcal{D}$ denote the `enchant` English dictionary. Let $C(d, i, j)$ denote the function that returns True if $d[i \dots j]$

[4]https://pypi.org/project/homoglyphs/

[5]https://pypi.org/project/pyenchant/

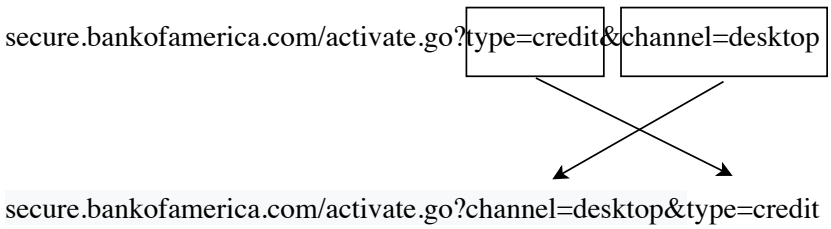


Figure 2.2: An example of parameter reordering

can be split into one or more parts, each of which is a word in the dictionary $\mathcal{D}$. The compound word problem can be formulated recursively as

$$C(d,i,j) = \begin{cases} \text{True}, & d[i \dots j] \in \mathcal{D} \\ \text{True} & \exists k, C(d,i,k) \text{ and } C(d,k+1,j) \\ \text{False} & \text{otherwise} \end{cases} \tag{2.4}$$

Using this recursive definition, we implemented a dynamic programming algorithm that can compute whether a domain can be split and the corresponding splits. These splits are then concatenated with hyphens between the discovered words. Note that the base case check $d[i \dots j] \in \mathcal{D}$ is performed in a case insensitive manner to ensure that the dictionary checks do not miss proper nouns.

2.4.3.3 Parameter Reordering

Forms of permutation=based denoising have shown improvements for langauge modeling (Lewis et al., 2020) and machine translation (Lample et al., 2018). We adapt these intuitions into the phishing URL domain. As the query parameters of a URL are interpreted as a key-value dictionary, this augmentation incorporates permutation invariance. An example of a URL and permutation is provided in Figure 2.2. We use this approach to generate benign examples. Reordering the parameters still results in a valid URL, i.e., parameter reordering does not represent a phishing attack, and therefore we do not modify the label.

2.5 Evaluation

We now present the numerical evaluation of the different approaches presented in the previous sections. Thereafter, we compare URLTran to several recently proposed baselines. We also report the model's training and inference times. Finally, we analyze the robustness of the model *adversarial test* URLs.

Setup: In our experiments, we set the hyperparameters for previously published models according to their settings in the original paper. For evaluating URLTran$_C$, we vary the number of layers between $\{3, 6, 12\}$, vary the number of tokens per input URL sequence between $\{128, 256\}$. We use both a byte-level and character-level BPE tokenizer with 1K- and 10K-sized vocabularies. We randomly pick 15 hyperparameter combinations among these settings and present the results for these. The Adam optimizer (Kingma and Ba, 2014) is used in both pre-training and fine-tuning, with the triangular scheduler (Smith, 2017) used for fine-tuning. The hyperparameter settings for all models are provided in Section 2.6. All training and inference experiments were conducted using PyTorch (Meta, 2016) version 1.2 with NVIDIA Cuda 10.0 and Python 3.6. The experiments were performed by extending the HuggingFace and Fairseq PyTorch implementations found on GitHub (HuggingFace, 2019; Ott et al., 2019). Given the large class imbalance, accuracy is a poor metric of model performance. To supplement accuracy, we evaluated all the models using the true positive rate (TPR) at low false positive rate (FPR) thresholds. We used the receiver operating characteristics (ROC) curve to compute this metric.

Baselines: To evaluate the performance of our models, we compared them to two baseline phishing URL detection models: URLNet and Texception. URLNet (Le et al., 2018) is a CNN-based model which was recently proposed for the task of detecting URLs which identify malicious web sites. For comparison purposes, we trained and tested the URLNet model

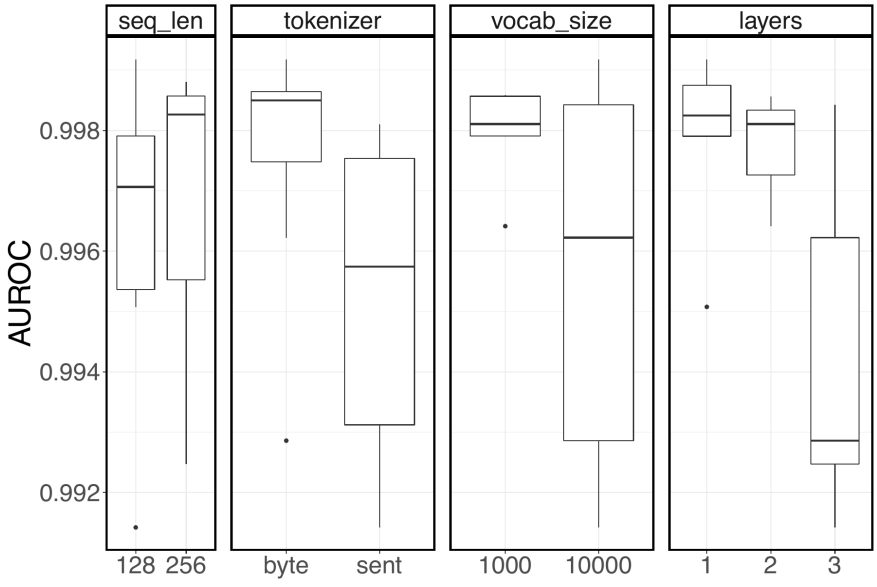
for the detection of phishing URLs. Texception (Tajaddodianfar et al., 2020) is another deep learning URL detection model for the task of identifying phishing URLs. Note that Tajaddodianfar et al. (2020) compared Texception to a Logistic Regression-based model and found that Texception offered better performance. Thus, we did not repeat that baseline experiment in this work.

2.5.1 End-to-end Training

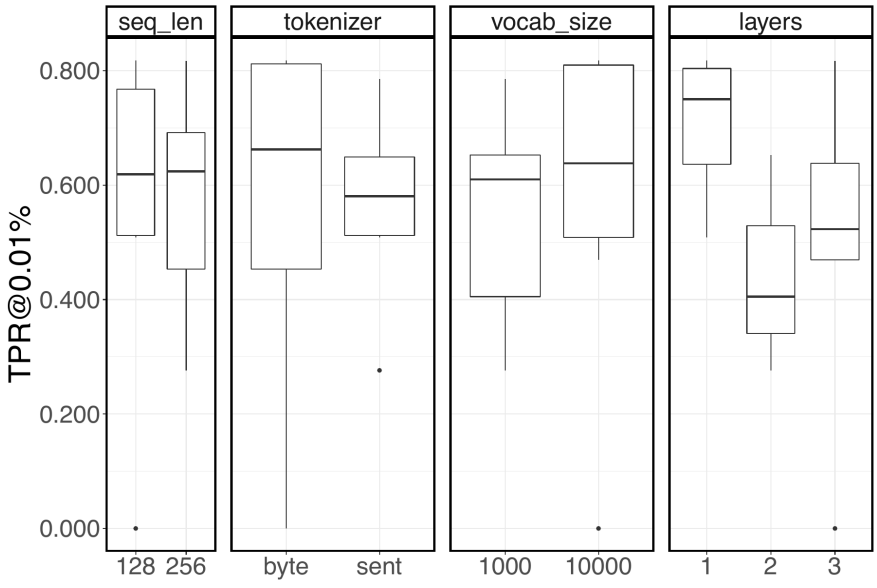
Transformers typically require large amounts of pre-training data (e.g., BERT (Devlin et al., 2019) used a corpus of $\approx$ 3.3 B tokens). However, this data is derived from text articles, which are structured differently from URLs. We trained the URLTran$_C$ model based solely on the URL data found in our datasets to compare the results of finetuning using BERT (URLTran$_B$) and RoBERTa (URLTran$_C$) pretrained models to models pretrained only on in-domain URL data. The difference in dataset size and data domain make it important to understand the impact of different hyperparameters used when training transformers from scratch. We compared runs across different hyperparameters on the basis of area under ROC (AUROC) and TPR@0.01% FPR. Figure 2.3 demonstrates that the training is not very sensitive to sequence length. Smaller byte-level vocabularies tend to be better overall, but at low FPR, the difference is not significant. Finally, we found that the 3 layer model generalized the best. We hypothesize that the better performance of the model with fewer layers is because of limited pre-training data.In the next few sections, we validate this hypothesis by evaluating fine-tuned model (URLTran$_B$, URLTran$_R$) that are tuned on a larger dataset.

2.5.2 Numerical Evaluation

Model Performance. We next analyzed the performance of the best parameters of all the proposed transformer variants. To understand how these models compare at *very low FPRs*



(a) Area under ROC vs hyperparameters



(b) TPR@FPR = 0.01% vs hyperparmeters

Figure 2.3: Variance in quality of URLTran$_C$ across different hyperparameter settings

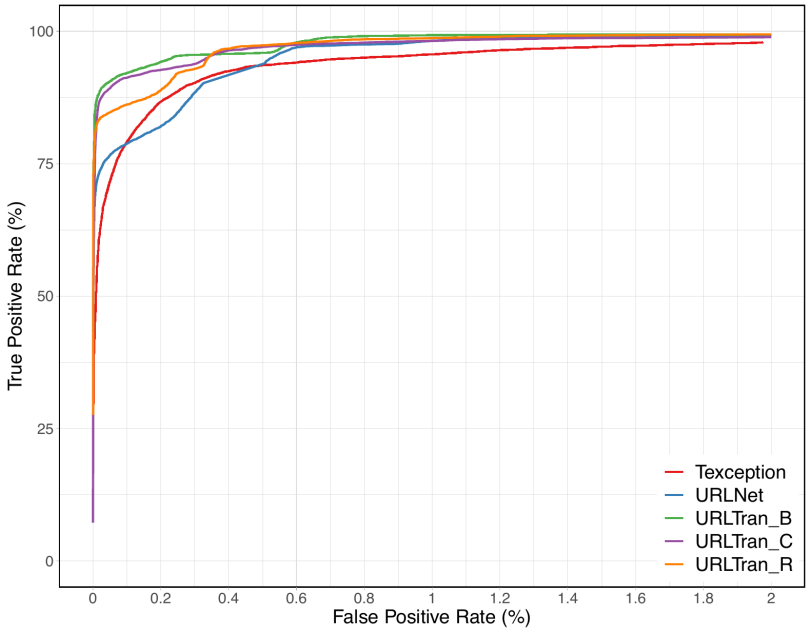


Figure 2.4: Receiver operating characteristic curve indicating the performance of the URLTran and several baseline models zoomed into a maximum of 2% false positive rate.

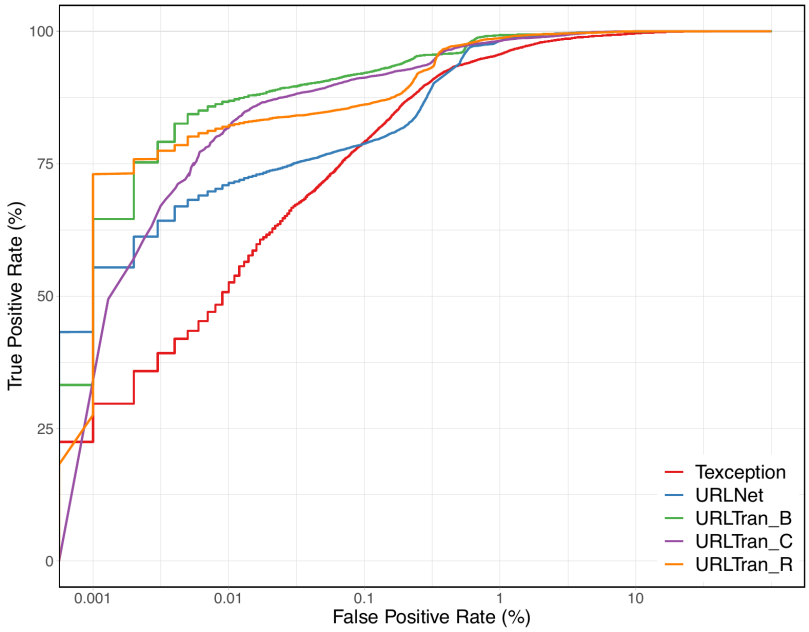


Figure 2.5: Zoomed in receiver operating characteristic curve with a log x-axis.

where detection thresholds must be set to operate in a production environment, we first plotted the ROC curves on a linear x-axis zoomed into a 2% maximum FPR in Figure 2.4. We also re-plot these ROC curves on a log x-axis in the semilog plot in Figure 2.5. These results indicate that all variants of URLTran offer a significantly better true positive rate over a wide range of extremely low FPRs. In particular, URLTran matches or exceeds the TPR of URLNet for the FPR range of 0.001% - 0.75%. The result is very important because phishing URL detection models must operate at very low FPRs (e.g., 0.01%) in order to minimize the number of times the security service predicts that a benign URL is a phishing site (i.e., a false positive). In practice, the browser manufacturer selects the desired FPR and tries to develop new models which can increase the TPR for the selected FPR value. Note that TPR@FPR is the standard metric commonly used both in production settings and in prior art such as Texception and URLNet. In addition to the ROC curve analysis, we also summarize a number of key performance metrics in Table 2.2, where 'F1' is the F1 score, and 'AUC' is the area under the model's ROC curve. The proposed URLTran model outperforms both Texception and URLNet for all of these metrics. In particular, we note that at an FPR of 0.01%, URLTran$_B$ has a TPR of 86.80% compared to 71.20% for URLNet and 52.15% for Texception.

Model	Accuracy (%)	Precision (%)	Recall (%)	TPR@FPR=0.01%	F1	AUC
Texception	99.6594	99.7562	99.6594	52.1505	0.9969	0.9977
URLNet	99.4512	99.7157	99.4512	71.1965	0.9954	0.9988
URLTran$_C$	99.5983	99.7615	99.5983	81.8577	0.9965	0.9992
URLTran$_R$	99.6384	99.7688	99.6384	82.0636	0.9968	0.9992
URLTran$_B$	99.6721	99.7845	99.6721	86.7994	0.9971	0.9993

Table 2.2: Comparison of different performance metrics for URLTran and the two baseline models.

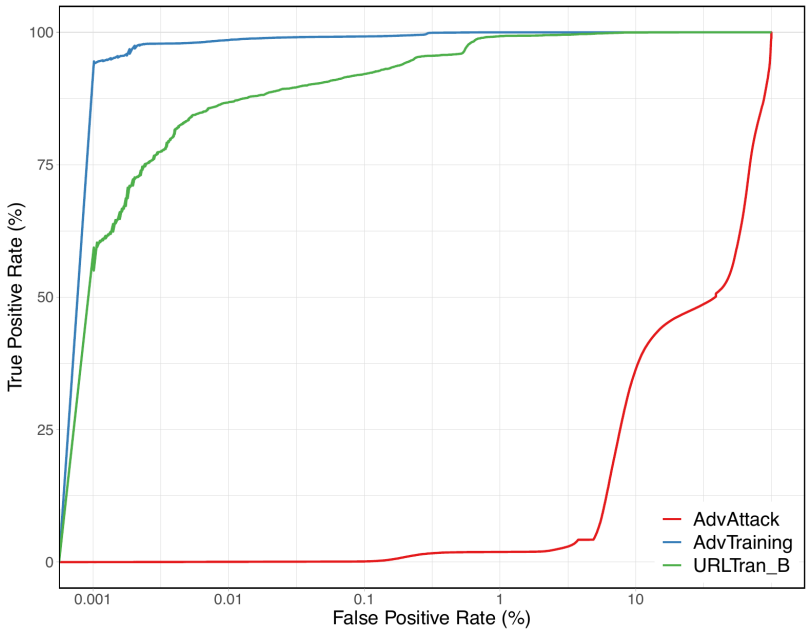


Figure 2.6: ROC curve for URLTran$_B$ when under adversarial attack, and adversarial robustness after augmented training

Training and Inference Times. The total time required to train the best URLTran$_B$ model was 4:57:11 on an NVIDIA V100. Inference required 0:10::44 to complete for an average of 0.36096 milliseconds per sample.

2.5.3 Adversarial Evaluation.

To understand URLTran's robustness to adversarial attacks, we first compared the low FPR regions of the ROC curve of the unprotected model tested with the original test set to the test set which includes adversarial samples (AdvAttack) generated through the methods described in Section 2.4.3 (Figure 2.6). There is a significant drop in performance of URLTran$_B$ when attacked with adversarial URLs. As discussed previously (Section 2.4.3, *adversarial testing* provides a mechanism to ensure that models can be made robust to attacks that follow known threat models. We test this hypothesis by considering the scenario

Parameter	Value
max_len_words	200
max_len_chars	1000
max_len_subwords	20
min_word_freq	1
dev_pct	0.001
delimit_mode	1
emb_dim	32
filter_sizes	[3,4,5,6]
default_emb_mode	char + wordCNN
nb_epochs	5
train_batch_size	128
train_l2_reg_lambda	0.0
train_lr	0.001

Table 2.3: Hyperparameters used for URLNet.

where attack strategies are incorporated into the training data (AdvTraining). On the addition of adversarial attack patterns to the training, the model is able to the adapt to novel attacks, and even exceeded the performance of the non-adversarially trained version of URLTran. These results demonstrate that creating adversarial can help models such as URLTran adapt to unseen attacks. Further, as new attack strategies are recognized (e.g., alternative homoglyph), a robust version of URLTran can be trained to recognize similar patterns in unseen test data.

2.6 Hyperparameter Settings

For replicability, this sectino provides the hyperparameter settings for the three variants of the proposed URLTran model as well as those for two baseline models. Tables 2.3 and 2.4 list the hyperparameters for the URLNet and Texception models that we use as baselines in our study. The hyperparameter settings for the best performing URLTran$_B$ model are provided in Table 2.5. In addition, the best hyperparameter settings for the URLTran$_R$ and URLTran$_C$ are given in Tables 2.6 and 2.7, respectively.

	Parameter	Value
Characters Branch	embedding dimension	32
	number of blocks	1
	block filters	[2,3,4,5]
	Adaptive MaxPool output	32,32
	maximum characters	1000
Words Branch	embedding dimension	32
	number of blocks	1
	block filters	[1,3,5]
	Adaptive MaxPool output	32,16
	maximum words	50
FastText Model	minimum words to include	50
	vocabulary size	120000
	window size	7
	n-grams	2-6
	embedding dimension	32
	epochs trained	30

Table 2.4: Hyperparameters used for Texception.

Parameter	Value
attention probs dropout prob	0.1
hidden act	gelu
hidden dropout prob	0.1
hidden size	768
initializer range	0.02
intermediate size	3072
layer norm eps	1e-12
max position embeddings	512
num attention heads	12
num hidden layers	12
type vocab size	2
vocab size	30522
bert model	bert-base-uncased
max seq length	128
train batch size	32
learning rate	2e-5
num train epochs	10

Table 2.5: Hyperparameters used for training the proposed Huggingface-based URLTran$_B$ model.

Parameter	Value
Number of Layers	12
Hidden size	768
FFN inner hidden size	3072
Attention heads	12
Attention head size	64
Dropout	0.1
Attention Dropout	0.1
Warmup Steps	508
Peak Learning Rate	1e-4
Batch Size	2k
Max Epochs	10
Learning Rate Decay	Linear
Adam ϵ	1e-6
Adam β_1	0.9
Adam β_2	0.98
Gradient Clipping	0.0
Tokens per sample	256

Table 2.6: Hyperparameters used for fine-tuning the proposed Fairseq-based URLTran$_R$ model.

Parameter	Value
Number of Layers	3
Hidden size	768
FFN inner hidden size	3072
Attention heads	12
Attention head size	64
Dropout	0.1
Attention Dropout	0.1
Tokens per sample	128
Peak Learning Rate	1e-4
Batch Size	2k
Tokenizer Type	Byte BPE
Weight Decay	0.01
Max Epochs	30
Learning Rate Decay	reduce on plateau
LR Shrink	0.5
Adam ϵ	1e-6
Adam β_1	0.9
Adam β_2	0.98
Gradient Clipping	0.0
Learning Rate	1e-4
vocab size	10000

Parameter	Value
Learning Rate	1e-4
Batch Size	2k
Max Epochs	10
Learning Rate Decay	Linear
Warmup ratio	0.06

Table 2.7: Hyperparameters used for pre-training (left) and fine-tuning (right) the proposed URLTran$_C$ model.

2.7 Conclusion

This work focused on the *pre-deplyment* stage for building more adaptive models by incorporating *adversarial testing*. We have proposed a new transformer-based system called URLTran whose goal is to predict the label of an unknown URL one which either references a phishing or a benign web page. Transformers have demonstrated state-of-the-art performance in many natural language processing tasks, and the secnd objective of this work is to understand if these methods can also work well in the cybersecurity domain. We demonstrated that transformers which are fine-tuned using standard BERT tasks and a BPE tokenizer also work remarkably well for the task of predicting phishing URLs. Results indicate that URLTran was able to significantly outperform recent baselines, particularly over a wide range of very low false positive rates. We also demonstrated that transformers can be made robust to novel attacks under specific threat models when we adversarially augment the training data used for training them.

Chapter 3: Auditing Fairness Online through Iterative Refinement

Machine learning algorithms are increasingly being deployed for high-stakes scenarios. A sizeable proportion of currently deployed models make their decisions in a black-box manner. Such decision-making procedures are susceptible to intrinsic biases, which has led to a call for accountability in deployed decision systems. In this work, we focus on user-specified accountability of decision-making processes of black box systems. Previous work has formulated this problem as run time fairness monitoring over decision functions. However, formulating appropriate specifications for situation-appropriate fairness metrics is challenging. We construct AVOIR, an automated inference-based optimization system that improves bounds for and generalizes prior work across a wide range of fairness metrics. AVOIR offers an interactive and iterative process for exploring fairness violations aligned with governance and regulatory requirements. Our bounds improve over previous probabilistic guarantees for such fairness grammars in online settings. We also construct a novel visualization mechanism that can be used to investigate the context of reported fairness violations and guide users toward meaningful and compliant fairness specifications. We then conduct case studies with fairness metrics on three different datasets and demonstrate how the visualization and improved optimization can detect fairness violations more efficiently and alleviate the issues with faulty fairness metric design. A part of this work was included in a conference paper (Maneriker et al., 2023a) presented at SIGKDD 2023, and the details of the implementation of AVOIR

were included in a poster presented at the OSU TDAI Interdisciplinary Research Fall Forum 2023.

3.1 Introduction

Advanced analytics and artificial intelligence (AI), along with its many benefits, pose significant threats to individuals and the broader society. Hirsch et al. (2020) identify invasion of privacy; manipulation of vulnerabilities; bias against protected classes; increased power imbalances; error; opacity and procedural unfairness; displacement of labor; pressure to conform, and intentional and harmful use as some of the key areas of concern. A core part of the solution to mitigate such risks is the need to make organizations accountable and ensure that the data they leverage and the models they build and use are both inclusive of marginalized groups and resilient against societal bias. Deployed AI and analytic systems are complex multi-step processes that can incorporate several sources of risk at each step. At each of these stages, determining accountability in the decision-making of AI processes requires a determination of who is accountable, for what, to whom, and under what circumstances (Nissenbaum, 1996; Cooper et al., 2022). A more comprehensive overview of the mechanisms that can support accountability for the different stages of machine learning system design can be found in the work of Cooper et al. (2022). Our analysis centers on auditing fairness claims of mathematical guarantees associated with automated, black-box decision-making processes. Governments worldwide are wrestling with different implementations of auditing regulations and practices to increase the accountability of decision processes. Recent examples include the New York City auditing requirements for AI hiring tools (Vanderford, 2022), European data regulation (GDPR (Parliament, 2018)), accountability bills (Congress, 2023; Office and Data, 2021) and judicial reports (Justice Srikrishna, 2018). These societal forces have

led to the emergence of checklists (Mitchell et al., 2019; Sokol and Flach, 2020), metrics of fairness (Verma and Rubin, 2018), and recently, algorithms and systems that observe and audit the behavior of AI algorithms. Such ideas date back to the 1950s (Moore, 1956). However, research has been sporadic until very recently, with the widespread use of AI-based decision-making giving rise to the vision of algorithmic auditing (Galdon Clavell et al., 2020). In this work, we present a framework called AVOIR[6], for auditing and verifying fairness online. AVOIR builds upon the ideas on distributional probabilistic fairness guarantees (Albarghouthi and Vinitsky, 2019; Bastani et al., 2019), generalizing them to real-world data.

3.2 Background and Key Contributions

Fairness criteria quantify the relationship between the outcome metric across multiple subgroups or similar individuals in the population. Formal definitions of fairness focus on observational criteria, i.e., those that can be written down as a probability statement involving the joint distribution of the features, sensitive attributes, decision-making function, and actual outcome. Consider a decision-making function that claims to satisfy certain fairness guarantees. In our setup, auditing a claim about a fairness guarantee would involve quantifying the probability of claim violations. Given a particular failure probability Δ and a stream of data $\ldots, (X_t, Y_t), \ldots$ over time steps t at run time, a fairness claim ψ would be considered valid if $\Pr[\forall t \geq 1, \psi] \geq 1 - \Delta$. Assuming that the data is sampled from a fixed, possibly unknown distribution p_{data}, a common strategy to test the validity of a claim is to use hypothesis testing with a predetermined sample size m. However, it is impossible to know a priori whether m will be large enough to verify this hypothesis (Waudby-Smith et al., 2021), and peeking at the data to determine the sample size would be considered

[6]AVOIR in French means "to have", and this acronym reflects both our aspirational goal to achieve fairness in advanced analytics and AI but also reflects what is currently verifiable given a dataset, a model, and a fairness specification.

'p-hacking.' Collecting labeled data for fairness-related applications is expensive (Ji et al., 2020); therefore, it is essential to ensure that a monitoring system used for auditing the fairness claim can *adaptively and continuously update its estimates* of the probability of validity. We consider a claim as invalid if $\Pr[\forall t \geq 1, \neg\psi] \geq 1 - \Delta$, where $\neg$ denotes negation. Another desirable feature in the auditing system would be a *finite-horizon stopping rule* that should be able to decide the validity/invalidity of a claim, given sufficient data.

We show that the framework of confidence sequences/sets (Howard et al., 2021) provides a mechanism for building confidence intervals for inference in sequential experiments with nonasymptotic (i.e., always valid for $t \geq 1$) intervals that approach zero width, ensuring that a stopping rule would have a finite termination. We would also like to be able to *localize and diagnose* terms within a fairness metric that leads to the inference of a negated claim. For example, suppose $r \in \{0, 1\}$ denotes the return value of a binary decision function (say, job candidate selection), and s is an indicator denoting whether a candidate belongs to a minority population. The 80%-rule for disparate impact (EEOC, 1979; Feldman et al., 2015) is a fairness criterion which states that

$$\frac{\Pr[r = 1|s]}{\Pr[r = 1|\neg s]} \geq 0.8$$

Assuming that a confidence sequence approach leads to the inference of a negated claim (invalid) for disparate impact, a diagnosis would determine whether the numerator or denominator in the criterion lead to the invalidity. AVOIR uses an inference framework that builds upon distributional guarantees for each term within the criterion, which can help with such a diagnosis. Further, overall uncertainty can be guaranteed across multiple groups by balancing it across subexpressions with differences in the number of observed samples. For example, consider Bernoulli r.vs[7] $X_{1,2}$ for which we derive concentration guarantees

[7] random variables

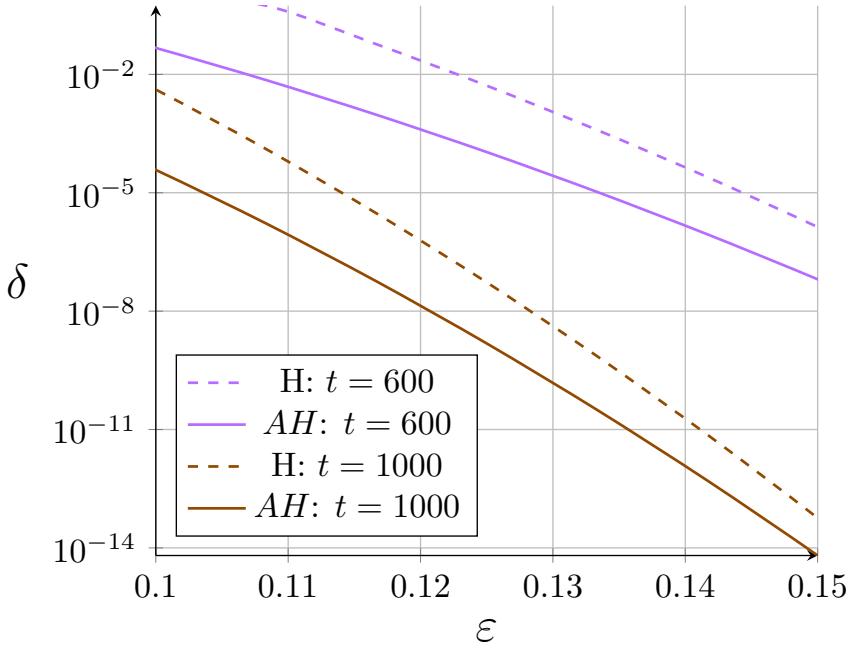


Figure 3.1: Failure probability δ of a Bernoulli r.v. vs concentrated around mean ε for different n. At the same concentration, lower failure probability for the majority class (greater n). H = (online) Hoeffding, AH = Adaptive Hoeffding.

$\Pr[|\mathbb{E}[X_i] - \overline{\mathbb{E}}[X_i]| \geq \epsilon_i] \leq \delta_i$ after t_i observations. Here, $\mathbb{E}[X]$ refers to the population mean, $\overline{\mathbb{E}}[X]$ refers to an empirical mean based on observations of X, and $\epsilon, \delta > 0$ are the concentration level and failure probability, respectively. From the Hoeffding inequality, $\delta = 2e^{-2t\epsilon^2}$. We can claim tighter bounds for X_2 if $t_2 > t_1$ as the failure probability δ is lower at the same concentration ϵ. That is, $\varepsilon_1 = \varepsilon_2, t_2 > t_1 \Rightarrow \delta_1 > \delta_2$. Varying ϵ across subexpressions to minimize the overall (union bounded) $\delta = \delta_1 + \delta_2$ allows an earlier stopping time for a valid/invalid claim, i.e., *fewer iterations and fewer data samples.* Adaptive versions of these inequalities also have similar patterns (see Figure 3.1).

Consider R, a Bernoulli r.v corresponding to the output of a binary decision function, with s being an indicator of class membership. Let $X = r \vee s$ and $Y = r \vee \neg s$ be r.vs corresponding to a favorable decision for the majority and minority classes, respectively. Suppose we aim to

Symbol	Description
Δ	Overall failure probability for a specificaiton
X_i	Bernoulli random variable for i^{tg} tern
$\overline{\mathbb{E}}[X]$	Empirical estimate of expectation $\mathbb{E}[X]$
t	No. of observed samples
$\overline{\mathbb{E}}[X_i]_t$	Empirical estimate after t steos
δ_i	Failure probability $0 \leq \delta_i \leq 1$ corresponding to X_i
ε_i	Concetration bound for $\lvert E[X_i] - \overline{\mathbb{E}}[X_i] \rvert \leq \varepsilon_i$
ϕ_X	Concentration bound for empirical estimate of $E[X]$
ψ	Fairness specification
r, R	Return value of the function being monitored
y, Y	True label
s, S	Indicator for group membership
c	Constant $\in \mathbb{R}$
C	A set of constraints

Table 3.1: The AVOIR symbol descriptions table.

estimate a criterion $\psi := E[X] - E[Y] < \epsilon_T$ Previous work on inference from distributional guarantees (Albarghouthi and Vinitsky, 2019; Bastani et al., 2019) assumes equal failure probability across all groups, i.e., the assumption $A_\delta := \delta_1 = \delta_2$. Suppose we want the upper bound of the failure probability $\Delta = 0.1$ for the specified criterion. Consider a n_X, n_Y observations for X, Y such that $\overline{\mathbb{E}}[X] = 0.8, n_X = 1550$ and $\overline{\mathbb{E}}[Y] = 0.5, n_Y = 310$. Figure 3.2 shows that no solution is feasible for the optimization problem with A_δ. However, AVOIR can find a solution. For the optimal solution, $\delta_2 \approx 2.35\delta_1$, which aligns with our intuition about allocating higher failure probability to terms with the majority of observations. The optimization problem inferred by AVOIR:

$$\min_{\delta_X, \delta_Y} \delta_X + \delta_Y$$

$$\text{s.t. } \epsilon_X + \epsilon_Y \leq \overline{\mathbb{E}}[X] - \overline{\mathbb{E}}[Y] - \epsilon_T$$

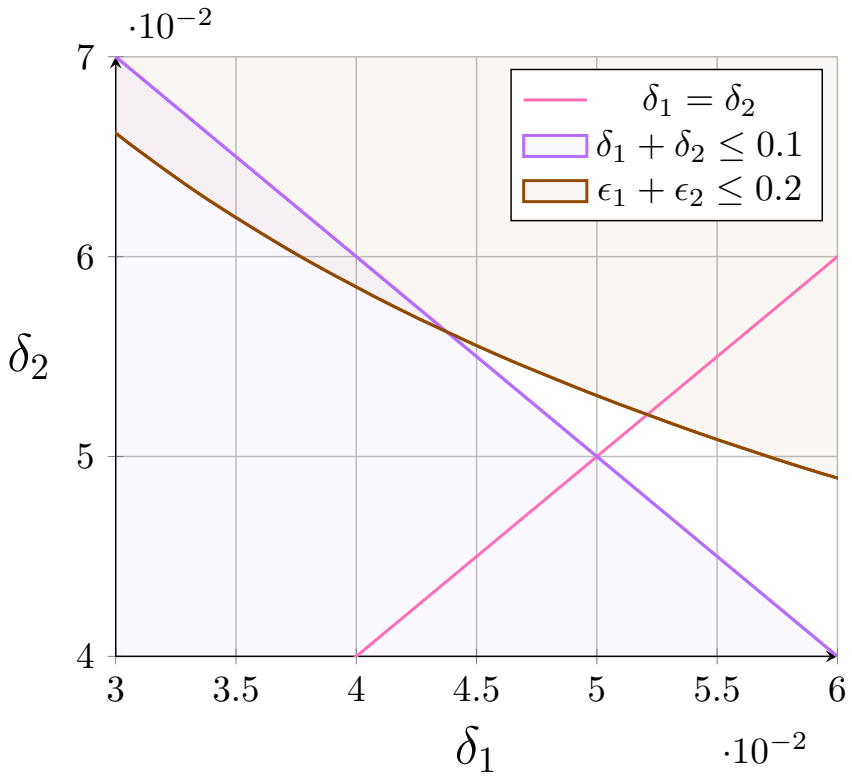


Figure 3.2: AVOIR finds a solution for a *theoretical* scenario with $\delta_1 + \delta_2 \leq \Delta$ under constraint $\epsilon_1 + \epsilon_2 \leq \epsilon_T$. No solution exists with additional constraint $A_\delta : \delta_X = \delta_Y = \Delta/2$ - common assumption in prior work.

3.3 AVOIR Framework

3.3.1 Definitions

AVOIR supports implementing an extensive range of group fairness criteria, including demographic parity (Calders et al., 2009), equal opportunity (Hardt et al., 2016), disparate mistreatment (Zafar et al., 2017), and combinations of these criteria. For instance the above 80%-rule is E[r|S==s]/E[r|S!=s] > 0.8 in AVOIR's DSL[8]. Here, the term E[r|S==s]/E[r|S!=s] is a *subexpression* of the specification. The smallest units involving an expectation (e.g., E[r|S!=s]) are denoted as *elementary subexpressions*. We focus on fairness criteria that can be expressed using Bernoulli r.v. as it allows the simplification of probabilities into expectation, as $\Pr[r = 1] = \mathbb{E}[r]$ (hereafter, used interchangably). Our algorithm uses adaptive concentration sets (Zhao et al., 2016; Howard et al., 2021) to build estimates for *elementary subexpressions* and then derive the estimates for expressions that combine them. A combination of multiple such elementary expressions is denoted as a *compound* expression. We aim to derive statistical guarantees about fairness criteria based on estimates from observed inputs and outputs. For example, let X be an observed Bernoulli r.v, then an assertion $\phi_X = (\overline{\mathbb{E}}[X], \epsilon, \delta)$ over X, corresponds to an estimate satisfying $\phi_X := \Pr[|\mathbb{E}[X] - \overline{\mathbb{E}}[X]| \geq \epsilon] \leq \delta$ where $\overline{\mathbb{E}}[X]$ denotes an empirical estimate of $E[X]$. We then use assertions ϕ_X, ϕ_Y to assert claims for expressions involving X, Y. For example, for the 80%-rule, assertions over $\mathbb{E}[X]/\mathbb{E}[Y]$. A *specification* involves either a comparison of expressions with constants (e.g., $\mathbb{E}[X]/\mathbb{E}[Y] > 0.8$) or combinations of multiple such comparisons. Such a specification may be True (T) or False (F) with some probability. For a given specification ψ, we denote the claim that $P[\psi = F] \geq 1 - \Delta$ as $\psi : (F, \Delta)$, where Δ denotes the failure probability of a guarantee. Given a stream of observations and outcomes

[8]Domain Specific Language

$\langle spec \rangle$ ::= $\langle ETerm \rangle$ $\langle comp\text{-}op \rangle$ c
| $\langle spec \rangle \wedge \langle spec \rangle$
| $\langle spec \rangle \vee \langle spec \rangle$

$\langle ETerm \rangle$ ::= $\mathbb{E}[\langle E \rangle]$
| $\mathbb{E}[\langle E \rangle$, given=$\langle E \rangle]$
| c $\in \mathbb{R}$
| $\langle ETerm \rangle$ $\{+, -, \times, \div\}$ $\langle ETerm \rangle$

Figure 3.3: Grammar for specification. $\langle E \rangle$ refers to expressions of r.vs and $\langle comp - op \rangle$ = comparison operator $\in \{>, <, =, \neq\}$.

from the decision functions, and a specified threshold probability Δ, we will continue to refine the estimate for a given specification until we reach the threshold. Specifications involving variables that take more than two values can be implemented using transformations and boolean operators (examples in Appendix 3.D).

3.3.2 Language Specification

We describe AVOIR's DSL used for specifying fairness metrics (Figure 3.3). We focus on binary decision-making functions; Bernoulli r.v.s can characterize their outputs. Consider a decision function $f : X \to \{0, 1\}$, where $X = (X_1, \ldots, X_k)$ denotes a real-valued input vector. We use $R = f(X)$ to simplify the remainder of the definitions. The grammar can be used to construct Bernoulli r.vs to support expressions beyond those that produce binary outputs. For example, a ν-threshold based real-valued output, $R' = (R > \nu)$ and a multi-class output, for class j, $R' = (R == j)$ correspond to Bernoulli r.vs. Expressions involving R and X_i act as the arguments <E> to construct an <ETerm>. For example, $\mathbb{E}[R > 0 | X_1 + X_2 > a]$. c terms represent constant real values used as bounds for specifications. We modified the grammar from prior work to include two additional operations. First, we added a `given` argument to $\mathbb{E}$, which allows a user to specify conditional probabilities directly, in contrast

to specifying it as a ratio of joint/marginal probabilities.

$$\frac{\mathbb{E}(A \vee (B = b))}{\mathbb{E}(B = b)} \rightarrow \mathbb{E}(A, \texttt{given} = (B = b))$$

which is used to represent $\mathbb{E}[A|B = b]$, simplifying expressions for group fairness specification. Additionally, we add comparison operators, which further simplify the process of writing specifications.

3.3.3 Propagating Bounds

Generating the bounds for a specification requires propagating them from elementary subexpressions. Assuming that observed values for each `<E>` correspond to an underlying random variable X, a probabilistic guarantee ϕ_X for an *elementary* subexpression consists of an empirical estimate $\overline{\mathbb{E}}[X]$, a concentration bound ϵ_X, and a failure probability δ_X, such that $\Pr[|\mathbb{E}[X] - \overline{\mathbb{E}}[X]| \geq \epsilon_X] \leq \delta_X$. For compound expressions, we must infer the implied guarantees that can be inferred with corresponding constraints. Each inference rule corresponds to a derivation in the DSL grammar. Inference rules have preconditions and postconditions that are in the form:

$$\frac{\bigcup\{r|r \in \{\phi, \psi, C\}\}}{\bigcup\{s|s \in \{\phi, \psi, C\}\}}$$

where ϕ denotes a claim for a subexpression, ψ for a `<spec>`.For example, consider the sum/difference rule. Starting with the assumptions $\phi_X := (\overline{\mathbb{E}}[X], \epsilon_X, \delta_X)$, $\phi_Y := (\overline{\mathbb{E}}[Y], \epsilon_Y, \delta_Y)$. Then we have

$$\begin{aligned} &|\mathbb{E}[X] \pm \mathbb{E}[Y] - (\overline{\mathbb{E}}[X] \pm \overline{\mathbb{E}}[Y])| \\ &\qquad \leq |\mathbb{E}[X] - \overline{\mathbb{E}}[X]| + |\mathbb{E}[Y] - \overline{\mathbb{E}}[Y]| \\ &\qquad \leq \epsilon_X + \epsilon_Y \end{aligned}$$

i.e., $\phi_X, \phi_Y \Rightarrow X \pm Y : \big(\overline{\mathbb{E}}[X] \pm \overline{\mathbb{E}}[Y], \epsilon_X + \epsilon_Y, \delta_X + \delta_Y\big)$. Inference rules may require constraints, for e.g., assume $\phi_X := (\overline{\mathbb{E}}[X], \epsilon_X, \delta_X)$, $\overline{\mathbb{E}}[X] > c$. Then we have $\Pr[\mathbb{E}[X] < \overline{\mathbb{E}}[X] - \epsilon_X] > 1 - \delta$ If we add the constraint that $\overline{\mathbb{E}}[X] - \epsilon_X \geq c$, we have $\Pr[X < c] > 1 - \delta$, thus,

$$\phi_X \Rightarrow \psi := X > c : (T, \delta_X)$$

$$\text{under the constraint } \{\overline{\mathbb{E}}[X] - \epsilon_X \geq c\}$$

The complete set of inference rules required for the DSL is provided in the appendix (Figure 3.A.1). The implementation in AVOIR follows these rules but could be extended to other rule inference templates that support the DSL. Note that these rules extend the ones implemented by VF (Bastani et al., 2019) with constraints that enable the optimizations required in AVOIR.

3.3.4 Optimizing Bounds

3.3.4.1 AVOIR Algorithm

The pseudocode for the optimization procedure in AVOIR is described in Algorithm 1. The input to the algorithm is the reporting threshold probability Δ and a specification ψ. We then infer a symbolic optimization problem corresponding to the bounds and failure probabilities of the elementary subexpressions. At each step, the `OBSERVE(X)` function is called with the new observation of every elementary subexpression and output. The empirical running means and counts of observations are updated. The final optimization problem `OPT` corresponding to each specification is a nonlinear constrained optimization problem. If a solution is successfully found for `OPT`, the algorithm terminates, and the estimate for the specification has reached the required threshold. If no solution is found, the estimates will be updated with $\delta_i = \Delta$ for each *elementary* subexpression. The intuition behind the algorithm

Algorithm 1 AVOIR Algorithm

Require: Δ, ψ ▷ Δ, Specification
Ensure: T_s time step when the value of ψ can be guaranteed with probability $\geq 1 - \Delta$

for $X_i \in \psi$ **do**
    $\delta_{X_i} = \Delta$ ▷ Set initial value $\forall i$
    $S_{X_i} = 0$ ▷ Sum of observations
    $n_{X_i} = 0$ ▷ Number of observations
end for
$T = 0$ ▷ Time step
Initialize OPT_ψ ▷ Initialize Optimization Problem (Fig. 3.A.1)
procedure OBSERVE(X)
    for $X_i \in X$ **do**
        $S_{X_i} = S_{X_i} + X_i$
        $n_{X_i} = n_{X_i} + 1$
        $\overline{\mathbb{E}}[X_i] = S_{X_i}/n_{X_i}$
        Initialize δ_{X_i} as a symbolic variable
        Assign $\epsilon(\delta_{X_i}, n_{X_i})$ symbolic variable
    end for
    Propagate δ_{X_i} using the inference rules
    Initialize constraints g_K in OPT_ψ using the computed values
    $\delta_T^* = \texttt{Solve}(OPT_\psi)$
    if $\delta_T^* \leq \Delta$ **then**
        $\delta_{X_i} = \delta_T^*[X_i]$
        return $T_s = T$
    end if
    $T = T + 1$
end procedure

is to use a confidence sequence corresponding to the estimates of elementary subexpressions at each time step. The inferred `OPT` has the form

$$\begin{aligned} &\min_{0\leq\delta_i\leq 1} \sum_{i=1}^{n} \delta_i \\ &\text{s.t. } g_k(\delta_{1,\ldots,n}, \overline{\mathbb{E}}[X_1], \ldots, \overline{\mathbb{E}}[X_n]) \leq \epsilon_k \end{aligned} \tag{3.1}$$

where g_k, ϵ_k are the functions/bounds derived using the transformations carried out through the inference rules (Appendix 3.A.2).

Definition 1. *For $\delta \in [0,1]$, a $1-\delta$ confidence sequence is a sequence of confidence sets, usually intervals $(\mathrm{CI_t})_{\mathrm{t}=1}^{\infty}$,, $\mathrm{CI_t} := (\mathrm{L_t}, \mathrm{R_t}) \subseteq \mathbb{R}$ satisfying a uniform convergence guarantee. After observing the t^{th} unit, we calculate an updated confidence set $\mathrm{CI_t}$ for an unknown quantity θ_t with the coverage property $\Pr(\forall t \geq 1, \theta_t : \theta_t \in \mathrm{CI_t}) \geq 1-\delta$ (Howard et al., 2021).*

In this paper, we focus on the mean of r.v.s $\mathbb{E}[X]$ that constitute estimates for *elementary* subexpressions as the quantities of interest. We use adaptive concentration inequalities to construct these confidence sequences. Any adaptive concentration inequality that can be applied to an r.v. $X \in \{0,1\}$ such that

$$\Pr[|\overline{\mathbb{E}}_t[X] - \mathbb{E}[X]| \geq \epsilon(t,\delta)] \leq \delta \tag{3.2}$$

can be used in AVOIR. Here, $\overline{\mathbb{E}}_t[X]$ is the empirical estimate of $\mathbb{E}[X]$ after the t^{th} observation. For comparison with previous work (e.g., VF), we use the Adaptive Hoeffding Inequality $\mathrm{AIN_H}$ (Zhao et al., 2016).

Theorem 1 (AIN_H)**.** *Given a Bernoulli random variable X with distribution P_X. Let $\{X_i \sim P_X\}, i \in \mathbb{N}$ be i.i.d samples of X. Let*

$$\overline{\mathbb{E}}_t[X] = \frac{1}{t}\sum_{i=1}^{t} X_i.$$

Let $\mathcal{T}$ be a r.v on $\mathbb{N} \cup \{\infty\}$ such that $\Pr[\mathcal{T} < \infty] = 1$, and let

$$\epsilon(\delta, t) = \sqrt{\left(\frac{3}{5}\log\left(\log_{1.1} t + 1\right) + \frac{5}{9}\log\left(24/\delta\right)\right) \Big/ t}$$

Then, for any $\delta \in \mathbb{R}_+$, we have

$$\Pr[|\overline{\mathbb{E}}_{\mathcal{T}}[X] - \mathbb{E}[X]| \leq \epsilon(\delta, \mathcal{T})|] \geq 1 - \delta.$$

We will generate estimates using AIN_{H} and Corollary 4.1 for *elementary* subexpressions that are valid nonasymptotically (i.e., $\forall t > 1$) and then expand this to compound subexpressions.

Theorem 2. *The sequences of estimates generated by AVOIR form a confidence set.*

The intuition for the proof is as follows: first, for elementary subexpression X, let the failure probability at the stopping time be δ_X^*. From equation 3.1, we can show that $\Delta \geq \delta_X^*$. Further, $\epsilon(\delta, t)$ is monotonically decreasing in δ. Thus, setting $\delta_X(t) = \Delta$ as per Algorithm 1 before stopping time will ensure that the estimated confidence intervals before the stopping time corresponding to each time step for X would be a subset of the optimized values,

$$\left(\overline{\mathbb{E}}[X]_t \pm \epsilon(\delta_X^*, t)\right) \subseteq \left(\overline{\mathbb{E}}[X]_t \pm \epsilon(\Delta, t)\right)$$

where $(\mu \pm \sigma) = (\mu - \sigma, \mu + \sigma)$. Next, for compound subexpressions and specifications, the correctness of the inference rules used for propagating bounds (Figure 3.A.1) can be used to prove that the confidence sequence is valid nonasymptotically. We now proceed with the detailed proof. First, we assume the existence of a confidence sequence for the mean of each elementary subexpression (e.g., using Theorem 1). That is, we need an AIN for $\epsilon(t, \delta)$ such that

$$\Pr[\forall t \geq 1, |\overline{\mathbb{E}}_t[X] - \mathbb{E}[X]| \leq \epsilon(t, \delta_X)] \geq 1 - \delta_X. \tag{3.3}$$

We assume $\varepsilon(t, \delta)$ to be monotonically non-increasing in δ and n. We expect this to be the case for most AIN, since increasing the number of observations of increasing the failure threshold should allow for additional concentration around the mean (e.g., this holds for AIN_{H}) Second, we assume that except in degenerate cases, AVOIR terminates at finite stopping time $\mathcal{T}$ (termination criteria in Corollary 3.2, Appendix).

Proof. Elementary subexpressions: Consider a specification ψ consisting of *elementary* subexpressions $X_1, \ldots, X_n$. At stopping time, let $\phi_{X_i}^{\mathcal{T}} := (\overline{\mathbb{E}}_{\mathcal{T}}[X_i], \epsilon(\mathcal{T}, \delta_{X_i}), \delta_{X_i})$ be the stopping time estimates. Then, from the termination criterion, a solution to the optimization problem `OPT` exists, i.e,

$$\Delta \geq \sum_i \delta_{X_i} \tag{3.4}$$

The sequence of bounds claimed by AVOIR are

$$\epsilon_{X_i}(t) = \begin{cases} \epsilon(\Delta, t), & t < \mathcal{T}, \\ \epsilon(\delta_{X_i}, t), & t \geq \mathcal{T} \end{cases} \tag{3.5}$$

From equation 3.4 and since $\delta_i \in [0, 1]$ we have $\Delta \geq \delta_{X_i}$. From the non-decreasing behavior of AIN

$$\varepsilon(\Delta, t) \leq \varepsilon(\delta_i, t) \tag{3.6}$$

Now

$$\begin{aligned}
&\Pr[\forall t \geq 1, |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| \leq \epsilon_{X_i}(t)] \\
&= 1 - \Pr[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)] \\
&= 1 - \Pr\left[\bigcup_{t \geq 1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)\right\}\right] \\
&= 1 - \Pr\left[\bigcup_{t=1}^{\mathcal{T}-1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)\right\} \cup \right. \\
&\left. \bigcup_{t \geq \mathcal{T}} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)\right\}\right] \qquad (\cup \text{ associativity})
\end{aligned}$$

$$
\begin{aligned}
&= 1 - \Pr\Bigg[\bigcup_{t=1}^{\mathcal{T}-1} \big\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t) \\
&\cup |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| \in (\epsilon(\Delta, t), \epsilon(\delta_{X_i}, t)]\big\} \cup \\
&\bigcup_{t \geq \mathcal{T}} \big\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\big\}\Bigg] && \text{(Using 3.5, 3.6)} \\
&= 1 - \Pr\Bigg[\bigcup_{t=1}^{\mathcal{T}-1} \big\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| \in \\
&(\epsilon(\Delta, t), \epsilon(\delta_{X_i}, t)]\big\} \cup \\
&\bigcup_{t \geq 1} \big\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\big\}\Bigg] && \text{(Rearranging)} \\
&\geq 1 - \Pr\left[\bigcup_{t \geq 1} \big\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\big\}\right] \\
&= 1 - \Pr\left[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\right] \\
&\geq 1 - \delta_{X_i}
\end{aligned}
$$

where the last step follows from the definition of the AIN used. Thus, $\epsilon_{X_i}(t)$ defines a $1 - \delta_{X_i}$ confidence sequence for $\mathbb{E}[X_i]$.

Compound subexpressions: Consider a non-specification compound (`<ETerm>`) C_j consisting of *elementary* subexpressions with indices $\mathbf{C}_j = \{\{j_1, j_2, \ldots, j_M\}\}$ as the decision r.v.s, i.e, $X_{j_1} \ldots, X_{j_M}$. Note that $\mathbf{C}_j$ is a multiset as the same expression could occur multiple times within C_j. At stopping time $\mathcal{T}$,

$$\phi^{\mathcal{T}}_{C_j} : (\overline{\mathbb{E}}_{\mathcal{T}}[C_j], \delta_{C_j}, \varepsilon_{C_j}) \tag{3.7}$$

where $\overline{\mathbb{E}}_{\mathcal{T}}[C_j], \delta_{C_j}, \varepsilon_{C_j}$ are the corresponding values computed through the inference rules. In general, we denote by

$$\overline{\mathbb{E}}_t[C_j], \delta_{C_j}(t), \varepsilon_{C_j}(t) = \text{INFER}(\phi^{\text{t}}_{\text{X}_{\text{j}_1}}, \ldots, \phi^{\text{t}}_{\text{X}_{\text{j}_\text{M}}}) \tag{3.8}$$

the values inferred at t, using the inference rules INFER. Now,

$$
\begin{aligned}
&\Pr[\exists t \geq 1, |\mathbb{E}[C_j] - \overline{\mathbb{E}}[C_j]| > \varepsilon_{C_j}(t)] \\
&\quad \leq \Pr\left[\bigcup_{i=1}^{M} \exists t \geq 1, \neg\phi^t_{X_{j_i}}\right] && \text{(eq. 3.8)} \\
&\quad \leq \sum_{i \in \mathbf{C}_j} \Pr\left[\exists t \geq 1, \neg\phi^t_{X_{j_i}}\right] && \text{(union bound)} \\
&\quad = \sum_{i \in \mathbf{C}_j} \Pr\left[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_{j_i}] - \mathbb{E}_t[X_{j_i}| > \epsilon_{X_{j_i}}(t)\right] && \text{(definition of } \phi^t_{X_{j_i}}\text{)} \\
&\quad \leq \sum_{i \in \mathbf{C}_j} \delta_{X_{j_i}} && \text{(elementary subexpressions)} \\
&\quad \leq \delta_{C_j} && \text{(eq. 3.8 at } t = \mathcal{T}\text{)}
\end{aligned}
$$

Therefore $\varepsilon_{C_j}(t)$ defines a $1 - \delta_{C_j}$ confidence sequence for $\mathbb{E}[C_j]$ A similar proof can be constructed for any `<spec>` (section 3.B.1). □

Corollary 2.1. *The estimates for the overall specification ψ form a confidence sequence which staisfies $\psi : (b, \Delta), b \in \{T, F\}$ at $\mathcal{T}$.*

Proof. We initialize the main specification with the required failure probability Δ. At termination, $\sum \delta_i \leq \Delta$. From Theorem 2, we can infer that the confidence sequence corresponding to the termination achieves the threshold Δ, as required. □

3.3.4.2 Improvements over Baseline

In all prior work (Albarghouthi et al., 2017; Albarghouthi and Vinitsky, 2019; Bastani et al., 2019), δ_i for each *elementary* subexpressions is set to Δ/n, where n is the number of elementary subexpressions in the specification. This simplification uses the assumption $A_\delta := \delta_i = \delta_j \ \forall i, j$ for *elementary* subexpressions. As we do not make this assumption, we can prove the following critical theorem (note, Corollary 3.2 describes the conditions required for finite stopping).

Definition 2. *We define the specification stopping time $\mathcal{T}$ for a confidence sequence as the smallest t such that given a threshold Δ and a specification ψ, an inference algorithm terminates with* $\Pr[\forall t \geq 1, \psi_t = \widehat{\psi}_{\mathcal{T}}] \geq 1 - \Delta$, *where $\widehat{\psi}_{\mathcal{T}}$ is the estimate of ψ at $\mathcal{T}$.*

Theorem 3. *Given a threshold probability Δ for a specification ψ, let the stopping time for AVOIR be $\mathcal{T}$ and the stopping time with the A_δ assumption be $\mathcal{T}^+$. Then $\mathcal{T} \leq \mathcal{T}^+$*

Proof. Under A_δ, at the stopping time $\mathcal{T}^+$, $\delta_i^+ = \Delta/n$, with $\sum_{i=1}^{n} \delta_i^+ = \Delta$. As δ_i^+ are propagated using INFER (without constraint rules), we know that they must satisfy the constraints of the optimization problem in eq. 3.1. At time $\mathcal{T}^+$ AVOIR would find solution δ_i^* such that minimizes $\sum_{i=1}^{n} \delta_i$.

$$\sum_{i=1}^{n} \delta_i^* \leq \sum_{i=1}^{n} \delta_i^+ = \Delta$$

Thus, AVOIR would have terminated by step $\mathcal{T}^+$, but may find a feasible solution at an earlier step, i.e., $\mathcal{T} \leq \mathcal{T}^+$. □

3.3.5 Implementation Details

We built a Python library to create specifications as a decorator over decision functions. New input/output observations are monitored to update all the terms in a specification. Inference for evaluating the value and bounds is carried out via operator overloading.In line with previous work (Albarghouthi et al., 2017; Bastani et al., 2019; Albarghouthi and Vinitsky, 2019) on distributional verification, we use rejection sampling for conditional probability estimation. We use the COIN-OR implementation of IPOPT (Wächter and Biegler, 2006), accessed through the Pyomo (Hart et al., 2011) interface for optimization. Code for reproducing this work is available at https://github.com/pranavmaneriker/AVOIR.

3.4 Evaluation

In this section, we evaluate AVOIR.variants through three real-world case studies. Direct comparisons with existing work are impossible since no other work (to our knowledge) facilitates a general-purpose inference engine for online fairness auditing using arbitrary measures. We can, however, implement VF's (Bastani et al., 2019) inference rules within AVOIR (denoted as AVOIR-VF). Note that AVOIR-VF sidesteps the assumptions of having a known data-generating distribution (made possible by AVOIR's reliance on confidence sets), making this variation a more practical and efficient algorithm. We denote AVOIR-OB as the implementation that utilizes the abovementioned optimizations. Across the studies, the role of chosen threshold probabilities is similar to that of p-values in statistics. Typical p-values tend to be 0.05 and 0.1, which we demonstrate in the RateMyProfs and COMPAS risk assessment study. In our case study of prior work (Angwin et al., 2016), we stick to the available definitions and thresholds used in the original analysis. We expect that regulators will set the threshold probabilities on a case-by-case basis, e.g,, 0.15 for illustration purposes in the adult income study.

3.4.1 Rate My Profs

This section provides a detailed black-box machine learning model-based case study on a real-world dataset. This case study uses the Rate My Professors (RMP) dataset (Keymanesh et al., 2021). This dataset includes professor names and reviews for them written by students in their classes, ratings, and certain self-reported attributes of the reviewer. Ratings are provided on a five-point scale (1-5 stars). We use the preprocessing described in (Keymanesh et al., 2021) to infer the gender attribute for the professors. This dataset is divided into an 80-20 split (train-test). We then train a BERT-based transformer model (Devlin et al., 2019)

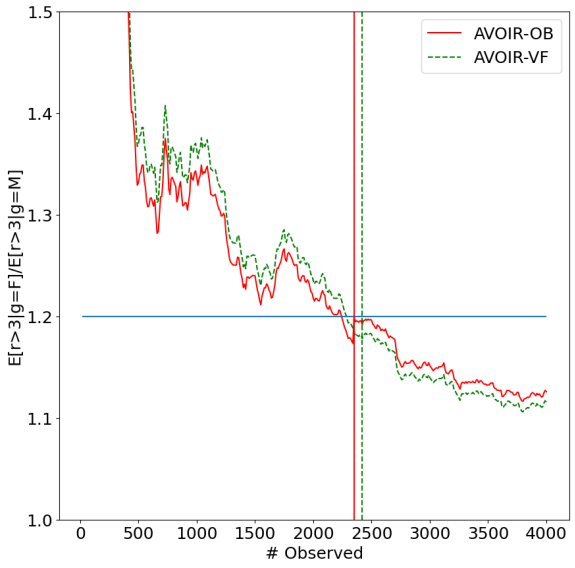


Figure 3.4: Bounds for first half of a gender-fairness specification generated by AVOIR-OB and AVOIR-VF for *RateMyProfs*, a real-world dataset. Vertical lines show the step at which the methods can provide a guarantee of failure for the upper bounds with $\Delta <= 0.05$. Blue horizontal line represents the constant term in the inequality.

on the training split. We use the implementation from the simpletransformers[9] package. The loss function chosen is the mean-squared error from the true ratings. On the test set, we track a gender-fairness specification in the model outputs:

(E[r $>$ 3 | gender = F] / E[r $>$ 3 | gender = M $<$ 1.2) &
(E[r $>$ 3 | gender = M)] / E[r $>$ 3 | gender = F] $>$ 0.8)

We set the failure probability $\Delta = 0.05$. `OPT` is run after each batch (5 items/batch). Figure 3.4 shows that AVOIR-OB[10] can provide a guarantee in **2.5**% fewer iterations than AVOIR-VF. Note also that the OB guarantee provided tries to optimize for the failure probability while staying under the required threshold, remaining closer to the required threshold in subsequent steps.

[9]https://simpletransformers.ai/

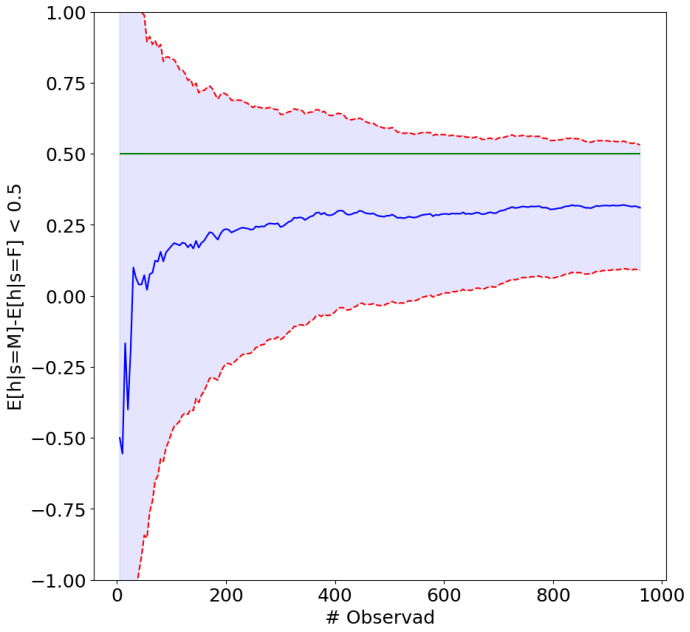
[10]OB = Optimized Bounds

3.4.2 Adult Income

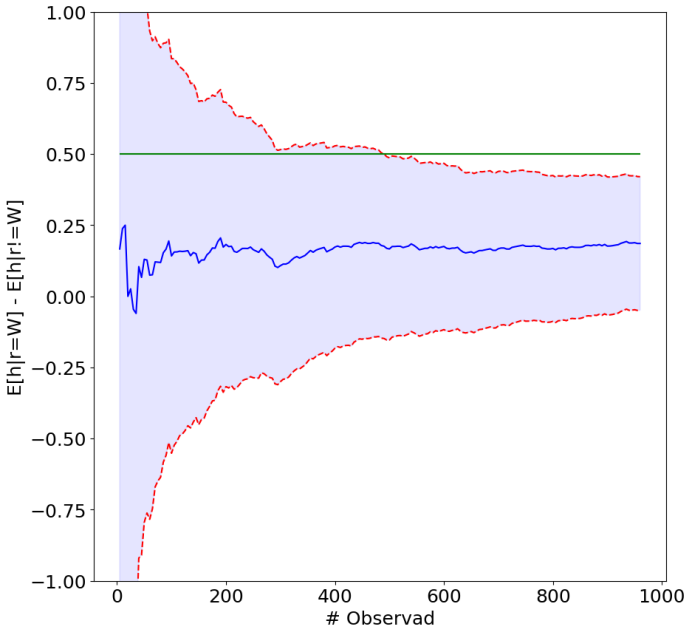
In this case study, we analyze the Adult income dataset (Kohavi, 1996). The historical dataset labels individuals from the 1994 census as having a *high-income* ($>$ 50k a year) or not ($\leq$ 50k a year). We consider this column of data as a black-box measurement. US Federal laws mandate against race and sex-based discrimination. Thus, the specification we start our analysis with is a group fairness property for federal employees that monitors the difference of the proportions of individuals with sex marked male vs. female with a high income should be less than 0.5. In addition, we ensure that the difference between individuals with race marked white and non-white should have a difference of less than 0.5. Thus, we use an *intersectional* fairness criterion. The associated specification is given below, where `h` is an indicator for whether an individual is *high-income* is the binary classification output of our model:

(E[h | sex=M] − E[h | sex=F] < 0.5) & \
(E[h | race=W] − E[h | race!=W] < 0.5)

In this example, we set the failure threshold probability $\Delta = 0.15$ When run with this specification, the generated bounds cannot be achieved with the available data. We can then use the iterative refinement associated with subexpressions to analyze different components of the specification. The plot corresponding to the left subexpression is shown in Figure 3.5a shows that guarantees cannot converge under the threshold with the given number of data samples. An auditor can now choose to either reduce the guarantee (i.e. increase Δ) or increase the threshold. Next, analyzing the right subexpression, the race group fairness term can be guaranteed to be under the threshold (Figure 3.5b). Using this information, an auditor can make a decision to increase the threshold on the group fairness term for sex. As a hypothetical, suppose they increase it from 0.5 to 0.55 and rerun the analysis. OB can provide a guarantee at this threshold within 870 steps, whereas VF can provide it at 960



(a) Group fairness for sex. Difference in ratio of high income (left subexpression).



(b) Group fairness for race. Difference in ratio of high-income earners (right subexpression).

Figure 3.5: *(Top)* Red dotted lines, the upper bounds of the value cannot be guaranteed to be under the threshold at the specified failure probability. *(Bottom)* Guarantee possible with given data. Green lines represent the constant term, and dark blue is the empirical mean.

steps, demonstrating a relative improvement of about **10.35**%. Additionally, the optimal Δ split across the terms is $\approx (0.135, 0.36 * 10^4)$, which is far from the equal split allocated by VF. The reason for this split is that increasing the threshold for the first time provides the optimizer with additional legroom to better distribute the failure probabilities between the two terms.

3.4.3 COMPAS Risk Assessment

The Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) recidivism risk score data is a popular dataset for assessing machine bias of commercial tools used to assess a criminal defendant's likelihood to re-offend. The data is based on recidivism (re-offending) scores derived from software released by Northpointe and widely used across the United States for making sentencing decisions. In 2016, Angwin et al. (2016) at ProPublica released an article and associated analysis code critiquing machine bias associated with race present in the COMPAS risk scores for a set of arrested individuals in Broward County, Florida, over two years. The analysis concluded that there were significant differences in the risk assessments of African-American and Caucasian individuals. Northpointe pushed back in a report (Dieterich et al., 2016) firmly rejecting the claims made by the ProPublica article; instead, they claimed that Angwin et al. (2016) made several statistical and technical errors in the report. In this case study, we use AVOIR to study the claims of the two reports mentioned above. We create a materialized view of the ProPublica dataset by reproducing the preprocessing steps in the publicly available ProPublica analysis notebook[11]. We look at "Sample A" (Dieterich et al., 2016), where the analysis of the "not low" risk assessments using a logistic regression model reveals a high coefficient associated with the factor associated with race being African-American. In terms of a fairness metric, this corresponds to false

[11] https://github.com/propublica/compas-analysis

positive rate (FPR) balance (predictive equality) (Verma and Rubin, 2018) metrics. The associated specification in AVOIR grammar would be

E[hrisk | race=African−American & recid=0] /
E[hrisk | race=Caucasian & recid=0] < 1.1

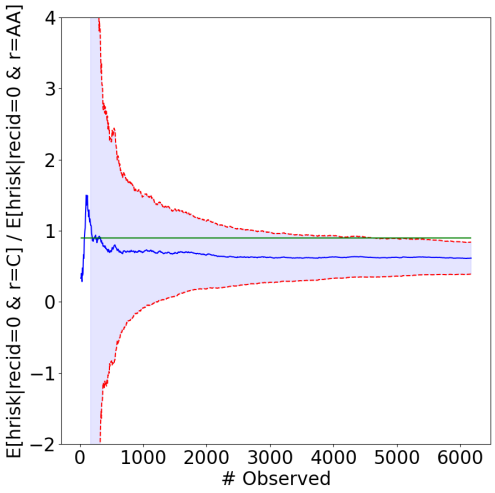
Where `hrisk` is an indicator for high-risk assessments made by the *black-box* COMPAS tool as defined by Angwin et al. (2016), `recid` is an indicator for re-offending within two years of first arrest, and a 90%-rule is used as the threshold. We choose a failure threshold probability of $\Delta = 0.1$, with the optimization run after every batch of 5 samples. AVOIR finds that when the decisions are made sequentially, online, the assertion for specification violation cannot be made with the required failure guarantee.

By analyzing the component subexpressions, one can glean that AVOIR cannot optimize since the lower FPR in the denominator (FPR for Caucasian individuals) increases the overall variance and limits the ability to optimize for guarantees. We follow this analysis by using the reciprocal specification, i.e.,

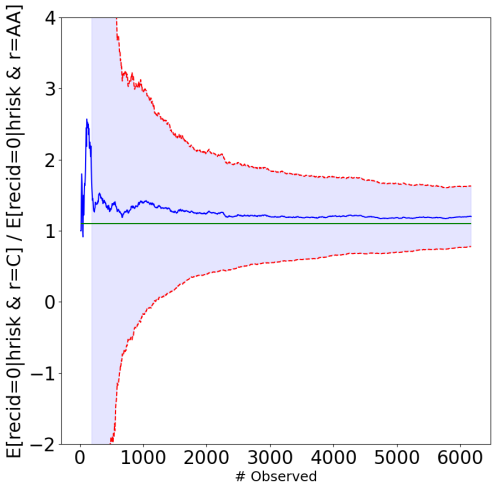
E[hrisk | race=Caucasian & recid=0] /
E[hrisk | race=African−American & recid=0] > 0.9

We find that the specification is guaranteed to be violated with a confidence of over $1 - \Delta = 0.9$ probability, and AVOIR can detect this violation within about half the number of available assessments (3350 steps) when run in an online setting. Figure 3.6a demonstrates the progression of the tracked expectation term. Thus, if deployed with the corrected specification, AVOIR would be able to alert Northpointe/an auditor of a violation/potentially-biased decision-making tool.

The Northpointe report (Dieterich et al., 2016) makes several claims about the shortcomings of this analysis. One of the primary claims is that Angwin et al. (2016) used an analysis based on "Model Errors" rather than "Target Population Errors". In fairness specification terms, this refers to the difference between a False Positive Rate (FPR) balance vs. False



(a) (ProPublica) COMPAS, "Sample A" False Positive Rate Bias specification required to *above* the 10% $\implies$ 0.9 threshold converges to a value that can be guaranteed to be *under* the required threshold.



(b) (Northpointe) "Sample B" analysis done by Northpointe using False Discovery Rate that opposed the ProPublica reports.

Figure 3.6: COMPAS dataset case study.

Discovery Rate (FDR) balance, i.e., balancing for predictive parity over predictive equality. In probabilistic terms, the difference amounts to comparing $\Pr[\hat{Y} = 1|Y = 0, g = 1, 2]$ (FPR) vs $\Pr[Y = 0|\hat{Y} = 1, g = 1, 2]$ (FDR), where $\hat{Y}$ refers to the decision made by the algorithm, Y refers to the true value, and $g = 1, 2$ reflects group membership (Verma and Rubin, 2018). This analysis is run on the dataset subset dubbed "Sample B". To test their hypothesis, we reproduce the corresponding preprocessing steps and run both versions (numerator and denominator being Caucasian) of the corresponding specification under the same setup as earlier. Despite the point estimate being within the required threshold, we find that neither version can be guaranteed with the required confidence in the given data. Due to the paucity of space, we describe only one of the two variants with the corresponding figure (Figure 3.6b).

E[recid=0 | race=Caucasian & hrisk] /
E[recid=0 | race=African−American & hrisk] > 0.9

We note that the Northpointe report (Dieterich et al., 2016) does not provide confidence intervals for their claim. Further, even though the report does not release associated code, the point estimates of the False Discovery Rates (FDRs) match those present in the report, which increases our confidence in our AVOIR-based analysis.

The back-and-forth exchange has been the subject of much discussion in academic and journalistic publications (Feller et al., 2016; Washington, 2018). Seminal work by Kleinberg et al. (2017) proved the impossibility of simultaneously guaranteeing certain combinations of fairness metrics. While AVOIR cannot circumvent this problem, its usage can help audit claimed guarantees on defined metrics. We conclude this case study by noting that AVOIR lends itself to successful analysis that is not possible with the VF implementation available online, which only provides support for a predefined set of specifications and requires access to a data-generating function. In addition, we choose 0.1 as the failure probability because it is one of the thresholds used in (Angwin et al., 2016). We set it to the highest used threshold

to allow leeway for the claim by Northpointe. Even under this lax threshold, the analysis by Northpointe fails.

3.5 Related Work

There are a plethora of fairness criteria, and subtle changes in their definition can change the implications on decision-making (Castelnovo et al., 2021). Practitioners need support when selecting, designing, and guaranteeing fairness for deployed machine learning algorithms. Prior work on fairness has helped develop nuanced notions and algorithms to help train more 'fair' machine learning models. These include group fairness measures such as inter alia, minimizing disparate impact (Calders et al., 2009; Feldman et al., 2015), maximizing the equality of opportunity (Hardt et al., 2016) In contrast with group fairness notions, causal notions of fairness (Kusner et al., 2017) and individualized notions of fairness (Dwork et al., 2012) provide alternative statistical mechanisms for understanding discriminatory behaviors of automated decision systems. Thomas et al. (2019) proposed the Seldonian Framework as a generic mechanism for model users to design algorithms that help train machine learning models that can regulate them against undesirable behaviors. Yan and Zhang (2022) propose a query-efficient framework to audit an unknown function chosen from a known hypothesis class of decision-making functions.

We focus on the problem of detecting and diagnosing whether systems designed under any framework follow any prescribed regulatory constraints supported within the grammar of AVOIR. That is, we are agnostic to the framework; instead, we are interested in testing the adherence of models to specified criteria. We use a probabilistic framework to verify this behavior. Alternative frameworks such as the AI Fairness 360 (Bellamy et al., 2019) provide mechanisms to quantify fairness uncertainty, though they are restricted to pre-supported

metrics. Uncertainty quantification (Ghosh et al., 2021b; Ginart et al., 2022) is an alternative mechanism to provide adaptive guarantees. However, existing work is designed for commonly used outcome metrics, such as accuracy and F1-score, rather than for fairness metrics. Justicia (Ghosh et al., 2021a) optimizes uncertainty for fairness metrics estimates using stochastic SAT solvers but can only be applied to a limited class of tree-based classification algorithms.

Machine learning testing (Zhang et al., 2020) is an avenue that can expose undesired behavior and improve the trustworthiness of machine learning systems. Prior work on fairness testing is most closely related to AVOIR. Fairness testing (Galhotra et al., 2017) provides a notion of causal fairness and generates tests to check the fairness of a given decision-making procedure. Given a specific definition of fairness, Fairtest (Tramèr et al., 2017) and Verifair (VF) (Bastani et al., 2019) build a comprehensive framework for investigating fairness in data-driven pipelines. Fairness-aware Programming (FP) (Albarghouthi and Vinitsky, 2019) combined the two demands of machine learning testing and fairness auditing to make fairness a first-class concern in programming. Fairness-aware programming applies a runtime monitoring system for a decision-making procedure with respect to an initially stated fairness specification. The overall failure probability of an assertion is computed as the sum of the failure probabilities of each constituting sub-expression (using the union bound). FP does not provide any specific mechanism for splitting uncertainty, and Verifair splits it equally across all constituent *elementary subexpressions*. Thus, assertion bounds for subexpressions in both FP and VF are split inefficiently compared to AVOIR.

3.6 Conclusion

We presented the AVOIR framework to easily define and monitor fairness specifications online and aid in the refinement of specifications. AVOIR is easy to integrate within modern database systems but can also serve as a standalone system evaluating whether black-box machine learning models meet specific fairness criteria on specific datasets (including both structured and unstructured data) as described in our case studies. AVOIR extends the grammar from Fairness Aware Programming (Albarghouthi and Vinitsky, 2019) with operations that enhance expressiveness. In addition, we derive probabilistic guarantees that improve the confidence with which specification violations are reported. Through case studies, we demonstrate that AVOIR can provide users with insights and context that contribute directly to refinement decisions. To understand the robustness of AVOIR, we evaluated it along two dimensions: the data/ML model used and changing parameters (thresholds, fairness definitions). We demonstrated the robustness of the data/model used by evaluating three datasets of varying domains and types (criminal justice - COMPAS, text classification - RateMyProfs, census data - Adult Income). For robustness to the thresholds, we used varying failure probability levels (0.05, 0.1, 0.15) in our case studies. Note that any probability thresholds over these values for the corresponding studies would converge in fewer iterations, while lower thresholds would require additional data samples. Our framework builds the foundation for further improvements in fairness specification, auditing, and verification workflows. Although contextual information from AVOIR makes decisions more straightforward, it is not always clear how to alter a specification in light of a violation and its relevant context.

To assist in these decisions, we are currently examining mechanisms that suggest edits that are likely to achieve the desired intent of a model developer. We plan to extend this work

to provide intelligent specification refinement suggestions and support distributed machine learning settings. In addition to improving the usability of our tools for making fairness specification refinements, we also envision a more scalable framework. Our case studies looked at a single model with respect to a single dataset. However, real-world deployment of machine learning often contains many clients with models and datasets that may evolve and drift over time. We also expect to examine efficient monitoring of machine learning behavior for a fairness specification in a distributed context, enabling horizontal scalability. We believe techniques such as decoupling the observation of data and reporting results from monitoring the results are promising and can lead to the desired scalability.

Appendix

3.A Inference Rules

In Figure 3.A.1, we provide the rules used to determining the constraints and guarantees for a specification. We represent $X \odot Y : (E, \epsilon, \delta) \equiv \Pr\left(|\mathbb{E}[X] \odot \mathbb{E}[Y] - E| \geq \epsilon\right) \leq \delta$ where $\odot$ represents a binary operator. Constraints are represented in $\{\}$. The proof of correctness for each inference rule starts from the assumptions above the horizontal line and derives the assertions below. These proofs use ideas similar to those in (Bastani et al., 2019). We reproduce the proofs in Appendix 3.A.1 here for completeness. Note that the assertions in the base case (elementary subexpressions) can be arrived at by applying AIN.

3.A.1 Inference rules with Constraints

In Section 3.3.3 we provided the proofs for $X \pm Y$, $X > c$. In the following text, we provide the remaining proofs.

Product Starting with ϕ_X, ϕ_Y First, from union bound, both of these hold true with probability at least $1 - \delta_X - \delta_Y$. Then,

$$\begin{aligned} |\mathbb{E}[X]| &= |\overline{\mathbb{E}}[X] - \overline{\mathbb{E}}[X] + \mathbb{E}[X]| \\ &\leq ||\overline{\mathbb{E}}[X]| + |\overline{\mathbb{E}}[X] + \mathbb{E}[X]|| \leq ||\overline{\mathbb{E}}[X]| + \epsilon_X \end{aligned}$$

$$\frac{X : \left(\overline{\mathbb{E}}[X], \epsilon_X, \delta_X\right), Y : \left(\overline{\mathbb{E}}[Y], \epsilon_Y, \delta_Y\right)}{X \pm Y : \left(\overline{\mathbb{E}}[X] \pm \overline{\mathbb{E}}[Y], \epsilon_X + \epsilon_Y, \delta_X + \delta_Y\right)}$$

$$\frac{X : \left(\overline{\mathbb{E}}[X], \epsilon_X, \delta_X\right), Y : \left(\overline{\mathbb{E}}[Y], \epsilon_Y, \delta_Y\right)}{X \times Y : (\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y], \epsilon_X\epsilon_Y + \overline{\mathbb{E}}[X]\epsilon_Y + \overline{\mathbb{E}}[Y]\epsilon_X, \delta_X + \delta_Y)}$$

$$\frac{X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right), \overline{\mathbb{E}} - \epsilon > 0}{X^{-1} : \left(\overline{\mathbb{E}}^{-1}, \frac{\epsilon}{\overline{\mathbb{E}}(\overline{\mathbb{E}} - \epsilon)}, \delta\right)} \text{ (Inverse)}$$

$$\frac{X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right)}{X^{-1} : \left(\overline{\mathbb{E}}^{-1}, \frac{\epsilon}{\overline{\mathbb{E}}(\overline{\mathbb{E}} - \epsilon)}, \delta\right), \{\overline{\mathbb{E}} - \epsilon > 0\}} \text{ (Inverse C)}$$

$$\frac{X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right), \overline{\mathbb{E}} - \epsilon > c}{X > c : (T, \delta)} \text{ (True)} \quad \frac{X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right), \overline{\mathbb{E}} + \epsilon < c}{X < c : (F, \delta)} \text{ (False)}$$

$$\frac{X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right)}{X > c : (T, \delta), \{\overline{\mathbb{E}} - \epsilon > c\}} \text{ (True C)}$$

$$\frac{X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right)}{X < c : (T, \delta), \{\overline{\mathbb{E}} + \epsilon < c\}} \text{ (False C)}$$

$$\frac{\psi_1 : (\mathbb{B}_1, \delta_1), \psi_2 : (\mathbb{B}_2, \delta_2)}{\psi_1 \wedge \psi_2 : (\mathbb{B}_1 \wedge \mathbb{B}_2, \delta_1 + \delta_2)} \text{ (and)} \frac{\psi_1 : (\mathbb{B}_1, \delta_1), \psi_2 : (\mathbb{B}_2, \delta_2)}{\psi_1 \vee \psi_2 : (\mathbb{B}_1 \vee \mathbb{B}_2, \delta_1 + \delta_2)} \text{ (or)}$$

$$\frac{\psi_1 : (\mathbb{B}_1, \delta_1), \{C_{11,\ldots,1k}\}, \psi_2 : (\mathbb{B}_2, \delta_2), \{C_{21,\ldots,2m}\}}{\psi_1 \wedge \psi_2 : (\mathbb{B}_1 \wedge \mathbb{B}_2, \delta_1 + \delta_2), \{C_{11,\ldots,1k}, C_{21,\ldots,2m}\}} \text{ (and C)}$$

$$\frac{\psi_1 : (\mathbb{B}_1, \delta_1), \{C_{11,\ldots,1k}\}, \psi_2 : (\mathbb{B}_2, \delta_2)}{\psi_1 \vee \psi_2 : (\mathbb{B}_1 \vee \mathbb{B}_2, \delta_1 + \delta_2), \{C_{11,\ldots,1k}\} \vee \{C_{21,\ldots,2m}\}} \text{ (or C)}$$

Figure 3.A.1: Inference rules used to guarantees for expressions.The inference rules for each compound expression build on the union bound, triangle inequality, and structural induction approach described by Bastani et al. (2019). C: Constraint.

$$
\begin{aligned}
|\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y] - \mathbb{E}[XY]| &= |\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y] - \mathbb{E}[X]\mathbb{E}[Y]| \\
&= |\overline{\mathbb{E}}[X](\overline{\mathbb{E}}[Y] - \mathbb{E}[Y]) + \mathbb{E}[Y](\overline{\mathbb{E}}[X] \ - \mathbb{E}[X])| \\
&\leq |\overline{\mathbb{E}}[X]||(\overline{\mathbb{E}}[Y] - \mathbb{E}[Y])| + |\mathbb{E}[Y]||(\overline{\mathbb{E}}[X] \ - \mathbb{E}[X])| \\
&\leq |\overline{\mathbb{E}}[X]|\varepsilon_Y + |\mathbb{E}[Y]|\varepsilon_X \\
&\leq |\overline{\mathbb{E}}[X]|\varepsilon_Y + (|\overline{\mathbb{E}}[Y]| + \varepsilon_Y)\varepsilon_X \\
&= |\overline{\mathbb{E}}[X]|\varepsilon_Y + |\overline{\mathbb{E}}[Y]|\varepsilon_X + \varepsilon_X\varepsilon_Y
\end{aligned}
$$

where the first step follows as X, Y are Bernoulli r.vs. Therefore, $X \times Y : (\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y], \epsilon_X\epsilon_Y + \overline{\mathbb{E}}[X]\epsilon_Y + \overline{\mathbb{E}}[Y]\epsilon_X, \delta_X + \delta_Y)$

Inverse/Inverse C Assume $X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right)$ and $\overline{\mathbb{E}} - \epsilon > 0$. In the constrained case, we start with only the prior assumption. Then,

$$
\begin{aligned}
|\mathbb{E}[X]| &= |\mathbb{E}[X] - \overline{\mathbb{E}}[X] + \overline{\mathbb{E}}[X]| \\
&\leq |\mathbb{E}[X] - \overline{\mathbb{E}}[X]| + |\overline{\mathbb{E}}[X]| \leq \epsilon_X + |\overline{\mathbb{E}}[X]|
\end{aligned}
$$

i.e., $|\mathbb{E}[X]| \leq \epsilon_X + |\overline{\mathbb{E}}[X]|$. Also,

$$
\begin{aligned}
|\mathbb{E}[X]^{-1} - \overline{\mathbb{E}}[X]^{-1}| &= \left|\frac{\overline{\mathbb{E}}[X]^{-1} - \mathbb{E}[X]^{-1}}{\overline{\mathbb{E}}[X]\mathbb{E}[X]^{-1}}\right| \\
&\leq \frac{\epsilon}{|\mathbb{E}[X]||\overline{\mathbb{E}}[X]|} \leq \frac{\epsilon}{|\mathbb{E}[X]|(\mathbb{E}[X] - \epsilon_X)}
\end{aligned}
$$

VF adds $E[X] - \epsilon_X > 0$ as a precondition; AVOIR as a post-constraint.

3.A.1.0.1 Boolean Operators Starting from $\psi_1 : (b_1, \delta_1)$, $\psi_2 : (b_2, \delta_2)$, we can apply the union bound for $\psi_1 \wedge \psi_2$, $\psi_1 \vee \psi_2$ to derive the rules for and/or. Similarly, constraints follow the semantics specified by the rules as they also follow from the union bound.

3.A.2 Inferred Optimization Problem

For a given overall specification ψ, suppose (ϵ_i, δ_i), $i \in \{1, \ldots, n\}$ represents the concentration bounds associated with each constituent elementary subexpression. Using the inference rules, we can derive the overall $\delta_T = \sum_i \delta_i$, along with a set of (say) K constraints

$$g_k(\epsilon_1, \ldots, \epsilon_n, \overline{\mathbb{E}}[X_1], \ldots, \overline{\mathbb{E}}[X_n]) \leq \epsilon_k$$

$$\text{where } \epsilon_k = \left| c_k - \overline{\mathbb{E}}[f(\overline{\mathbb{E}}[X_1], \ldots, \overline{\mathbb{E}}[X_n])] \right|$$

denotes the maximum allowed margin for the k$^{\text{th}}$ subexpression of form `<ETerm> <comp-op> c`). The objective is to minimize the overall failure probability δ_T. The overall optimization problem can then be formulated as shown in 3.1, having n optimization variables δ_i and $2n + K$ constraints (bounds on δ_i provide the $2n$ constraints). A developer using AVOIR inputs a required acceptable upper bound of failure probability Δ. If the solution to the optimization problem $\delta_T^* = \sum_i \delta_i \leq \Delta$, then the optimization can conclude with the required confidence in the proved guarantee. At this point, the developer may choose to terminate AVOIR. However, using Corollary 4.1, they may continue to run and refine the estimates.

3.B Concentration bounds

Theorem 1 provides a mechanism for choosing the stopping time using arbitrary methods for a fixed δ. In general, any adaptive concentration inequality suffices; we use AIN_{H} However, we use confidence intervals to visualize the evolution of sub-expressions (and overall specification) over the sequence of observations. To do so, we require an additional result.

Theorem 4. *(Zhao et al., 2016, Proposition 1, Lemma 1) Let $S_n = \sum_{i=1}^n X_i$ be a random walk from i.i.d. random variables $X_1, \ldots, X_t \sim D$. For any $\delta > 0$, $\Pr[S_{\mathcal{T}} \geq f(\mathcal{T})] \leq \delta$ for any stopping time $\mathcal{T}$ if and only if $\Pr[\exists n, S_t \geq f(t)] \leq \delta$*

Corollary 4.1. *For $\delta > 0$, $\Pr[|\overline{\mathbb{E}}_{\mathcal{T}}[X] - \mathbb{E}[X]| \leq \epsilon(\delta, \mathcal{T})|] \geq 1 - \delta$ for any stopping time $\mathcal{T}$ if and only if*

$$\Pr\left[\forall t, |\overline{\mathbb{E}}_t[X] - \mathbb{E}[X]| \leq \epsilon(\delta, t)|\right] \geq 1 - \delta$$

Corollary 4 follows directly from applying Theorem 4 to Theorem 1. Intuitively, Theorem 1 holds since we can choose an adversarial stopping rule for $\mathcal{T}$ that terminates as soon as the boundary for $\epsilon(\delta, t)$ is crossed (Zhao et al., 2016). Thus, when we establish a bound with a stopping rule, the bound will hold prior to and after the stopping rule is enforced. Corollary 4.1 implies that once we choose an optimal bound for each subexpression, we can extend the bounds derived using Theorem 1 to following observations with continued guarantees for subexpressions.

3.B.1 Proof of Theorem 2 for Specifications

Consider any specification ψ_k. Let $\psi_k^t : (\hat{b}_{\psi_k}(t), \delta_{\psi_k}(t))$, where $\hat{b}_{\psi_k}(t) \subseteq \{T, F\}$ is the inferred value and $\delta_{\psi_k}(t)$ corresponds to the confidence for the assertion at time t. Let the *elementary* subexpressions involved be $X_{k_1}, \ldots, X_{k_D}$ corresponding to the index multiset $\mathcal{B}_k = \{\{k_1, \ldots, k_D\}\}$. Denote b_{ψ_k} as the true value of ψ_k, and δ_{ψ_k} as the inferred threshold at stopping time $\mathcal{T}$. From INFER, we have

$$\hat{b}_k(t), \delta_{\psi_k}(t) = \text{INFER}(\phi^{\mathrm{t}}_{\mathrm{X}_{\mathrm{k}_1}}, \ldots, \phi^{\mathrm{t}}_{\mathrm{X}_{\mathrm{k}_D}}) \tag{3.9}$$

$$
\begin{aligned}
&\Pr[\exists t \geq 1, b_k \notin \hat{b}_k(T)] \\
&\quad \leq \Pr\left[\bigcup_{i=1}^{D} \exists t \geq 1, \neg\phi^t_{X_{k_i}}\right] && \text{(From 3.9)} \\
&\quad \leq \sum_{i \in B_k} \Pr\left[\exists t \geq 1, \neg\phi^t_{X_{k_i}}\right] && \text{(union bound)} \\
&\quad = \sum_{i \in B_j} \Pr\left[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_{k_i}] - \mathbb{E}_t[X_{k_i}| > \epsilon_{X_{k_i}}(t)\right] \\
&\quad \leq \sum_{i \in B_j} \delta_{X_{k_i}} && \text{(elementary subexpressions)} \\
&\quad \leq \delta_{\psi_k} && \text{(applying 3.8 for } t = \mathcal{T})
\end{aligned}
$$

Thus, $b_{\psi_k}(t)$ is a $1 - \delta_{\psi_k}$ confidence sequence for b_{ψ_k}

3.C Termination Criterion for AVOIR

Corollary 3.2. *Under mild conditions, AVOIR terminates in finite steps with an assertion over the required specification.*

Proof. We know that the stopping time $\mathcal{T} \leq \mathcal{T}^+$, the stopping time for AVOIR. Thus, AVOIR would terminate whenever Verifiar can. For completeness, we provide the conditions under which Verifair terminates. Note that $c \in \mathbb{R}$ corresponds to a constant threshold involved in specification, also presented in the grammar and bound proagation rules.

- For every subexpression C_k occurring in the specification such that it is involved in the inverse or inverse constr. rules (i.e., $\overline{\mathbb{E}}[C_k]^{-1}$), $\overline{\mathbb{E}}[C_k] \neq 0$, $C_k \neq 0$
- For every subexpression C_k such that it occurs a True/False type inequality (such as $C_k > c$), $\overline{\mathbb{E}}[C_k] \neq c$, $C_k \neq c$

□

Metric Name	Definition/DSL
Statistical Parity (Dwork et al., 2012)	$\Pr[R\|S] = \Pr[R\|\neg S]$
	$\mathbb{E}[r\|s]/\mathbb{E}[r\|!s] < c$
Predictive Parity (Chouldechova, 2017)	$\Pr[Y\|R,S] = \Pr[Y\|R,S]$
	$\mathbb{E}[y\|r,s] - \mathbb{E}[y\|r,s] > c$
Equal Opportunity (Hardt et al., 2016)	$\Pr[\neg R\|Y,S] = \Pr[\neg R\|Y,\neg S]$
	$\mathbb{E}[!r\|y,s] - \mathbb{E}[!r\|y,!s] < c$
Equalized Odds (Hardt et al., 2016)	$\Pr[R\|Y=i,S] = \Pr[R]Y=i,\neg S], i=0,1$
	$(\mathbb{E}[r\|y=0,!s] - \mathbb{E}[r\|y=0,s] > c_0)$ & $(\mathbb{E}[r\|y=1,!s] - \mathbb{E}[r\|y=1,s] > c_1)$

Table 3.D.1: Examples of supported metrics.

3.D Supported Metrics

We provide a non-exhaustive list of statistical group-based fairness criteria and show an exact/approximate equivalent in the AVOIR DSL in Table 3.D.1. We use the notation from Table 3.1, assuming that the return value R is a Bernoulli r.v. We assume that the decision function f tracked by AVOIR as a signature that takes X, G, Y as input and produces S or d as output. Note that in their python implementation, $=$ would be replaced by $==$ and $|$ by the given keyword.

3.E Implementation Details

We built a python library to create specifications that can be implemented as a decorator over decision functions. The front end interactive application was implemented using streamlit[12] and the visualizations were built in Vega (Satyanarayan et al., 2015). Each term in the DSL is implemented through a corresponding python class. New input/output observations are monitored to update all the terms in a specification. Inference for evaluating

[12]https://streamlit.io/

the value and bounds is carried out via operator overloading in these classes. In line with previous work (Albarghouthi et al., 2017; Bastani et al., 2019; Albarghouthi and Vinitsky, 2019) on distributional verification, we use rejection sampling for conditional probability estimation.

3.E.1 Visual Analysis

Using our specification framework as a backend, we built an interactive application for analysis and refinement of specifications provided in our grammar. Given a user provided machine learning model, dataset, and specification the application simulates a stream of observations to the provided model. Following the simulation, a visualization is provided that represents the specification as a syntax tree where each node of the tree corresponds to an element of our grammar. Figure 3.E.1 shows the visualization.

Note that for each observation made by our machine learning model, the specification is evaluated to check for violations. Each grammar element that makes up the specification is evaluated as well, and thus each grammar element is associated with the value it evaluates to for a given observation. For specifications `<spec>`, there is a boolean value associated with each observation, whereas an expectation term, `<ETerm>`, is associated with a real value. By selecting one of the nodes in the syntax tree, a user can see a plot of the evaluation values associated with the selected grammar element. We call these plots evaluation plots and two can be observed at a time each with shared scales along the horizontal axis which denotes observations over time. This allows for comparison of multiple grammar elements. The ability to analyze and compare these evaluation values provides context surrounding specification violations, and assists the user in deciding how to refine a specification. The case studies in section 3.4 demonstrate the usefulness of the context provided by these visualizations.

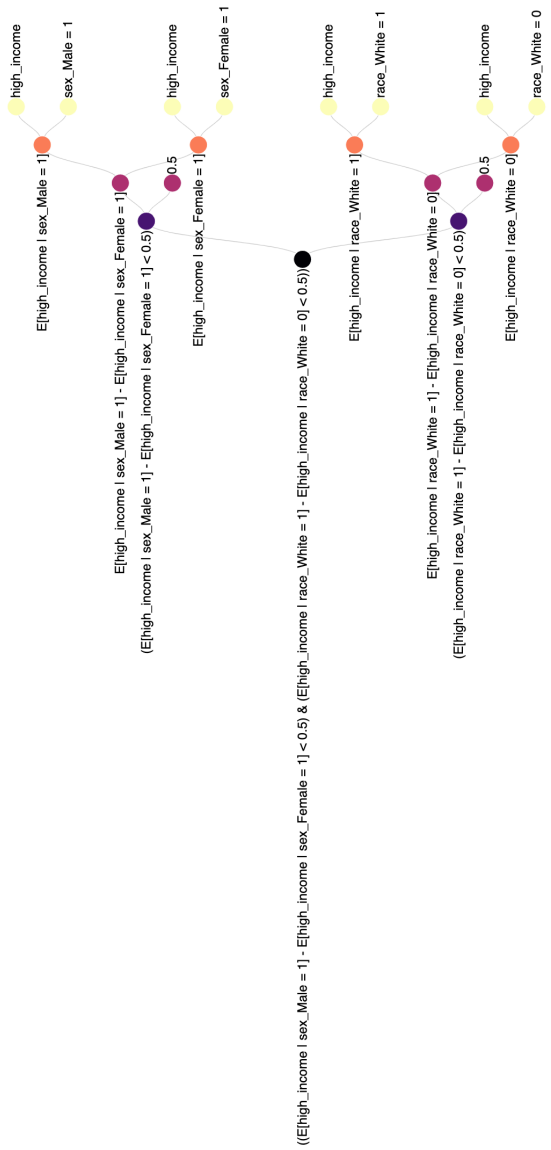


Figure 3.E.1: Tree corresponding to the initial specification for the Adult Income dataset.

The app for interaction with the backend is built using streamlit. It proceeds in multiple stages,

1. First, a user selects a dataset of interest. We built support for four datasets, but our framework is generic enough for any arbitrary csv dataset.

2. Following this choice, the input variables and output variable for a machine learning model must be specified.

3. A machine learing model is then selected from a dropdown. We provide support for three models. However, this is for demonstration purposes only - the specification is agnostic to the choice of a machine learning model.

4. Finally, a specification is input by the user of the app. On the press of a button, the model is trained and then evaluated on the selected dataset. The output monitored by the spec is passed off to the Vega module for further analysis.

3.F AVOIR in Database Setting

In the database literature researchers (Nargesian et al., 2021), have explored an approach to tailoring data integration strategies to ensure that the data set used for analysis has an appropriate representation of relevant (demographic) groups and it meets desired distribution requirements. The authors describe how to acquire such data in an approximate cost-optimal manner for several realistic settings. This work is orthogonal to our work and yet AVOIR can potentially integrate with the authors approach to examine if fairness criteria are being met during the integration process. In other studies on fairness researchers (Yang et al., 2018; Asudeh et al., 2019; Sun et al., 2019), have considered the problem of personalized fair ranking functions and discuss approaches to determine if a proposed ranking function

satisfies a set of desired fairness criteria and, if it does not, to suggest modifications that do. AVOIR attempts to solve a more general purpose problem (not limited to any particular fairness criteria) and is agnostic to the specific model (treats it as a blackbox). While we have not examined the performance of AVOIR for fair ranking problems, it is something we plan to examine in the future.

To demonstrate how AVOIR can be integrated within a database system we use pandas[13] dataframes to simulate the application of AVOIR in the database setting. Specifically, we wrap pandas dataframes with a python 'Database' class, and provide a query mechanism to create materialized views. Queries are provided in the form of python functions that take a dataframe as input and output a corresponding dataframe. The corresponding view thus generated can be updated with insertion/update/deletion of data. The specification is added as a decorator inside the refresh function, allowing AVOIR to track specifications in a database setting. Note that this tie-in with pandas is only for ease of implementation; the inference engine and optimization can be extended to any database engine.

[13]https://pandas.pydata.org/

Chapter 4: Conformal Prediction for Graph Structured Data

The previous chapter used confidence intervals to generate online, anytime-valid bounds for the outputs associated with different decision-making models. However, confidence intervals require a strong assumption about the data-generating distribution at test time, i.e., independent and identically distributed (IID) data. For graph-structured data, the edges between different nodes denote potential dependencies between the nodes. Thus, the IID assumption is violated, and the corresponding confidence intervals are no longer a viable option. *Conformal prediction* is a method that provides valid confidence sets/intervals for graph-structured data under a weaker assumption - exchangeability. While the estimates generated by conformal prediction are no longer online or anytime-valid, the associated guarantees can provide a foundation for understanding the uncertainty associated with the predictions in graph-structured data. This chapter will discuss the theoretical underpinnings of conformal prediction and the tradeoffs associated with its application to graph-structured data.

Conformal prediction has become increasingly popular for quantifying the uncertainty associated with machine learning models. Compared to confidence intervals, conformal prediction provides distribution-free valid coverage under exchangeability with finite sample guarantees and can be computed efficiently. The computational efficiency of the split conformal prediction approach has made it the method du jour for exploring new approaches

for quantifying the uncertainty of predictions from large, computationally expensive machine learning models. Recent work in graph uncertainty quantification has built upon this approach for conformal prediction on graphs. The nascent nature of these explorations has led to conflicting choices for implementations, baselines, and evaluation of approaches. We critically analyze the choices made and describe the tradeoffs associated with existing graph conformal prediction work. Our theoretical and empirical results provide the rationale for our recommendations for future scholarship in graph conformal prediction.

4.1 Introduction

Modern machine learning models which are trained on losses based on point predictions are prone to being overconfident in their predictions (Guo et al., 2017). The Conformal Prediction (CP) framework (Vovk et al., 2005) provides a mechanism for generating statistically sound post-hoc prediction sets (or intervals, in case of continuous outcomes) with coverage guarantees under mild assumptions. The usual assumption made in CP is that data are exchangeable, i.e, the joint distribution of the data is invariant to permutations of the data points.

Definition 3. *A sequence of random variables $X_1, \ldots, X_n$ is said to be exchangeable if for any permutation σ of the natural numbers, $(X_1, \ldots, X_n) \stackrel{d}{=} (X_{\sigma(1)}, \ldots, X_{\sigma(n)})$, where $\stackrel{d}{=}$ denotes equality in distribution.*

The guarantees provided by CP are distribution-free, and can be added post-hoc to the scores produced by arbitrary, black-box predictors. This makes CP an ideal candidate for quantifying uncertainty in complex models such as neural networks. Variations of CP include full CP (Vovk et al., 2005), cross-conformal prediction (Vovk, 2015), split CP (Vovk et al., 2005), and the CV+/Jackknife+ approach (Barber et al., 2021). Full CP has a significant computational cost as an expensive conformity score function must be computed with

replacement for each data point within the calibration set. Cross-conformal prediction (Vovk, 2015) and CV+/Jackknife+ (Barber et al., 2021) are other variations of CP which are computationally more efficient than full CP. Split CP, using a fixed conformity score function of a single data point at a time, is the most popular variation of CP. In addition to its computational efficiency, the ease of implementation, and distribution-free guarantees with black-box models make split CP a popular choice.

Network-structured data such as social networks, transportation networks, and biological networks are ubiquitous in modern data science applications. Graph Neural Networks (GNNs) have been developed to model vector representations of such network-structured data, and have been shown to be effective in a variety of tasks such as node classification, link prediction, and graph classification (Hamilton, 2020; Wu et al., 2022). Uncertainty quantification approaches built for independent and identically distributed (IID) data cannot directly be applied to graph data as the network structure introduces dependencies between the data points. However, recent work (Clarkson, 2023; Zargarbashi et al., 2023; Huang et al., 2023) has demonstrated that in certain settings, CP can be applied to graph data to generate statistically sound prediction sets which provide a coverage guarantee for the node classification task. In line with prior work in CP on graphs, we focus on split CP in this work. There is a lack of consensus for the choice and setup of baselines, splitting of common datasets, and evaluation metrics for methods. In this work, we aim to analyze the choices made by existing work and understand the tradeoffs associated with these choices.

4.2 Conformal Prediction

Conformal prediction is used to quantify the uncertainty of a model by providing prediction sets/intervals with coverage guarantees. We will focus on conformal prediction in the

classification setting. The dataset is partitioned as $\mathcal{D} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{valid}} \cup \mathcal{D}_{\text{calib}} \cup \mathcal{D}_{\text{test}}$. Given a calibration dataset $\mathcal{D}_{\text{calib}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$, where $\boldsymbol{x}_i \in \mathcal{X} = \mathbb{R}^d$ and $y_i \in \mathcal{Y} = \{1, \ldots, K\}$, conformal prediction can be used to construct a prediction set C such that

$$\Pr\left[y_{n+1} \in C(\boldsymbol{x}_{n+1})\right] \geq 1 - \alpha$$

where $1 - \alpha \in [0, 1]$ is a user-specified coverage level. The only assumption required for the coverage guarantee is that $\mathcal{D}_{\text{calib}} \cup \{(\boldsymbol{x}_{n+1}, y_{n+1})\}$ is exchangeable. The following theorem provides a general recipe for constructing a prediction set with coverage guarantee.

Theorem 4 (Vovk et al. (2005))**.** *Suppose* $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n+1}$ *are exchangeable,* $s : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ *is a score function measuring the non-conformity of* $(\boldsymbol{x}, y)$*, with higher scores indicating lower conformity, and a target* $\alpha \in [0, 1]$*. Let* $\hat{q}(\alpha) = \textit{Quantile}\left(\frac{\lceil (n+1)(1-\alpha) \rceil}{n}; \{s(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}\right)$ *Define* $C_\alpha(X) = \{y \in \mathcal{Y} : s(\boldsymbol{x}, y) \leq \hat{q}(\alpha)\}$*. Then,*

$$1 - \alpha + \frac{1}{n+1} \geq \Pr\left[y_{n+1} \in C_\alpha(\boldsymbol{x}_{n+1})\right] \geq 1 - \alpha \tag{4.1}$$

s is usually called the non-conformity score function and measures the degree of non-agreement between the input $\boldsymbol{x}$ and the label y, given exchangeability with the calibration data $\mathcal{D}_{\text{calib}}$ i.e., larger scores indicate worse agreement between $\boldsymbol{x}$ and y. While Theorem 4 does not place any restrictions on the choice of the score function, this choice can have a significant impact on the size of the prediction set. Note that the setup of theorem 4 is called split CP, as the score function remains fixed for the calibration split. In other versions of CP, the score function is usually more expensive as it maps $\mathcal{X}^k \times \mathcal{Y}^k \to \mathbb{R}$, for some $k \in \mathbb{N}$ which varies between n for full conformal prediction and smaller values for cross-conformal prediction and CV+/Jackknife+.

4.3 Node Classification and Conformal Prediction in Graphs

The usual tasks of interest in graph data are node classification, link prediction, and graph classification. In this work, we focus on node classification and its extensions to conformal prediction. Consider an attributed homogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of edges and $\boldsymbol{X}$ is the set of node attributes. Let $\boldsymbol{A}$ denote the adjacency matrix for the graph. Further, let $\mathcal{Y} = \{1, \ldots, K\}$ denote set of class labels associated with the nodes. For $v \in \mathcal{V}$, $\boldsymbol{x}_v \in \mathbb{R}^d$ denotes its features and $y_v \in \{1, \ldots, K\}$ denotes the corresponding class label. The task of node classification is to learn a model $F : \mathcal{X} \to Y$ which predicts the label for each node given node features as input. Corresponding to the CP partitions, we denote the nodes in the training set as $\mathcal{V}_{\text{train}}$, validation set as $\mathcal{V}_{\text{valid}}$, calibration set as $\mathcal{V}_{\text{calib}}$, and test set as $\mathcal{V}_{\text{test}}$. We denote $\mathcal{V}_d = \mathcal{V}_{\text{train}} \cup \mathcal{V}_{\text{valid}}$ as the development set of the base model (non-conformalized). Note that labels are available only for nodes in the train, validation and calibration sets, and must be predicted for the test set. The model cycle will involve four phases, viz. training, validation, calibration, and testing. Next, we discuss the different settings for node classification in graphs and the applicability of conformal prediction.

Transductive setting In this setting, the model has access to the fixed graph $\mathcal{G}$ during training, validation, calibration, and testing. However, the labels associated with the test nodes $\mathcal{D}_{\text{test}}$ are unknown. We designate a fixed set of nodes disjoint from the training and validation set as $\mathcal{V}_{\text{test}} \cup \mathcal{V}_{\text{calib}}$ and then randomly sample nodes from this set to form $\mathcal{V}_{\text{calib}}$ and $\mathcal{V}_{\text{test}}$. This is the setting considered in Zargarbashi et al. (2023) and Huang et al. (2023). Note that the labels for the calibration nodes are not available for training/validation of the base model, though the neighborhood information $(\mathcal{V}, \mathcal{E})$ and the features $\boldsymbol{x}_v$ and labels y_v, $v \in \mathcal{V}_d$ are available. During the calibration phase, the features and labels for the calibration

nodes, along with the neighborhood information, are used to compute the non-conformity scores. This split ensures that the base model cannot distinguish between the calibration and test nodes, and hence exchangeability holds for $v \in \mathcal{V}_{\text{calib}} \cup \mathcal{V}_{\text{test}}$.

Inductive setting We briefly describe the inductive setting and note that the exchangeability assumption will be violated in this setting (in general). The base model is provided with the graph induced by the development nodes only $(\mathcal{V}_d, \mathcal{E}_d, \boldsymbol{X}_d)$. In the calibration/test phases, the nodes arrive either one at a time or in batches. Thus, nodes arriving later in the sequence will have access to neighbors that arrived earlier, breaking the exchangeability assumption.

In line with previous work, we focus on the transductive setting. The following theorem shows that in the transductive setting, a score model trained on the calibration set will generate scores exchangeable with the test set, and thus allow the use of conformal prediction in the transductive setting.

Theorem 5 (Zargarbashi et al. (2023); Huang et al. (2023))**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{X})$ be an attributed graph, and $\mathcal{V}_{calib} \cup \mathcal{V}_{test}$ be exchangeable. Let $F : \mathcal{X}^{|V|} \to \Delta^{|V| \times K}$ be any permutation equivariant model on the graph (for instance, GNN). Define $F(G) = \Pi \in \Delta^{|V| \times K}$ be the output probability matrix for a model trained on only $\mathcal{V}_d$. Then any score function $s(v, y) = s(\Pi_v, y, \mathcal{G})$ is exchangeable for all $v \in \mathcal{V}_{calib} \cup \mathcal{V}_{test}$*

The intuition for this theorem is that if the output of the permutation equivariant function (e.g., GNN) F does not depend on the order of the nodes in the graph (for e.g. GNN output depends on the neighbors, not the order of the nodes), then the outputs of the GNN will also be exchangeable. The formal proof for this theorem is available in Zargarbashi et al. (2023); Huang et al. (2023). This theorem paves the way for using conformal prediction for transductive node classification in graphs.

For the following sections, we will assume that the base model $\hat{\pi} : \mathcal{X} \to \Delta_{\mathcal{Y}}$, where $\Delta_{\mathcal{Y}}$ is the probability simplex over the elements of $\mathcal{Y}$ and is learned using the training and validation sets $\mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{valid}}$. The calibration set $\mathcal{D}_{\text{calib}}$ is used to determine the $\hat{q}(\alpha)$ from Theorem 4 and the test set $\mathcal{D}_{\text{test}}$ is the set for which we want to compute our prediction sets. In general, the outputs $\hat{\pi}$ need not lie over a simplex; they can be in $\mathbb{R}^K$. However, this greatly simplifies the exposition for the following sections and is the standard practice in prior work.

4.4 Conformal Scores for Graphs: Choices and Trade-offs

In this section, we critically examine some decisions made in the implementations of existing graph conformal prediction work. We discuss the trade-offs associated with these choices and provide recommendations for future scholarship in graph conformal prediction.

4.4.1 Dataset Splits and Training

There are several methods of partitioning the data to generate the different partitions of the sets. Two methods which are used in other works on graph conformal prediction for classification are (1) full-split paritioning (Huang et al., 2023) and (2) label-count sample partitioning (Zargarbashi et al., 2023).

Full-Split (FS) Partitioning In this scheme, the data is split such that each subset of the partition adheres to a size constraint defined in terms of a percentage/fraction of the full node set $\mathcal{V}$. For example, in CF-GNN (Huang et al., 2023) the authors split the datasets in their experiments randomly, but adhering to a 20%/10%/35%/35% split of $\mathcal{D}_{\text{train}}/\mathcal{D}_{\text{valid}}/\mathcal{D}_{\text{calib}}/\mathcal{D}_{\text{test}}$, respectively. Note that the overall percentage of data for which we do provide labels (in either the development or calibration set) is a large proportion (65%) of the full dataset. For non-conformal score models with numerous trainable parameters, this splitting scheme

is ideal as it allows for a large amount of data to be used for the calibration model. We explore the following splitting schemes under FS partitioning: $(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{valid}}, \mathcal{D}_{\text{calib}}, \mathcal{D}_{\text{test}}) = (0.2, 0.1, 0.35, 0.35)$, $(0.2, 0.2, 0.3, 0.3)$, $(0.3, 0.1, 0.3, 0.3)$, and $(0.3, 0.2, 0.25, 0.25)$.

Label-Count (LC) Sample Partitioning In this splitting scheme, the data is split to ensure an equal number of samples for each class/label are present in the train, validation, and calibration set. The remaining nodes are then used for the test set. Such settings are used when simulating scenarios where only a small proportion of training/labeled nodes are available, such as in semi-supervised learning. Intuitively, this setting is ideal for methods that do not have many parameters to train. We explore setting the number of samples per class to 10, 20, 40, and 80. Note that we assign nodes for each class into train, validation, calibration, and test sets sequentially, so it is feasible in this setup to have some classes having no representative samples in some partitions.

4.4.2 On TPS and Adaptability

Threshold Prediction Sets (TPS) (Sadinle et al., 2019) is a simple technique for generating conformal prediction sets. The score function $s(\boldsymbol{x}, y) = 1 - \hat{\pi}(\boldsymbol{x})_y$ directly maps the probability from the base model for the correct class into a non-conformity score. The score is higher if the model has a lower probability assigned to the correct class, indicating the label is less conforming with the model. A $1 - \alpha$ (approximate) quantile creates a probability inclusion threshold for this score over the calibration set ensures coverage and can be shown to generate prediction sets with the smallest expected size (Sadinle et al., 2019). However, the TPS score has been known to undercover hard examples and overcover easy ones (Angelopoulos et al., 2021b; Zargarbashi et al., 2023) to achieve this efficiency. Here, hard/easy refers to the coverage achieved by the prediction set in relation to the prediction set size. By overcovering

easy examples, TPS can still maintain the overall coverage guarantee without having to correctly account for coverage over harder examples.

We note that this discrepancy is claimed to occur as the TPS scores are not 'adaptive' and consider only one dimension of the score for each calibration sample. However, Sadinle et al. (2019) also proposed a labelwise control version of TPS. Instead of defining a single threshold for all classes, they separately compute the threshold for each class and a corresponding α. Thus, they define classwise quantile thresholds as

$$\hat{q}(\alpha, y_j) = \text{Quantile}\left(\frac{\lceil (n+1)(1-\alpha) \rceil}{n}; \{s(\boldsymbol{x}_i, y_i)\, i = 1, \ldots, n, y_i = y_j\}\right)$$

and the corresponding prediction sets as

$$C_{\text{TPS}}(\boldsymbol{x}) = \{y \in \mathcal{Y} : s(\boldsymbol{x}, y) \leq \hat{q}(\alpha, y)\}$$

Note that this version would provide coverage for each class label, making it more 'adaptive'. The version defined by Sadinle et al. (2019) allows controlling α_y for each class label, though, for simplicity, we set $\alpha_y = \alpha$ for label-adaptability. The tradeoff here is that we have fewer calibration samples used for each quantile threshold dimension, which may lead to higher variance in the distribution of coverage (Vovk, 2012). We call this variation of TPS as TPS-Classwise, and consider it in our baselines for comparison.

4.4.3 APS and Randomized Sets

The most popular baseline in work on graph conformal prediction is adaptive prediction sets (APS). Romano et al. (2020) introduce APS by defining an optimal prediction set construction mechanism under oracle probability. Suppose we estimate a prediction function $\hat{f}$ that correctly models the oracle probability $\Pr[Y = y | X_{test} = \boldsymbol{x}] = \pi_y(\boldsymbol{x})$ for each $y \in \mathcal{Y} = \{1, \ldots, K\}$ Let $\pi_{(1)}(\boldsymbol{x}), \ldots, \pi_{(K)}(\boldsymbol{x})$ be the sorted probabilities in descending order.

For any $\tau \in [0,1]$, define the generalized conditional quantile funciton at τ as

$$L(x;\pi,\tau) = \min\left\{k \in \{1,\ldots,K\}, \sum_{j=1}^{k} \pi_{(j)}(\boldsymbol{x}) \geq \tau\right\} \tag{4.2}$$

Then the corresponding prediction set, $C_\alpha^{\text{or}}(\boldsymbol{x})$ can be constructed from the probabilities needed to reach $1-\alpha$ coverage.

$$C_\alpha^{\text{or}}(\boldsymbol{x}) = \{y \in \mathcal{Y} : \pi_y(\boldsymbol{x}) \geq \pi_{(L(\boldsymbol{x};\pi,1-\alpha))}(\boldsymbol{x})\}$$

where or indicates the usage of the oracle probability. Further, they define tighter prediction sets in a randomized fashion using an additional uniform random variable $u \sim \text{Uniform}(0,1)$ as a parameter to construct a generalized inverse. This idea draws upon the idea of uniformly most powerful tests in the Neyman-Pearson lemma for level-α sets (Neyman and Pearson, 1933). Define

$$S(\boldsymbol{x},u;\pi,\tau) = \begin{cases} \{y \in \mathcal{Y} : \pi_y(\boldsymbol{x}) > \pi_{(L(\boldsymbol{x};\pi,\tau))}(\boldsymbol{x})\} & u < V(\boldsymbol{x};\pi,\tau) \\ \{y \in \mathcal{Y} : \pi_y(\boldsymbol{x}) \geq \pi_{(L(\boldsymbol{x};\pi,\tau))}(\boldsymbol{x})\} & \text{otherwise} \end{cases} \tag{4.3}$$

i.e., the class at the $L(\boldsymbol{x};\pi,\tau)$ rank is included in the prediction set with probability $1 - V(\boldsymbol{x};\pi,\tau)$, where

$$V(\boldsymbol{x};\pi,\tau) = \frac{1}{\pi_{(L(\boldsymbol{x};\pi,\tau))}(x)}\left\{\left[\sum_{j=1}^{L(\boldsymbol{x};\pi,\tau)} \pi_{(j)}(x)\right] - \tau\right\}$$

The corresponding randomized prediction sets are $C_\alpha^{\text{or}}(\boldsymbol{x}) = S(\boldsymbol{x},U;\pi,1-\alpha)$, $U \sim U(0,1)$ Note that in general, the coverage guarantees provided in conformal prediction hold only in expectation over the randomness in $(\boldsymbol{x}_i,y_i), i = 1,\ldots,n+1$. The randomized prediction sets continue to provide the guarantee with additional randomness over u_i. To make this work for a non-oracle probability $\hat{\pi}(\boldsymbol{x})$, they define a non-conformity score A

$$A(\boldsymbol{x},y,u;\hat{\pi}) = \min\{\tau \in [0,1] : y \in S(\boldsymbol{x},u;\hat{\pi},\tau)\} \tag{4.4}$$

Assume that $\hat{\pi}$ are all distinct - for ease of defining rank. Suppose the rank of the true class amongst the sorted $\hat{\pi}$ be r_y, i.e., $\sum_{i=1}^{K} \mathbf{1}[\hat{\pi}_i(\boldsymbol{x}) \geq \hat{\pi}_y] = r_y$ Solving for τ as a function of $\hat{\pi}$ (see Appendix 4.A, for proof),

$$A(\boldsymbol{x}, y, u; \hat{\pi}) = \left[\sum_{i=1}^{r_y} \hat{\pi}_{(i)}(\boldsymbol{x})\right] - u\hat{\pi}_y \tag{4.5}$$

Instead, if a deterministic set is used to define the conformal score instead (i.e., the randomized set construction is not carried out), then we could just add the probabilities until the true class is included:

$$\tilde{A}(\boldsymbol{x}, y; \hat{\pi}) = \left[\sum_{i=1}^{r_y} \hat{\pi}_{(i)}(\boldsymbol{x})\right] \tag{4.6}$$

This version of APS still provides the same conditional coverage guarantees and has a simpler exposition as the prediction sets are constructed by greedily including the classes until the true label is included. Thus, this version is provided as the implementation in the popular monographs on conformal prediction by Angelopoulos and Bates (2021); Angelopoulos et al. (2023). However, the lack of randomization may sacrifice on the efficiency. This modification of score affects both the quantile threshold computation during the calibration phase and the prediction set during the test phase. We will now show the conditions that impact the efficiency more formally. Let

$$\hat{q}_A = \text{Quantile}\left(\frac{\lceil (n+1)(1-\alpha) \rceil}{n}; \{A(\boldsymbol{x}_i, y_i, u_i; \hat{\pi})\}_{i=1}^{n}\right)$$

and

$$\hat{q}_{\tilde{A}} = \text{Quantile}\left(\frac{\lceil (n+1)(1-\alpha) \rceil}{n}; \{\tilde{A}(\boldsymbol{x}_i, y_i; \hat{\pi})\}_{i=1}^{n}\right)$$

Define $A_i(y) := A(\boldsymbol{x}_i, y, u_i; \hat{\pi})$ and $\tilde{A}_i(y) := \tilde{A}(\boldsymbol{x}_i, y, u_i; \hat{\pi})$. From the definition of the prediction sets and non-conformity scores, we have

$$C_A(\boldsymbol{x}_{n+1}) = \{y \in \mathcal{Y} : A_{n+1}(y) \leq \hat{q}_A\}$$

and

$$C_{\tilde{A}}(\boldsymbol{x}_{n+1}) = \{y \in \mathcal{Y} : \tilde{A}_{n+1}(y) \leq \hat{q}_{\tilde{A}}\}$$

denote the prediction sets corresponding to the two score functions (with and without randomization). Define $C_A^i = C_A(\boldsymbol{x}_i)$. Let $y_i' \in \{1, 2, \ldots, K\} \setminus \{y_i\}$ be any incorrect class label for each $\boldsymbol{x}_i$. Define

$$\alpha_c^A \in [0,1], \hat{q}_A \widetilde{=} \text{Quantile}\left(\frac{\lceil (n+1)(1-\alpha_c^A)\rceil}{n}; \{A(\boldsymbol{x}_i, y_i', u_i; \hat{\pi})\}_{i=1}^n\right)$$

$$\alpha_c^{\tilde{A}} \in [0,1], \hat{q}_{\tilde{A}} \widetilde{=} \text{Quantile}\left(\frac{\lceil (n+1)(1-\alpha_c^{\tilde{A}})\rceil}{n}; \{A(\boldsymbol{x}_i, y_i', u_i; \hat{\pi})\}_{i=1}^n\right)$$

as the thresholds for which the corresponding quantile of the scores for the correct classes $A_i(y_i)$ and $\tilde{A}_i(y_i)$ achieve $1-\alpha$ coverage. Then from the exchangeability of $A(\boldsymbol{x}_i, y_i', u_i; \hat{\pi})$

$$1 - \alpha_c^A \leq \Pr[y_{n+1}' \in C_A^{n+1}] \leq 1 - \alpha_c^A + \frac{1}{n+1}$$

and similarly, from the exchangeability of $\tilde{A}(\boldsymbol{x}_i, y_i', u_i, \hat{\pi})$

$$1 - \alpha_c^{\tilde{A}} \leq \Pr[y_{n+1}' \in C_{\tilde{A}}^{n+1}] \leq 1 - \alpha_c^{\tilde{A}} + \frac{1}{n+1}$$

We will show that as long as these thresholds are sufficiently separated, the randomized prediction set will be more efficient than the non-randomized one.

Theorem 6. *Assume that $\alpha_c^A - \alpha_c^{\tilde{A}} \geq \frac{2}{n+1}$ then prediction set constructed using randomization is more efficient than without. Formally,*

$$\mathbb{E}\left[|C_{\tilde{A}}(\boldsymbol{x}_{n+1})| - |C_A(\boldsymbol{x}_{n+1})|\right] \geq 0$$

Proof. Consider the case with only two potential class labels $K = \{1, 2\}$.

We have

$$\begin{aligned}
\mathbb{E}\left[|C_A^{n+1}|\right] &= \mathbb{E}\left[\sum_{i=1,2} \mathbf{1}[i \in C_A^{n+1})]\right] \\
&= \mathbb{E}\left[\mathbf{1}[y_{n+1} \in C_A^{n+1})]\right] + \mathbb{E}\left[\mathbf{1}[y'_{n+1} \in C_A^{n+1})]\right] && \text{linearity} \\
&= \Pr[y_{n+1} \in C_A^{n+1}] + \Pr[y'_{n+1} \in C_A^{n+1}] && \mathbb{E}[\mathbf{1}[A]] = \Pr[A] \\
&\leq 1 - \alpha + 1 - \alpha_c^A + \frac{2}{n+1} && \text{(Exchangeability, Theorem 4)}
\end{aligned}$$

From a similar argument, we can show that

$$\mathbb{E}\left[|C_{\tilde{A}}^{n+1}|\right] \geq 1 - \alpha + 1 - \alpha_c^A$$

Thus,

$$\mathbb{E}\left[|C_{\tilde{A}}^{n+1}| - |C_A^{n+1}|\right] \geq 1 - \alpha + 1 - \alpha_c^{\tilde{A}} - \left(1 - \alpha + 1 - \alpha_c^A + \frac{2}{n+1}\right) \tag{4.7}$$

$$= \alpha_c^A - \alpha_c^{\tilde{A}} - \frac{2}{n+1} \tag{4.8}$$

which is equivalent to our assumption, and this completes the proof. For K classes,

$$\mathbb{E}[|C_A^{n+1}|] = \Pr[y_i \in C_A^{n+1}] + (K-1)\sum_{y'_i} \Pr[y'_i \in C_A^{n+1}]$$

Thus,

$$\begin{aligned}
\mathbb{E}[|C_A^{n+1}|] &\leq 1 - \alpha + \frac{1}{n+1} + (K-1)\left(1 - \alpha_c^A + \frac{1}{n+1}\right) \\
&= 1 - \alpha + (K-1)\left(1 - \alpha_c^A\right) + \frac{K}{n+1}
\end{aligned}$$

and

$$\mathbb{E}[|C_A^{n+1}|] \geq 1 - \alpha + (K-1)\left(1 - \alpha_c^A\right)$$

similar bounds can be derived for $\mathbb{E}[|C_{\tilde{A}}^{n+1}|]$. Thus,

$$\begin{aligned}
\mathbb{E}\left[|C_{\tilde{A}}^{n+1}| - |C_{A}^{n+1}|\right] &\geq (K-1)\left(\alpha_c^A - \alpha_c^{\tilde{A}}\right) - \frac{K}{n+1} \\
&\geq (K-1)\left(\alpha_c^A - \alpha_c^{\tilde{A}} - \frac{K}{(K-1)(n+1)}\right) \\
&> (K-1)\left(\alpha_c^A - \alpha_c^{\tilde{A}} - \frac{2}{n+1}\right) \geq 0
\end{aligned}$$

Which completes the proof in the general case.

□

Intuitively, as each score in A gets shifted by a small $u\pi$ term to the left, q_A would be lower than $q_{\tilde{A}}$. Thus, the significance levels that we would search for in the complementary scores $1 - \alpha_c^A$ would be less than $1 - \alpha_c^{\tilde{A}}$. $1 - \alpha_c^A < 1 - \alpha_c^{\tilde{A}} \implies \alpha_c^A - \alpha_c^{\tilde{A}} > 0$. If the shift is sufficiently large, then the randomized prediction set will be more efficient than the non-randomized one. In Figure 4.4.1, we show what this looks like empirially, using an example graph dataset and classifier. In the plot on the bottom, the (normalized) sorted index at which the lower threshold q_A is reached over the scores A' is lower, i.e., $1 - \alpha_c^A$ is lower, and hence α_C^A is higher. Note that the dependence on $\frac{1}{n+1}$ indicates that the improvements would be more pronounced for larger $\mathcal{D}_{\text{calib}}$.

4.4.4 Notes on Transductive NAPS

Neighborhood Adaptive Prediction Sets (NAPS) can construct predictive sets via Conformal Prediction under relaxed exchangeability (or non-exchangeability) assumptions (Barber et al., 2023). In the context of graphs, NAPS was initially implemented in the inductive setting (Clarkson, 2023). However, it can be used in the transductive setting as well (Zargarbashi et al., 2023). NAPS in the transductive setting is based on APS where $s_i = A(\boldsymbol{x}_i, y_i, u_i; \hat{\pi}_i)$, or $A(\boldsymbol{x}_i, y_i; \hat{\pi}_i)$ (depending on whether the randomized version is used), is computed for each node in $\mathcal{D}_{\text{calib}}$. Using these scores, a weighted quantile is computed to produce the score

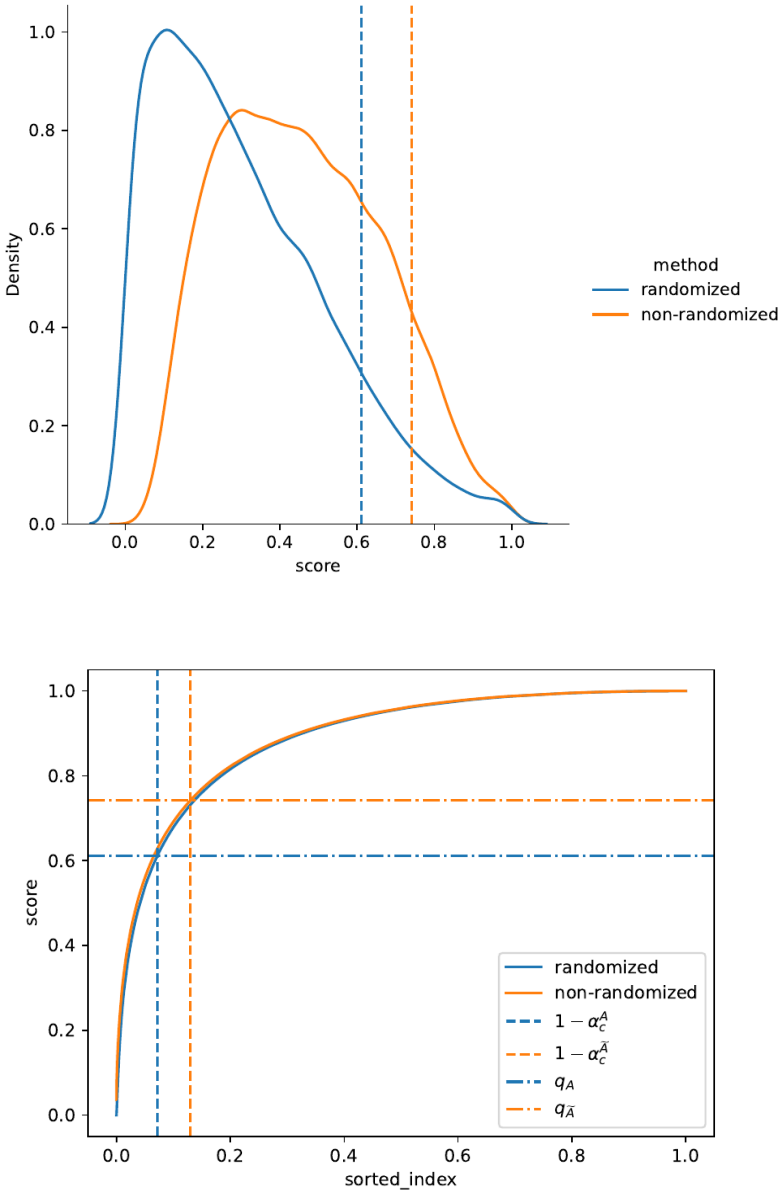


Figure 4.4.1: Figure showing the scores for an example dataset. (top) shows the shift in the quantile for A and $\tilde{A}$ for the correct class. (bottom) shows the shift α_c for A and $\tilde{A}$ using scores A' for the incorrect classes.

threshold for prediction sets (Equation 4.9) Unlike APS, the quantile is defined by placing weighted point masses (δ) at each score from the calibration set under consideration for quantile computation. The point mass at $+\infty$ indicates that the score for test node $n+1$ is unknown (and unbounded due to non-exchangeability), and thus, a point mass at the maximum value ($+\infty$) is required.

$$\hat{q}_{n+1}^{\text{NAPS}} = \text{Quantile}\left(1-\alpha, \left[\sum_{i \in \mathcal{D}_{\text{calib}}} \tilde{w}_i \cdot \delta_{s_i}\right] + \tilde{w}_{n+1} \cdot \delta_{+\infty}\right) \tag{4.9}$$

For NAPS to produce viable prediction sets, the weights, $w_i \in [0,1]$, for nodes under consideration in the calibration set must be chosen in a data independent fashion, i.e., they cannot leverage the feature vectors associated with the calibration nodes (Barber et al., 2023). NAPS leverages the graph structure to assign these weights, assigning non-zero weights to nodes within a k-hop neighborhood $\mathcal{N}_{n+1}^k$ of the test node v_{n+1}. The three implemented weight functions are uniform $w_u(d_i) = 1$, hyperbolic $w_h(d_i) = \frac{1}{d_i}$, and exponential, $w_e(d_i) = 2^{-d_i}$ for nodes in the k-hop neighborhood, where d_i is the distance from v_{n+1} to $v_i \in \mathcal{V}_{\text{calib}}$. Formally, the weight function for each node, $v_i \in \mathcal{V}_{\text{calib}}$ can be seen in Equation 4.10 below, where $w_x(d_i)$ is the selected weight function. These weights are then normalized to compute $\tilde{w}_i$ such that $\sum_{i \in \mathcal{D}_{\text{calib}}} \tilde{w}_i + \tilde{w}_{n+1} = 1$ (Barber et al., 2023).

$$w_i = \begin{cases} w_x(d_i), & i \in \mathcal{D}_{\text{calib}} \cap \mathcal{N}_{n+1}^k \\ 0, & i \in \mathcal{D}_{\text{calib}} \setminus \mathcal{N}_{n+1}^k \end{cases} \tag{4.10}$$

Using the NAPS quantile, $\hat{q}_{n+1}^{\text{NAPS}}$, the prediction sets can be constructed similarly to other Conformal Prediction algorithms. Note that NAPS was originally designed for the inductive setting; in transductive settings, fewer nodes have non-zero weights as only a subset of the graph nodes are assigned to the set of k-hop neighbors intersecting with the calibration nodes. In the inductive setting, no exchangeability cannot be assumed and the entire graph prior

to the test phase is considered as a calibration set leading to a larger number of non-zero weights.

NAPS Implementation NAPS is computationally more expensive with regard to time and memory as a k-hop intersection must be computed for each test node. We optimized this implementation using a batched approach that works with sparse tensors (Algorithm 2). Informally, the test nodes are first split up into batches. Then, for each batch, the distance to each node in the k-hop neighborhood is computed. Following this, the weights function for the corresponding nodes are computed before computing the quantile for each node. The batched approach ensures that sufficient memory is available for the necessary computations - especially for computing the distance to each node in the k-hop neighborhood without needing to densify a sparse graph.

Algorithm 2 NAPS Quantile Implementation

procedure NAPS_QUANTILE($w, k, \mathcal{D}_{\text{calib}}, \mathcal{D}_{\text{test}}, \mathcal{D}, \mathcal{S}_{\text{calib}}, b, \alpha$)
  $\{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_b\} \leftarrow$ SPLIT($\mathcal{D}_{\text{test}}, b$) ▷ Split test nodes into b batches
  $q \leftarrow$ ZEROS($\mathcal{D}_{\text{test}}, 1$) ▷ $q \in \mathbb{R}^{|\mathcal{D}_{\text{test}}| \times 1}$
  for $\mathcal{B}_n \in \{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_b\}$ **do**
    k_hop $\leftarrow$ SPARSE_K_HOP($k, \mathcal{B}_n, \mathcal{D}_{\text{calib}}, \mathcal{D}$) ▷ k_hop $\in \mathbb{R}^{|\mathcal{B}_n| \times |\mathcal{D}_{\text{calib}}|}$
    weights $\leftarrow$ COMPUTE_WEIGHTS(w, k_hop) ▷ weights $\in \mathbb{R}^{|\mathcal{B}_n| \times |\mathcal{D}_{\text{calib}}|}$
    $q[\mathcal{B}_n] \leftarrow$ COMPUTE_QUANTILE($1 - \alpha$, weights, $\mathcal{S}_{\text{calib}}$)
  end for
  return q ▷ Return the quantiles for each test node
end procedure

To ensure scalability for large graphs, all the computations until the quantile computation setep were done via sparse tensors. Algorithm 3 illustrates how the distance to each calibration node in the k-hop neighborhood can be computed via sparse tensorr primitives. The sign function based formulation uses the fact that subtracting $n + 1$-hop paths from a matrix

containing up to n hops to ensure negative values at paths of length exactly $n+1$, with the rest being 0.

Algorithm 3 Sparse K Hop Neighborhood Implementation

procedure SPARSE_K_HOP($k, \mathcal{B}, \mathcal{D}_{\text{calib}}, \mathcal{D}$)
  A $\leftarrow$ GET_ADJACENCY($\mathcal{D}$) ▷ Adjacency of $\mathcal{D}$, A $\in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$
  path_n $\leftarrow$ A[$\mathcal{B}$, :] ▷ path_n $\in \mathbb{R}^{|\mathcal{B}|\times|\mathcal{D}|}$
  k_hop $\leftarrow$ path_n[:, $\mathcal{D}_{\text{calib}}$] ▷ k_hop $\in \mathbb{R}^{|\mathcal{B}|\times|\mathcal{D}_{\text{calib}}|}$
  for $n \in \{2, 3, \ldots, k\}$ **do**
    path_n $\leftarrow$ (path_n)A
    neg_if_n $\leftarrow$ k_hop $-$ SGN(path_n[:, $\mathcal{D}_{\text{calib}}$]) ▷ negative value $\implies$ n hops away
    in_n_hop $\leftarrow$ (neg_if_n < 0) $\times\, n$ ▷ Nodes that are a min distance of n
    k_hop $\leftarrow$ k_hop + in_n_hop
  end for
  return k_hop ▷
$\forall_{i,j}$ **If** dist$(i,j) \leq k$ **then** k_hop$[i,j]$ = dist(i,j), **else** k_hop$[i,j] = 0$
end procedure

4.4.5 Diffusion Adaptive Prediction Sets

The Diffusion Adpative Prediction Sets (DAPS) approach for conformal node classification on graphs was introduced by Zargarbashi et al. (2023). The intuition behind DAPS is that the prevalence of homophily in graphs implies that the non-conformity scores for two connected scores should be related. DAPS uses a diffusion step to capture this relationship and uses the non-conformity scores modified by diffusion to generate the prediction sets. Formally, suppose $s(v, y)$ is a point wise non-conformity score for a node v and label y (e.g., TPS or APS)

$$\hat{s}(v, y) = (1 - \lambda)s(v, y) + \frac{\lambda}{|\mathcal{N}_v|} \sum_{u \in \mathcal{N}_v} s(u, y)$$

where $\mathcal{N}_v$ is the 1-hop neighborhood of v and $\lambda \in [0, 1]$ is a hyperparameter controlling the diffusion.

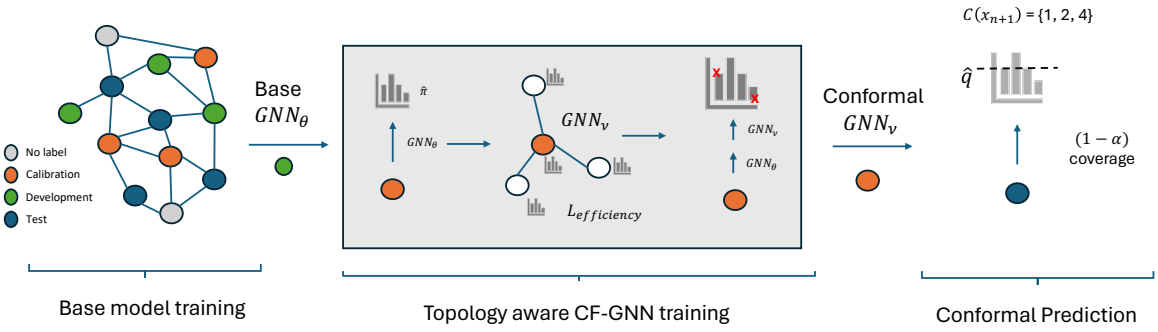


Figure 4.4.2: Procedure for training CF-GNN. First (left), the base model is trained on the training set. Then, (middle) the CF-GNN is trained to maximize efficiency over the calibration set. Finally , (right) the non-conformity scores from the combined models are used to generate the prediction sets.

Zargarbashi et al. (2023) use the APS score as the point wise score in diffusion process as it is adaptive and uniformly distribution in $[0, 1]$ under oracle probability. However, as we noted earlier, using class wise thresholds provides a mechanism to produce adaptive scores from TPS as well. Thus, we create DTPS, a variation of DAPS using TPS-Classwise scores as the point wise scores in the diffusion process.

4.4.6 Conformalized GNN

Conformalized GNN (CFGNN) (Huang et al., 2023) is a GNN-based approach for conformal prediction. The authors observed that inefficiencies are correlated between nodes having similar neighborhood topology in a graph setting. They use a GNN during the calibration phase, which is trained to correct the scores output from the base model such that the corrected scores maximize the efficiency of the conformal prediction. For classification-based losses, CFGNN utilizes the fact that all steps in the conformal prediction stage for computing the prediction sets (non-conformity score computation, quantile computation, thresholding) can be expressed as differentiable operations. Thus, a GNN can be trained

directly using efficiency as a loss function. Figure 4.4.2 provides a high-level overview of the CFGNN approach.

CFGNN Implementation Improvements The choice of the conformal loss during calibration and test plays an important role in determining the overall performance of the CFGNN. Huang et al. (2023) use a TPS loss for the calibration phase and the non-randomized APS loss for constructing the final prediction sets. Our preliminary experiments (Figure 4.4.3) with replacing the APS loss with a randomized version demonstrated that these losses must be tuned carefully to ensure that the CFGNN is able to improve upon the base models non-conformity scores. Some improvements shown in CFGNN (Figure 4.4.3, right) get nullified when the randomized APS loss is used (left).

Additionally, CFGNN uses full batch training which makes it unable to scale for larger graphs. We implemented a batched version of CFGNN to ensure that it can be used for larger graphs. Finally, to speed up computation, we allow the use of cached outputs from the base model rather than having to sample neighbors for both the base model and the CFGNN. Algorithm 4 shows these improvements in the CFGNN implementation. We cache the output of the base GNN_θ prior to running the CFGNN training loop, allowing the sampling of m layers of message passing graphs rather than $m + l$ layers required by the baseline CFGNN. In addition, we control the batch size b when sampling neighbors. These changes significantly speeds up the computation for CFGNNs (see Section 4.6.4.1 for speedup results).

4.5 Evaluation of Graph Conformal Prediction

4.5.1 Datasets

We selected datasets of varying sizes to evaluate the performance of the graph conformal prediction methods. For the citation datasets, the nodes are publications, and the edges denote citation relationships. Features are bag-of-words representations of the documents.

Algorithm 4 CFGNN batching + caching implementation

GNN_θ ▷ l layer base GNN, with pretrained, fixed weights
$\text{GNN}_\phi \leftarrow \text{RandomInitialization}(\phi)$ ▷ m layer trainable CFGNN
if cache_base **then**
    $\mathcal{X} \leftarrow \text{GNN}_\theta(\mathcal{G}, \mathcal{X})$ ▷ Compute base model output
end if
procedure $\text{CFGNNTrainStep}(\mathcal{D}_{\text{calib}}, \mathcal{G}, \mathcal{X})$
    $\mathcal{B} \leftarrow \text{SampleBatch}(\mathcal{D}_{\text{calib}}, b)$ ▷ batch size b is $|\mathcal{D}_{\text{calib}}|$ for base CFGNN
    if cache_base **then**
        CFFeats, CFMsgGraphs $\leftarrow$ $\text{NeighborSampler}(\mathcal{B}, m, \mathcal{X})$
    else
        Feats, MsgGraphs $\leftarrow \text{NeighborSampler}(\mathcal{B}, l+m, \mathcal{X})$
        CFFeats $\leftarrow \text{GNN}_\theta(\text{Feats}, \text{MsgGraphs}_{0,\ldots,m-1})$
        CFMsgGraphs $\leftarrow \text{MsgGraphs}_{m,\ldots,m+l-1}$
    end if
    scores $\leftarrow \text{GNN}_\phi(\text{CFFeats}, \text{CFMsgGraphs})$
    $\mathcal{L} \leftarrow \text{ConformalLoss}(\text{scores}, \mathcal{B})$
    $\text{UpdateWeights}(\text{GNN}_\phi, \mathcal{L})$
end procedure

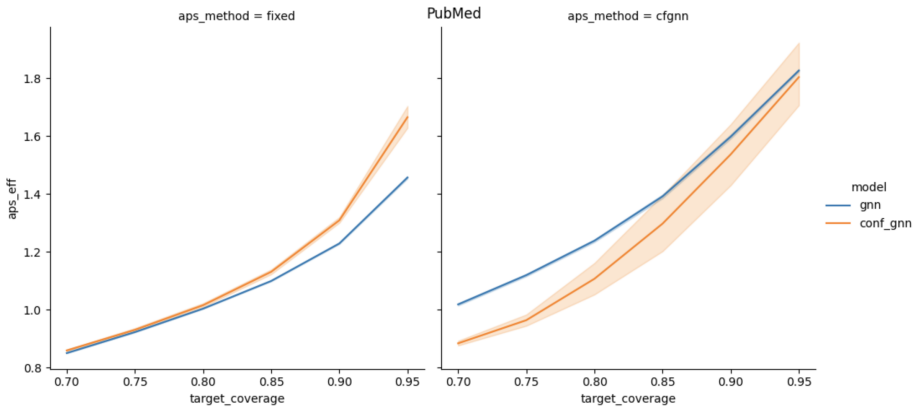


Figure 4.4.3: Comparing the efficiency (average output set size) for the base model and the CFGNN on the Pubmed dataset. The plot on the left uses the fixed version of the APS score (with randomized sets) while on the right uses the non-randomized version.

Dataset	Nodes	Edges	Classes	Features
CiteSeer	3,327	9,228	6	3,703
Amazon_Photos	7,650	238,163	8	745
Cora	19,793	126,842	70	8,710
PubMed	19,717	88,651	3	500
Coauthor_CS	18,333	163,788	15	6,805
Coauthor_Physics	34,493	495,924	5	8,415

Table 4.5.1: Summary statistics for Datasets chosen for evaluation.

The task is to predict the category of each publication. **CiteSeer** is a citation network dataset designed for the node classification task, with nodes as publications and edges denoting citation relationships. **Amazon_Photos** is a segment of the Amazon co-purchase graph (McAuley et al., 2015) where nodes represent goods, edges represent goods frequently bought together, features are bag-of-words representations of product reviews, and the task is to predict the category of each good. **Cora** We use CoraFull (Shchur et al., 2018), an extended version of the common Cora citation network dataset. The objective is to predict the category of each node (publication). **PubMed** is a citation network dataset designed for the node classification task, with nodes as publications and edges denoting citation relationships. The goal is to predict the category of each node (publication). **Coauthor_CS** and **Coauthor_Physics** are co-authorship graphs extracted from the Microsoft Academic Graph and used for KDD Cup 2016. In this dataset, nodes are authors and edges denoting co-authorship relationships. The task is to predict the most active field of study for each author. Summary statistics for the datasets are provided in Table 4.5.1. For all chosen datasets, we used the version provided by the Deep Graph Library (Wang et al., 2019a). To help characterize the behavior of different approaches, we categorize these into sizes based on

the number of nodes, with **CireSeer** and **Amazon_Photos** designated as small (S), **Cora**, **PubMed**, and **Coauthor_CS** as medium (M), and **Coauthor_Physics** as large (L).

4.5.2 Metrics

We evaluate the following metrics for the graph conformal prediction methods:

- **Coverage:** The proportion of test instances for which the true label is contained in the prediction set.
- **Efficiency:** The average size of the prediction set.
- **Label Stratified Coverage:** The mean of coverage for each class. This metric is useful for understanding whether a method is adaptive and has balanced coverage for different classes.
- **Size Stratified Coverage:** The mean of coverage across different sizes of prediction sets. This metric is useful for understanding whether a method is adaptive and does not under/over cover hard/easy samples.

4.5.3 Methods

We discussed the theoretical and empirical tradeoffs of different methods in Section 4.4. For completeness, we list all the methods that we compare here. **Threshold Prediction Sets** (Sadinle et al., 2019), with two variants, TPS and TPS-Classwise (using class wise thresholds for adapting to class imbalance). **Adaptive Prediction Sets** (Romano et al., 2020) with two variants, APS and APS-Randomized (using the uniform random quantile adjustments). **Regularized Adaptive Prediction Sets** (Angelopoulos et al., 2021b), a variation of APS with a regularization term to ensure that the prediction sets are not too large. **Diffused Adaptive Prediction Sets** (Zargarbashi et al., 2023), with two

variations DAPS and DTPS, which uses a diffusion process over TPS-Classwise. **Normalized Adaptive Prediction Sets** (Clarkson, 2023) with three variations corresponding to the weighing function used. **CF-GNN** (Huang et al., 2023), a GNN based approach for conformal prediction. We label the original implementation of CFGNN as CFGNN-Original and our improved implementations as CFGNN-APS (using randomized APS as the loss function for training/evaluation) and CFGNN-TPS (using TPS as the loss function for training/evaluation).

4.6 Results

First, we analyze the efficiency of the methods across different datasets. Figure 4.6.1 shows the efficiency of the methods across different datasets. We find that for each dataset, irrespective of the train/validation/calib split, TPS is consistently the most efficient method. However, this often comes at a cost to adaptability. In the next set of results, we show how using classwise thresholds can provide some degree of adaptability for TPS. Next, we focus on the adaptability provided by using classwise TPS.

4.6.1 Adaptability through Classwise TPS

From Figure 4.6.2, we see that using classwise TPS successfully provides stratified coverage over different labels without sacrificing size stratified coverage vs a baseline TPS. At the limit, even when reducing the number of samples per class from Figure 4.6.3, we can see that the loss in size stratified coverage is minimal. Thus, at least for the datasets we studied, TPS-Classwise is a good candidate for an adaptive version of TPS.

4.6.2 APS Randomized Sets

Figure 4.6.4 provide violin plots that compare the efficiency of randomized and non-randomized version of APS across different datasets and α. We observe that in each case,

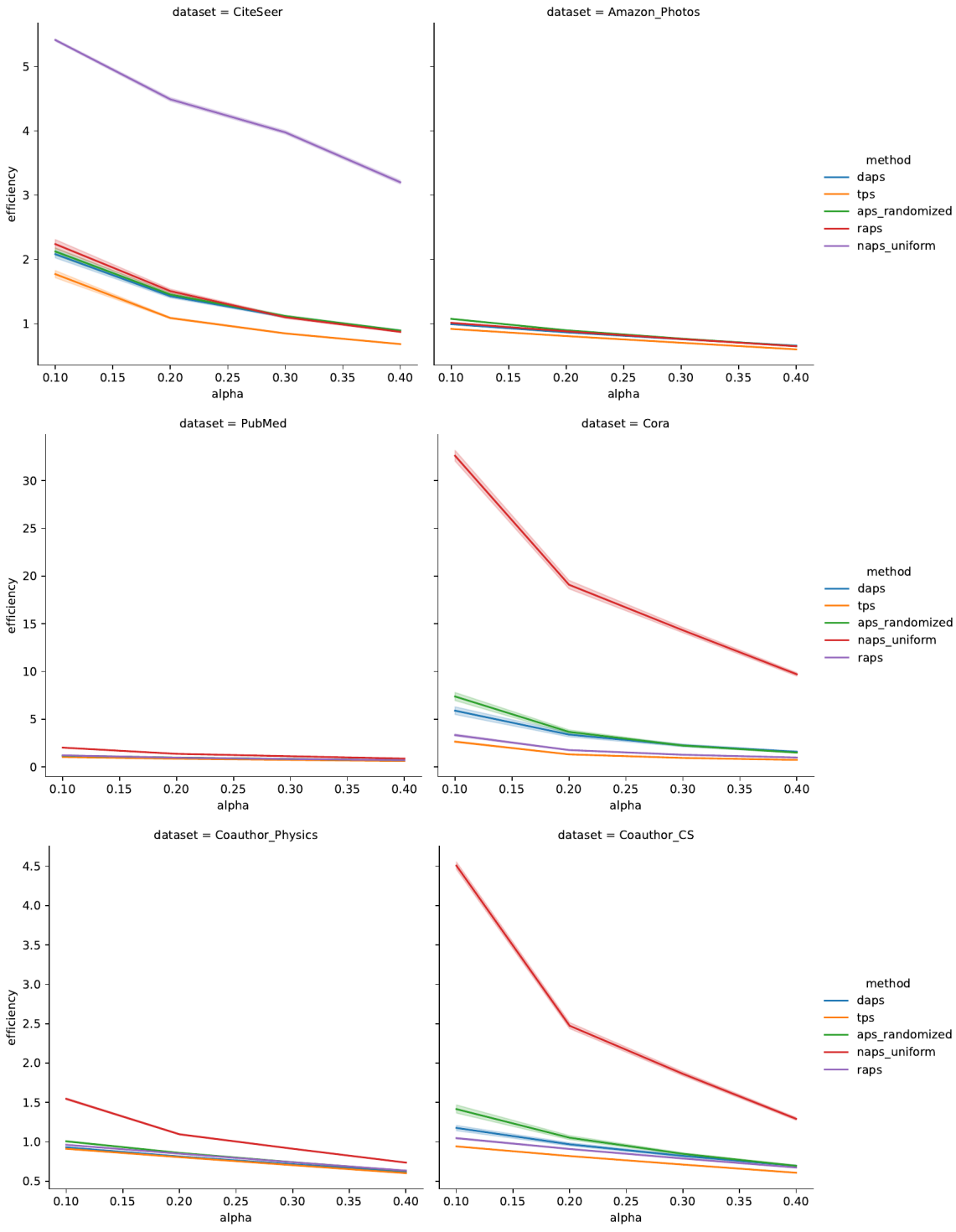


Figure 4.6.1: Plots for efficiency vs α for the major methods across the all the datasets. Among the baseline methods, TPS consistently has the best efficiency. Result for FS paritttion

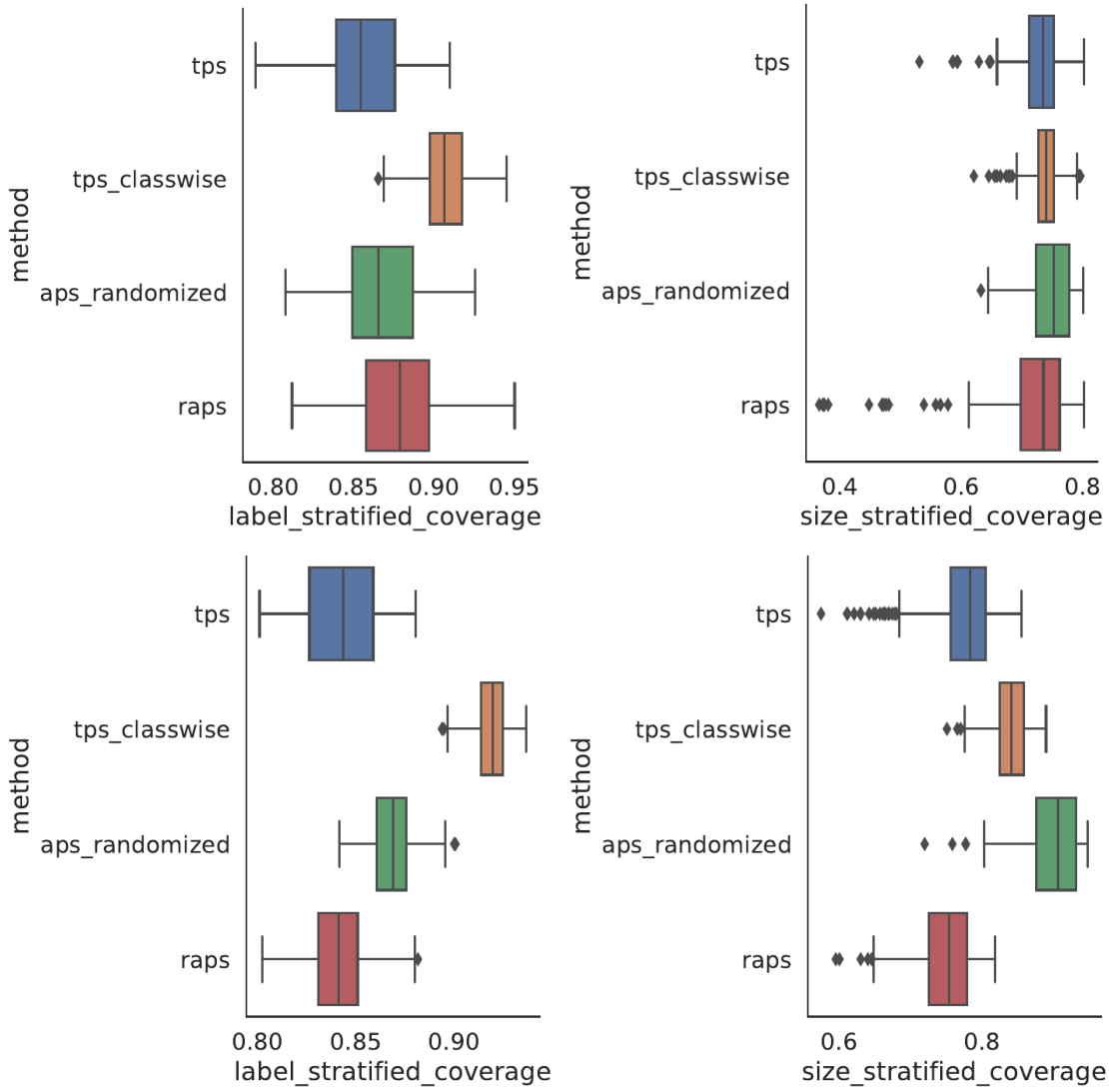


Figure 4.6.2: At a target $\alpha = 0.1$. Boxplots indicating (left) Label Stratified Coverage. (right) Size Stratified Coverage for CiteSeer (top) and Cora(bottom). Classwise TPS provides adaptability when stratified by labels without sacrificing size stratified coverage. Results for FS splits.

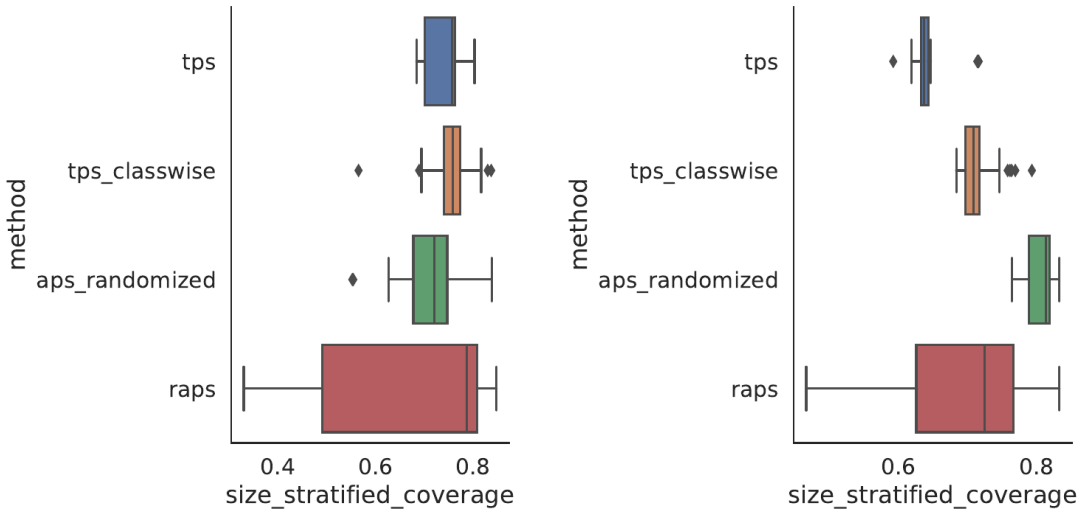


Figure 4.6.3: At a target $\alpha = 0.1$, boxplots for size stratified coverage with calibration sets having (left) 10 samples per class and (right) 40 samples per class for Amazon Photos.

the peaks associated with the randomized version lie to the left of those associated with the non-randomized version. This indicates that the randomized version consistently provides a more efficient prediction set. This effect is most pronounced for a dataset having a large number of potential classes (Cora), which matches with the intuition from Theorem 6 - with a $(K-1)\left(\alpha_c^A - \alpha_C^{\tilde{A}}\right)$ term contributing to the improved efficiency and least pronounced for PubMed, which has the smallest $K = 3$. Overall, the empirical results show that the effect of randomized APS is more apparent for larger number of classes K.

4.6.3 Diffusion Thresholded Adaptative Sets

We compare our proposed Diffusion method–of using TPS-Classwise as the base method–DTPS against DAPS, which was proposed in (Zargarbashi et al., 2023). From Figure 4.6.5, we see that when the calibration set is large (TS), DTPS can improve efficiency without sacrificing adaptiveness for PubMed but not for Cora. However, when we control the number

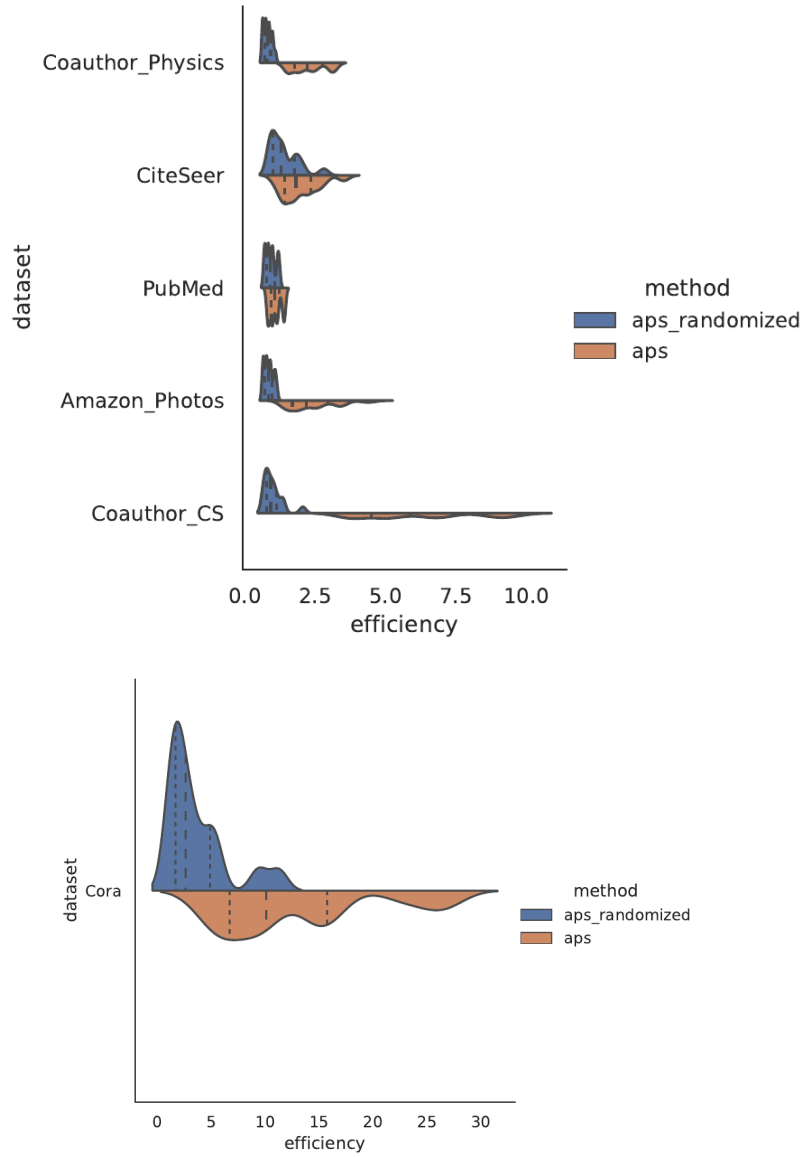


Figure 4.6.4: Violin plots denoting efficiencies of APS and Randomized APS across different datasets and multiple runs in FS split. Randomization consistently improves over the non-randomized version.

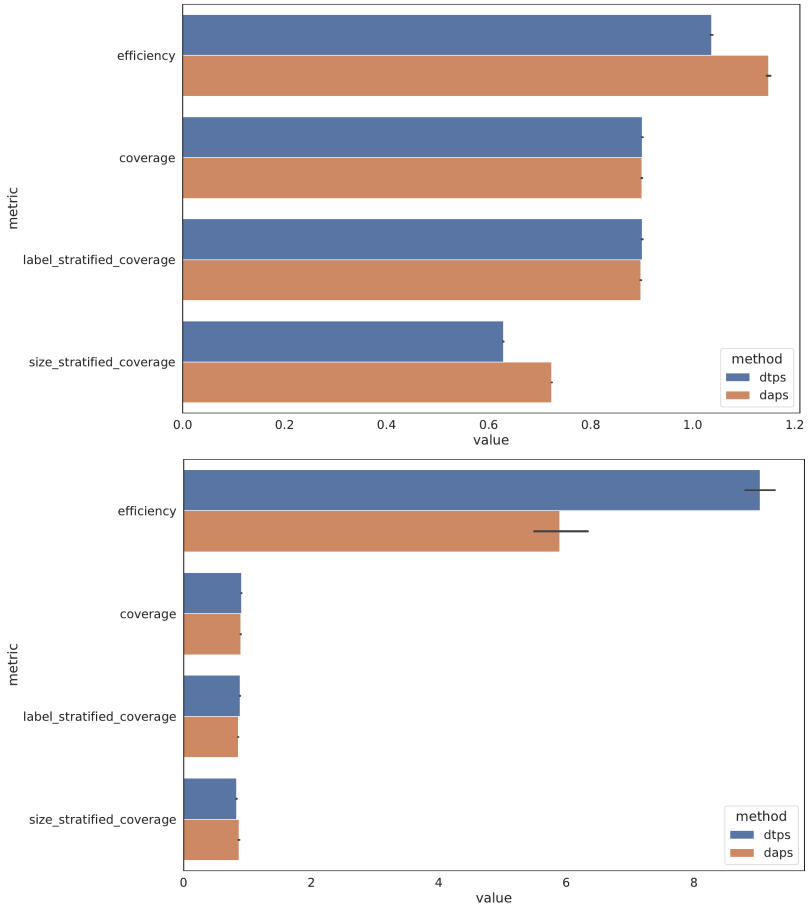


Figure 4.6.5: Bar charts denoting different metrics associated with DAPS and DTPS across PubMed (top) and Cora (bottom) for the TS split at $\alpha = 0.1$. We see that DTPS improves efficiency for PubMed but not for Cora, with minimal impact to other adaptive metrics.

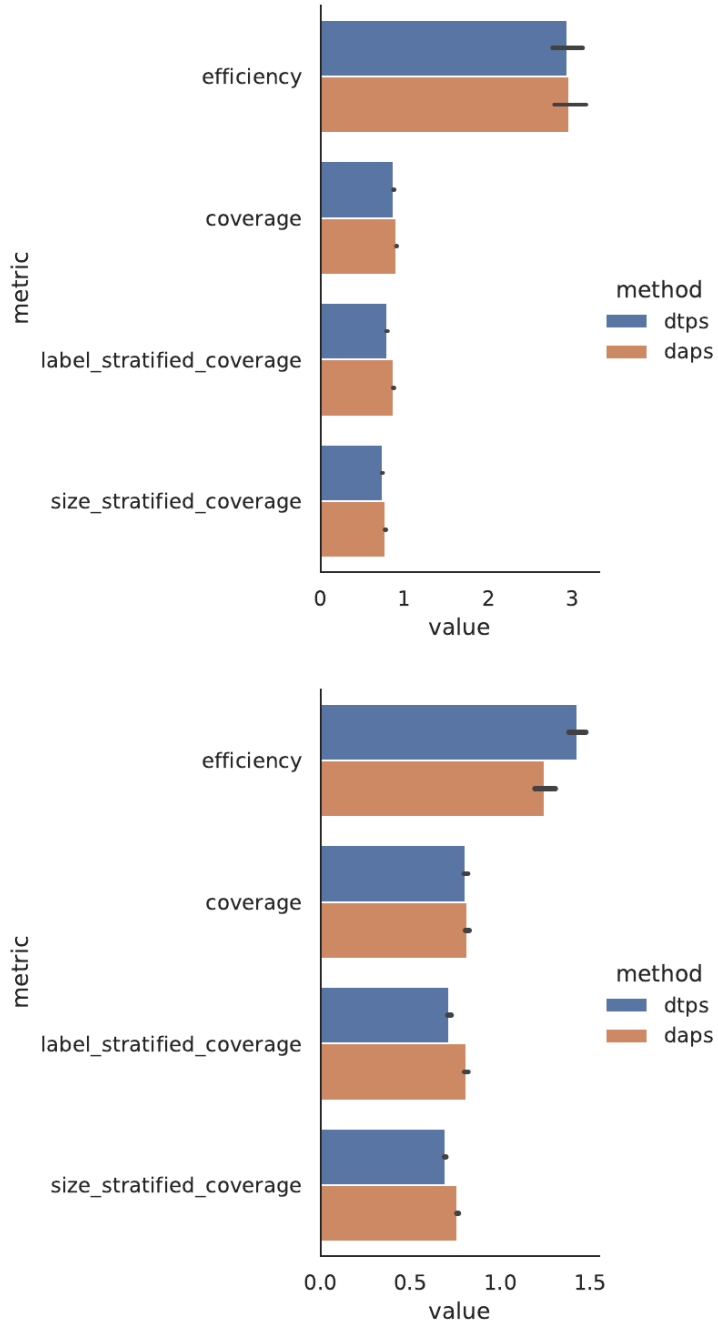


Figure 4.6.6: Bar charts denoting different metrics associated with DAPS and DTPS across the LC splits at $\alpha = 0.1$ (top) and $\alpha = 0.2$ (bottom). We see that DTPS deteriorates significantly as compared to DAPS at higher α.

of samples per class with LC splits (Figure 4.6.6), we see that DTPS deteriorates significantly as compared to DAPS at higher α. Based on these results, we can conclude that DTPS is not a universally better method than DAPS, and its performance is sensitive to the calibration set size and the number of classes. It may be a viable candidate over DAPS when there is a sufficiently large calibration set.

4.6.4 CFGNN

We first describe the runtime improvements achieved by using batching and caching in our CFGNN implementation, and follow it up with an evaluation of CFGNN-APS (randomized) and CFGNN-Original on the FS and LC splits.

4.6.4.1 Runtime

method dataset	baseline	batching	cache+batch
CiteSeer	186.61 ± 11.43	8.64 ± 1.76	4.43 ± 0.13
Amazon_Photos	291.11 ± 6.20	29.66 ± 1.03	8.27 ± 0.18
Cora	985.99 ± 62.42	89.80 ± 1.50	28.82 ± 2.08
PubMed	254.38 ± 8.09	58.26 ± 1.68	15.31 ± 0.63
Coauthor_CS	669.48 ± 34.75	72.97 ± 1.21	17.70 ± 1.64
Coauthor_Physics	2089.23 ± 80.00	758.22 ± 15.28	27.63 ± 0.81

Table 4.6.1: Runtime for CFGNN implementations starting from the baseline, then adding batching, and then adding caching and batching combined. For each setup we compare the results from 5 runs and provide 95% confidence intervals in the reported results. All runtimes in seconds, runs executed on a single A100 GPU.

We compare three variations of the CFGNN implementation to demonstrate the impact of batching and caching on the runtime. Across all comparisons, we use the FS split, with 20%/20% assigned to train/valid sets, and 35% to the calibration dataset. For ease of

comparison, we fix the CFGNN architecture to a 2-layer GCN having 128 hidden units. We use the best base GNN parameters for each dataset and split. The baseline implementation follows the setup used by (Huang et al., 2023), where the CFGNN is trained with full batch gradient descent for 1000 epochs. Our improved implementation, which uses batched descent, is able to achieve an equivalent efficiency in only 20 epochs, without any batch size tuning (we set the batch size to 64 for consistent comparison). Finally, we add caching of the output probabilities from the base GNN to the batched implementation, which further reduces the runtime. Table 4.6.1 describes the comparison of the batching, and the combined batching + caching improvements. We discard the first run in each experiment as it includes the warm up time for running on the GPU. We note that our implementation is able to achieve improvements ranging from 16.6x (PubMed) to 75.6x (Coauthor_Physics) in runtime over the baseline implementation.

4.6.4.2 Evaluation

We implement an improved version of CFGNN-APS which uses the randomized APS loss in both training and evaluation. In contrast, CFGNN-Original uses TPS during training and non-randomized APS during evaluation. We compare the efficiency of CFGNN-APS and CFGNN-Original in Figure 4.6.7. We see that CFGNN-APS improves or matches efficiency in 5/6 cases. For these results, we only trained the parameters of the CFGNN, keeping the architecture fixed. Further tuning of the architecture may improve the performance of CFGNN-APS.

Finally, originally, CFGNN was evaluated on FC splits. We benchmark its performance on LC splits in Figure 4.6.8. We see that CFGNN is unstable for the LC setting. One potential reason for this is that the CFGNN is not designed to handle the LC setting as

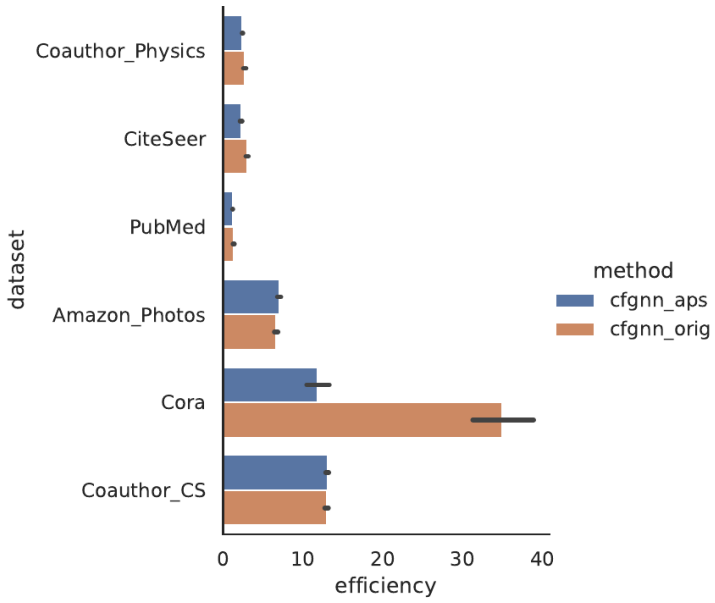


Figure 4.6.7: Bar charts denoting efficiency for CFGNN-APS and CFGNN-Original across the TS split at $\alpha = 0.1$. We see that CFGNN-APS improves or matches efficiency in most cases.

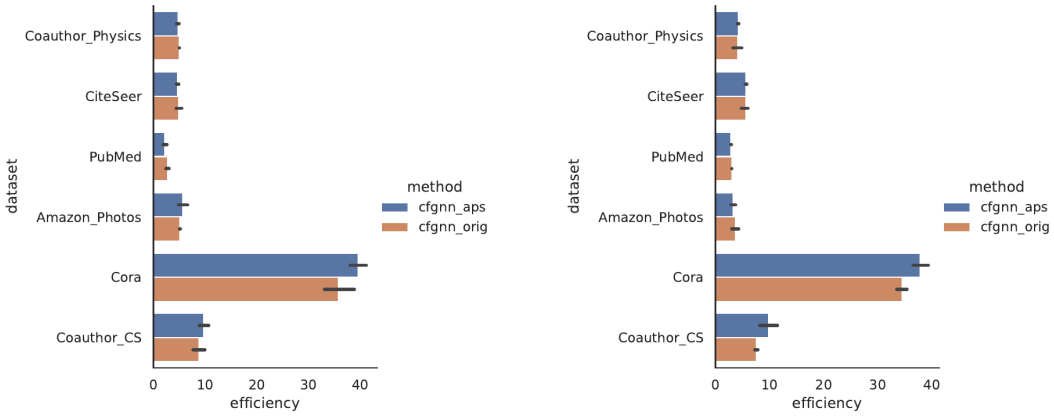


Figure 4.6.8: Bar charts denoting efficiency for CFGNN-APS and CFGNN-Original across the LC split at $\alpha = 0.1$ with 10 samples per class (left) and 20 samples per class (right). We see that CFGNN is unstable for the LC setting.

the data may be insufficient to train a conformal model. Exploring methods to improve the stability of CFGNN in the LC setting is an area for future work.

4.7 Conclusion

In this chapter, we demonstrated the tradeoffs associated with the choices made in the implementation of graph conformal prediction. We provide various recommendations for different dataset splits, methods, and evaluation metrics, which indicates relevant directions for future work in graph conformal prediction.

Appendix

4.A Optimal τ for APS

For simplicity, assume that the probabilities are distinct.

From the definition of A equation 4.4

$$A(\boldsymbol{x}, y, u; \hat{\pi}) = \min\{\tau \in [0,1] : y \in S(\boldsymbol{x}, u; \hat{\pi}, \tau)\}$$

Define

$$\Sigma_{\hat{\pi}}(\boldsymbol{x}, m) = \sum_{i=1}^{m} \hat{\pi}_{(i)}(\boldsymbol{x})$$

From the definition of $S(\boldsymbol{x}, u; \hat{\pi}, \tau)$ from equation 4.3, conisder the following cases:

Case 1: $\tau = \Sigma_{\hat{\pi}}(\boldsymbol{x}, r_y)$, then $L(x; \hat{\pi}, \tau) = y$ and thus, $V(\boldsymbol{x}; \pi, \tau) = 0$. Thus $\Pr[u > V(\boldsymbol{x}; \pi, \tau)] = 1$ and hence, $P[y \in S(\boldsymbol{x}, u; \hat{\pi}, \tau)] = 1$.

Case 2: $\tau = \Sigma_{\hat{\pi}}(\boldsymbol{x}, r_y - 1)$, then $y \notin S(\boldsymbol{x}, u, \hat{\pi}, \tau)$ in either case, since only classes with $\hat{\pi}_i(\boldsymbol{x}) > \hat{\pi}_y(\boldsymbol{x})$ could be included.

Case 3: $\tau = \Sigma_{\hat{\pi}}(\boldsymbol{x}, r_y) - \varepsilon\hat{\pi}_y$. Then we have $L(x; \hat{\pi}, \tau) = y$ again, and

$$\begin{aligned} V(\boldsymbol{x}; \pi, \tau) &= \frac{1}{\hat{\pi}_y(\boldsymbol{x})} \left\{ \left[\sum_{j=1}^{r_y} \hat{\pi}_{(j)}(\boldsymbol{x}) \right] - \tau \right\} \\ &= \frac{1}{\hat{\pi}_y(\boldsymbol{x})} \left\{ \left[\sum_{j=1}^{r_y} \hat{\pi}_{(j)}(\boldsymbol{x}) \right] - (\Sigma_{\hat{\pi}}(\boldsymbol{x}, r_y) - \varepsilon\hat{\pi}_y) \right\} \\ &= \varepsilon \end{aligned}$$

For y to be included in $S(\boldsymbol{x}, u; \hat{\pi}, \tau)$, we would require that $u \geq V(\boldsymbol{x}; \pi, \tau)$, i.e., $u \geq \varepsilon$. We want the minimal τ, which is equivalent to maximizing ε. Thus, $\tau = \Sigma_{\hat{\pi}}(\boldsymbol{x}, r_y) - u\hat{\pi}_y$ is the required solution.

4.A.1 Non-randomized set

The inclusion criterion for the score given the threshold τ is $\tilde{A}(\boldsymbol{x}, y; \hat{pi}) \leq \tau$

To include the currct label y_i while minimizing the chosen threshold τ, we would require $\tau = \sum_{j=1}^{r_{y_i}} \hat{\pi}_{(j)}(\boldsymbol{x})$

Chapter 5: Stylometry on the Darkweb

Darkweb market forums are frequently used to exchange illegal goods and services between parties who use encryption to conceal their identities. The Tor network is used to host these markets, which guarantees additional anonymization from IP and location tracking, making it challenging to link across malicious users using multiple accounts (sybils). Additionally, users migrate to new forums when one is closed further increasing the difficulty of linking users across multiple forums. We develop a novel stylometry-based multitask learning approach for natural language and model interactions using graph embeddings to construct low-dimensional representations of short episodes of user activity for authorship attribution. We provide a comprehensive evaluation of our methods across four different darkweb forums demonstrating its efficacy over the state-of-the-art, with a lift of up to 2.5X on Mean Retrieval Rank and 2X on Recall@10. Our approches demonstrate *domain adaptation* for author identification across different darkweb forums. The results in this chapter are based on our work published at EMNLP 2021 (Maneriker et al., 2021a).

5.1 Introduction

Crypto markets are *"online forums where goods and services are exchanged between parties who use digital encryption to conceal their identities"* (Martin, 2014). They are typically hosted on the Tor network, which guarantees anonymization in terms of IP and location

tracking. The identity of individuals on a crypto-market is associated only with a username; therefore, building trust on these networks does not follow conventional models prevalent in eCommerce. Interactions on these forums are facilitated by means of text posted by their users. This makes the analysis of textual style on these forums a compelling problem.

Stylometry is the branch of linguistics concerned with the analysis of authors' style. Text stylometry was initially popularized in the area of forensic linguistics, specifically to the problems of author profiling and author attribution (Juola, 2006; Rangel et al., 2013). Traditional techniques for authorship analysis on such data rely upon the existence of long text corpora from which features such as the frequency of words, capitalization, punctuation style, word and character n-grams, function word usage can be extracted and subsequently fed into any statistical or machine learning classification framework, acting as an author's 'signature'. However, such techniques find limited use in short text corpora in a heavily anonymized environment.

Advancements in using neural networks for character and word-level modeling for authorship attribution aim to deal with the scarcity of easily identifiable 'signature' features and have shown promising results on shorter text (Shrestha et al., 2017). Andrews and Bishop (2019) drew upon these advances in stylometry to propose a model for building representations of social media users on Reddit and Twitter. Motivated by the success of such approaches, we develop a novel methodology for building authorship representations for posters on various darknet markets. Specifically, our key contributions include:

First, a *representation learning* approach that couples temporal content stylometry with access identity (by levering forum interactions via *meta-path graph context information*) to model and enhance user (author) representation;

Second, a novel framework for training the proposed models in a *multitask setting* across multiple darknet markets, using a small dataset of labeled migrations, to refine the representations of users within each individual market, while also providing a method to correlate users across markets;

Third, a detailed drill-down *ablation study* discussing the impact of various optimizations and highlighting the benefits of both graph context and multitask learning on forums associated with four darknet markets - *Black Market Reloaded*, *Agora Marketplace*, *Silk Road*, and *Silk Road 2.0* - when compared to the state-of-the-art alternatives.

5.2 Related Work

Darknet Market Analysis: Content on the dark web includes resources devoted to illicit drug trade, adult content, counterfeit goods and information, leaked data, fraud, and other illicit services (Biryukov et al., 2014). Also included are forums discussing politics, anonymization, and cryptocurrency. Biryukov et al. (2014) found that while a vast majority of these services were in English (about 84%), a total of about 17 different languages were detected. Analysis of the volume of transactions and number of users on darknet markets indicates that they are resilient to closures; rapid migrations to newer markets occur when one market shuts down (ElBahrawy et al., 2019).

Recent work (Fan et al., 2018; Hou et al., 2017; Fu et al., 2017; Dong et al., 2017) has levered the notion of a heterogeneous information network (HIN) embedding to improve graph modeling, where different types of nodes, relationships (edges) and paths can be represented through typed entities. Zhang et al. (2019a) used a HIN to model marketplace vendor sybil[14] accounts on the darknet, where each node representing an object is associated with various features (e.g. content, photography style, user profile and drug information). Similarly, Kumar

[14]a single author can have multiple users accounts which are considered as *sybils*

et al. (2020) proposed a multi-view unsupervised approach which incorporated features of text content, drug substances, and locations to generate vendor embeddings. We note that while such efforts (Zhang et al., 2019a; Kumar et al., 2020) are related to our work, there are key distinctions. First, such efforts focus only on vendor sybil accounts. Second, in both cases, they rely on a host of multi-modal information sources (photographs, substance descriptions, listings, and location information) that are not readily available in our setting - limited to forum posts. Third, neither effort exploits multitask learning.

Authorship Attribution of Short Text: Kim (2014) introduced convolutional neural networks (CNNs) for text classification. Follow-up work on authorship attribution (Ruder et al., 2016; Shrestha et al., 2017) leveraged these ideas to demonstrate that CNNs outperformed other models, particularly for shorter texts. The models proposed in these works aimed at balancing the trade-off between vocabulary size and sequence length budgets based on tokenization at either the character or word level. Further work on subword tokenization (Sennrich et al., 2016), especially byte-level tokenization, have made it feasible to share vocabularies across data in multiple languages. Models built using subword tokenizers have achieved good performance on authorship attribution tasks for specific languages (e.g., Polish (Grzybowski et al., 2019)) and also across multilingual social media data (Andrews and Bishop, 2019). Non-English as well as multilingual darknet markets have been increasing in number since 2013 (Ebrahimi et al., 2018b). Our work builds upon all these ideas by using CNN models and experimenting with both character and subword level tokens.

Multitask learning (MTL): MTL (Caruana, 1997), aims to improve machine learning models' performance on the original task by jointly training related tasks. MTL enables deep neural network-based models to better generalize by sharing some of the hidden layers among the related tasks. Different approaches to MTL can be contrasted based on the

sharing of parameters across tasks - strictly equal across tasks (hard sharing) or constrained to be close (soft-sharing) (Ruder, 2017). Such approaches have been applied to language modeling (Howard and Ruder, 2018), machine translation (Dong et al., 2015), and dialog understanding (Rastogi et al., 2018).

5.3 Datasets

Munksgaard and Demant (2016) studied the politics of darknet markets using structured topic models on the forum posts across six large markets. We start with this dataset and perform basic pre-processing to clean up the text for our purposes. We focus on four of the six markets - *Silk Road* (**SR**), *Silk Road 2.0* (**SR2**), *Agora Marketplace* (**Agora**), and *Black Market Reloaded* (**BMR**). We exclude 'The Hub' as it is not a standard forum but an 'omni-forum' (Munksgaard and Demant, 2016) for discussion of other marketplaces and has a significantly different structure, which is beyond the scope of this work. We also exclude 'Evolution Marketplace' since none of the posts had PGP information present in them and thus were unsuitable for migration analysis.

Pre-processing We add simple regex and rule based filters to replace quoted posts (i.e., posts that are begin replied to), PGP keys, PGP signatures, hashed messages, links, and images each with different special tokens (`[QUOTE]`, `[PGP PUBKEY]`, `[PGP SIGNATURE]`, `[PGP ENCMSG]`, `[LINK]`, `[IMAGE]`). We retain the subset of users with sufficient posts to create at least two episodes worth of posts. In our analysis, we focus on episodes of up to 5 posts. To avoid leaking information across time, we split the dataset into approximately equal-sized train and test sets with a chronologically midway splitting point such that half the posts on the forum are before that time point. Statistics for data after pre-processing is provided in Table 5.3.1. Note that the test data can contain authors not seen during training.

Market	Train Posts	Test Posts	#Users train	#Users test
SR	379382	381959	6585	8865
SR2	373905	380779	5346	6580
BMR	30083	30474	855	931
Agora	175978	179482	3115	4209

Table 5.3.1: Dataset Statistics for Darkweb Markets.

Cross-dataset Samples Past work has established PGP keys as strong indicators of shared authorship on darkweb markets (Tai et al., 2019). To identify different user accounts across markets that correspond to the same author, we follow a two-step process. First, we select the posts containing a PGP key, and then pair together users who have posts containing the same PGP key. Following this, we still have a large number of potentially incorrect matches (including scenarios such as information sharing posts by users sharing the PGP key of known vendors from a previous market). We manually check each pair to identify matches that clearly indicate whether the same author or different authors posted them, leading to approximately 100 reliable labels, with 33 pairs matched as migrants across markets.

5.4 Methodology: SYSML Framework

Motivated by the success of social media user modeling using combinations of multiple posts by each user (Andrews and Bishop, 2019; Noorshams et al., 2020), we model posts on darknet forums using *episodes*. Each *episode* consists of the textual content, time, and contextual information from multiple posts. A neural network architecture f_θ maps each episode to combined representation $e \in \mathbb{R}^E$. The model used to generate this representation is trained on various metric learning tasks characterized by a second set of parameters $g_\phi : \mathbb{R}^E \to \mathbb{R}$. We design the metric learning task to ensure that episodes having the same

author have *similar* embeddings. Figure 5.4.1 describes the architecture of this workflow and the following sections describe the individual components and corresponding tasks. Note that our base modeling framework is inspired by the social media user representations built by Andrews and Bishop (2019) for a single task. We add meta-path embeddings and multitask objectives to enhance the capabilities of SYSML. Our implementation is available at: https://github.com/pranavmaneriker/SYSML.

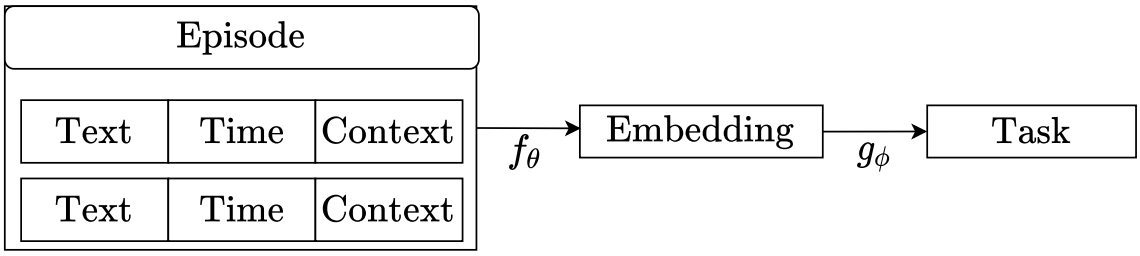


Figure 5.4.1: Overall SYSML Workflow.

5.4.1 Component Embeddings

Each episode e of length L consists of multiple tuples of texts, times, and contexts

$$e = \{(t_i, \tau_i, c_i) | 1 \leq i \leq L\}$$

. Component embeddings map individual components to vector spaces. All embeddings are generated from the forum data only; no pretrained embeddings are used.

Text Embedding First, we tokenize every input text post using either a character-level or byte-level tokenizer. A one-hot encoding layer followed by an embedding matrix E_t of dimensions $|V| \times d_t$ where V is the token vocabulary and d_t is the token embedding dimension embeds an input sequence of tokens $T_0, T_1, \ldots, T_{n-1}$. We get a sequence embedding of dimension $n \times d_t$. Following this, we use f sliding window filters, with filters sized

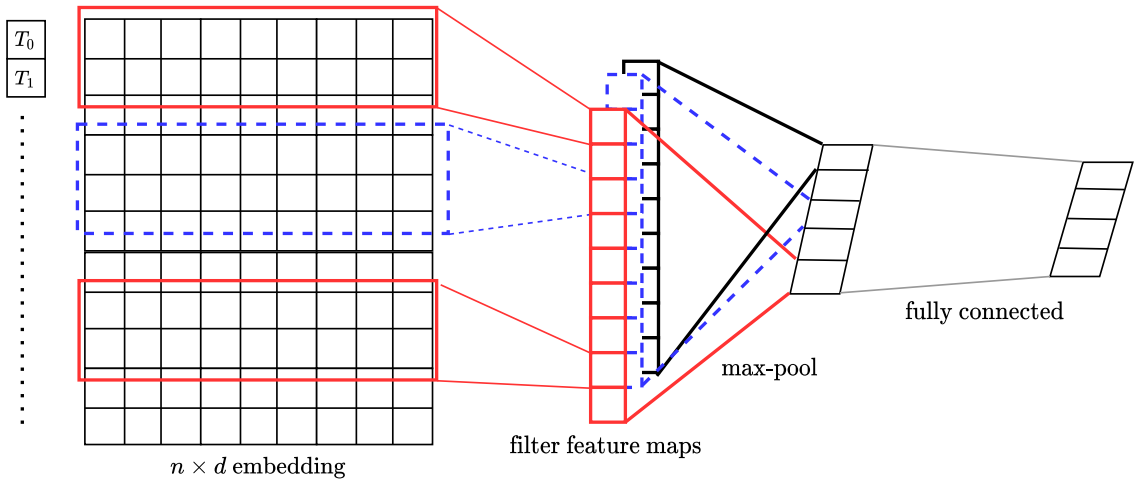


Figure 5.4.2: Text Embedding CNN (Kim, 2014).

$F = \{2, 3, 4, 5\}$ to generate feature-maps which are then fed to a max-over-time pooling layer, leading to a $|F| \times f$ dimensional output (one per filter). Finally, a fully connected layer generates the embedding for the text sequence, with output dimension d_t. A dropout layer prior to the final fully connected layer prevents overfitting, as shown in Figure 5.4.2.

Time Embedding The time information for each post corresponds to when the post was created and is available at different granularities across darknet market forums. To have a consistent time embedding across different granularities, we only consider the least granular available date information (date) available on all markets. We use the day of the week for each post to compute the time embedding by selecting the corresponding embedding vector of dimension d_τ from the matrix E_w.

Structural Context Embedding The context of a post refers to the threads that it may be associated with. Past work (Andrews and Bishop, 2019) used the subreddit as the context for a Reddit post. In a similar fashion, we encode the subforum of a post as a one-hot vector and use it to generate a d_c dimensional context embedding. In the previously mentioned work,

this embedding is initialized randomly. We deviate from this setup and use an alternative approach based on a *heterogeneous graph* constructed from forum posts to initialize this embedding.

Definition 4 (Heterogeneous Graph)**.** *A heterogeneous graph $G = (V, E, T)$ is one where each node v and edge e are associated with a 'type' $T_i \in T$, where the association is given by mapping functions $\phi(v) : V \rightarrow T_V$, $\psi(e) : E \rightarrow T_E$, where $|T_V| + |T_E| > 2$*

The constraint on $T_{V,E}$ ensures that at least one of T_V and T_E have more than one element (making the graph heterogeneous). Specifically, we build a graph in which there are four types of nodes: user (U), subforum (S), thread (T), and post (P), and each edge indicates either a post of new thread (U-T), reply to existing post (U-P) or an inclusion (T-P, S-T) relationship. To learn the node embeddings in such heterogeneous graphs, we leverage the metapath2vec (Dong et al., 2017) framework with specific meta-path schemes designed for darknet forums. Each meta-path scheme can incorporate specific semantic relationships into node embeddings. For example, Figure 5.4.3 shows an instance of a meta-path 'UTSTU', which connects two users posting on threads in the same subforum and goes through the relevant threads and subforum. Our analysis is user focused; to capture user behavior, we consider *all* metapaths starting from and ending at a user node. Thus, to fully capture the semantic relationships in the heterogeneous graph, we use seven meta-path schemes: UPTSTPU, UTSTPU, UPTSTU, UTSTU, UPTPU, UPTU, and UTPU. As a result, the learned embeddings will preserve the semantic relationships between each subforum, included posts as well as relevant users (authors). Metapath2vec generates embeddings by maximizing the probability of heterogeneous neighbourhoods, normalizing it across typed contexts. The

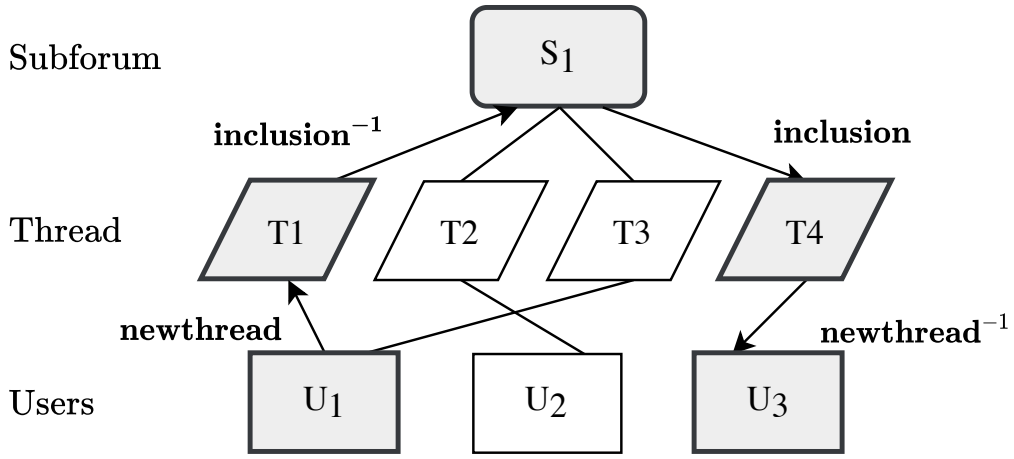


Figure 5.4.3: An instance of meta-path 'UTSTU' in a subgraph of the forum graph.

optimization objective is:

$$\arg\max_{\theta} \prod_{v \in V} \prod_{t \in T_v} \prod_{c_t \in N_t(v)} p(c_t|v;\theta)$$

Where θ is the learned embedding, $N_t(v)$ denotes v's neighborhood with the t^{th} type of node. In practice, this is equivalent to running a word2vec (Mikolov et al., 2013a) style skip gram model over the random walks generated from the meta-path schemes when $p(c_t|v;\theta) =$ is defined as a softmax function. Further details of metapath2vec can be found in the paper by Dong et al. (2017).

5.4.2 Episode Embedding

The embeddings of each component of a post are concatenated into a $d_e = d_t + d_\tau + d_c$ dimensional embedding. An episode with L posts, therefore, has a $L \times d_e$ embeddings. We generate a final embedding for each episode, given the post embeddings using two different models. In **Mean Pooling**, the episode embedding is the mean of L post embeddings, resulting in a d_e dimensional episode embedding. For the **Transformer**, the episode embeddings are fed as the inputs to a transformer model (Devlin et al., 2019; Vaswani et al.,

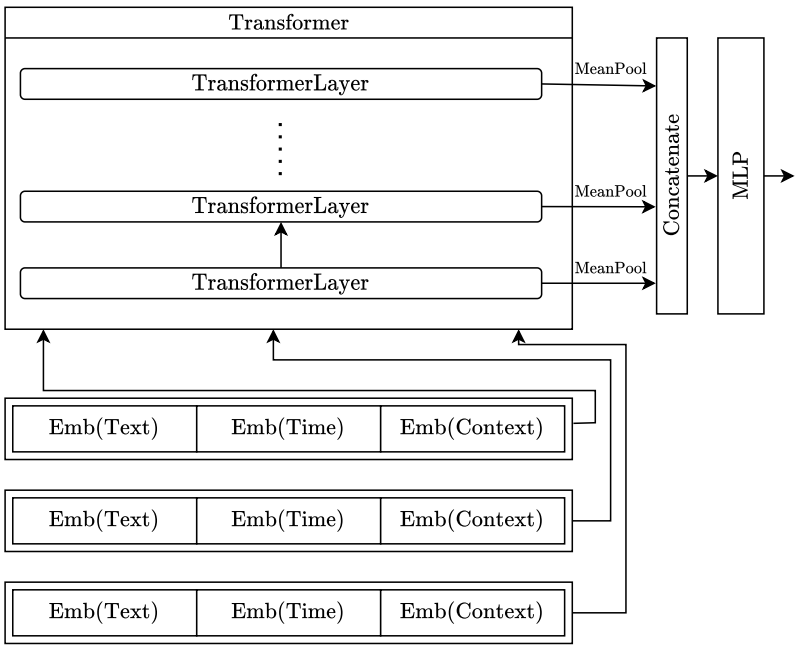


Figure 5.4.4: Architecture for Transformer Pooling.

2017), with each post embedding acting as one element in a sequence for a total sequence length L. We follow the architecture proposed by Andrews and Bishop (2019) and omit a detailed description of the transformer architecture for brevity (Figure 5.4.4 shows an overview). Note that we do not use positional embeddings within this pooling architecture. The parameters of the component-wise models and episode embedding models comprise the episode embedding $f_\theta : \{(t, \tau, c)\}^L \to \mathbb{R}^E$.

5.4.3 Metric Learning

An important element of our methodology is the ability to learn a distance function over user representations. We use the username as a label for the episode e within the market M and denote each username as a unique label $u \in U_M$. Let $W = |U_M| \times d_E$ represent a matrix denoting the weights corresponding to a specific metric learning method and let $x^* = \frac{x}{||x||}$.

An example of a metric learning loss would be Softmax Margin, i.e., cross-entropy based softmax loss.

$$P(u|e) = \frac{e^{W_u d_e}}{\sum_{j=1}^{|U_M|} e^{W_j d_e}}$$

We also explore alternative metric learning approaches such as Cosface (CF) (Wang et al., 2018), ArcFace (AF) (Deng et al., 2019), and MultiSimilarity (MS) (Wang et al., 2019b).

5.4.4 Single-Task Learning

The components discussed in the previous sections are combined together to generate an embedding and the aforementioned tasks are used to train these models. Given an episode $e = \{(t_i, \tau_i, c_i) | 1 \leq i \leq L\}$, the componentwise embedding modules generate embedding for the text, time, and context, respectively. The pooling module combines these embeddings into a single embedding $e \in \mathbb{R}^E$. We define f_θ as the combination of the transformations that generate an embedding from an *episode*. Using a final metric learning loss corresponding to the task-specific g_ϕ, we can train the parameters θ and ϕ. The framework, as defined in Figure 5.4.1, results in a model trainable for a single market M_i. Note that the first half of the framework (i.e., f_θ) is sufficient to generate embeddings for episodes, making the module invariant to the choice of g_ϕ. However, the embedding modules learned from these embeddings may not be compatible for comparisons across different markets, which motivates our multi-task setup.

5.4.5 Multi-Task Learning

We use authorship attribution as the metric learning task for each market. Further, a majority of the embedding modules are shared across the different markets. Thus, in a multi-task setup, the model can share episode embedding weights (except context, which

is market dependent) across markets. A shared BPE vocabulary allows weight sharing for text embedding on the different markets. However, the task-specific layers are not shared (different authors per dataset), and sharing f_θ does not guarantee alignment of embeddings across datasets (to reflect migrant authors). To remedy this, we construct a small, manually annotated set of labeled samples of authors known to have migrated from one market to another. Additionally, we add pairs of authors known to be distinct across datasets. The *cross-dataset* consists of all episodes of authors that were manually annotated in this fashion. The first step in the multi-task approach is to choose a market ($\mathcal{T}_M$) or cross-market ($\mathcal{T}_{cr}$) metric learning task $\mathcal{T}_i \sim \mathcal{T} = \{\mathcal{T}_M, \mathcal{T}_{cr}\}$. Following this, a batch of N episodes $\mathcal{E} \sim \mathcal{T}_i$ is sampled from the corresponding task. The embedding module generates the embedding for each episode $f_\theta^N : \mathcal{E} \rightarrow \mathbb{R}^{N \times E}$. Finally, the task-specific metric learning layer $g_\phi^{\mathcal{T}_i}$ is selected and a task-specific loss is backpropagated through the network. Note that in the *cross-dataset*, new labels are defined based on whether different usernames correspond to the same author and episodes are sampled from the corresponding markets. Figure 5.4.5 demonstrates the shared layers and the use of *cross-dataset* samples. The overall loss function is the sum of the losses across the markets: $\mathcal{L} = \mathop{\mathbb{E}}_{\mathcal{T}_i \sim \mathcal{T},\, \mathcal{E} \sim \mathcal{T}_i} [\mathcal{L}_i(\mathcal{E})]$.

5.5 Evaluation

While ground truth labels for a single author having multiple accounts are unavailable, individual models can still be compared by measuring their performance on authorship attribution as a proxy. We evaluated our method using retrieval-based metrics over the embeddings generated by each approach. Denote the set of all episode embeddings as $E = \{e_1, \ldots e_n\}$ and let $Q = \{q_1, q_2, \ldots q_\kappa\} \subset E$ be the sampled subset. We computed the cosine similarity of the query episode embeddings with all episodes. Let $R_i = \langle r_{i1}, r_{i2}, \ldots r_{in} \rangle$

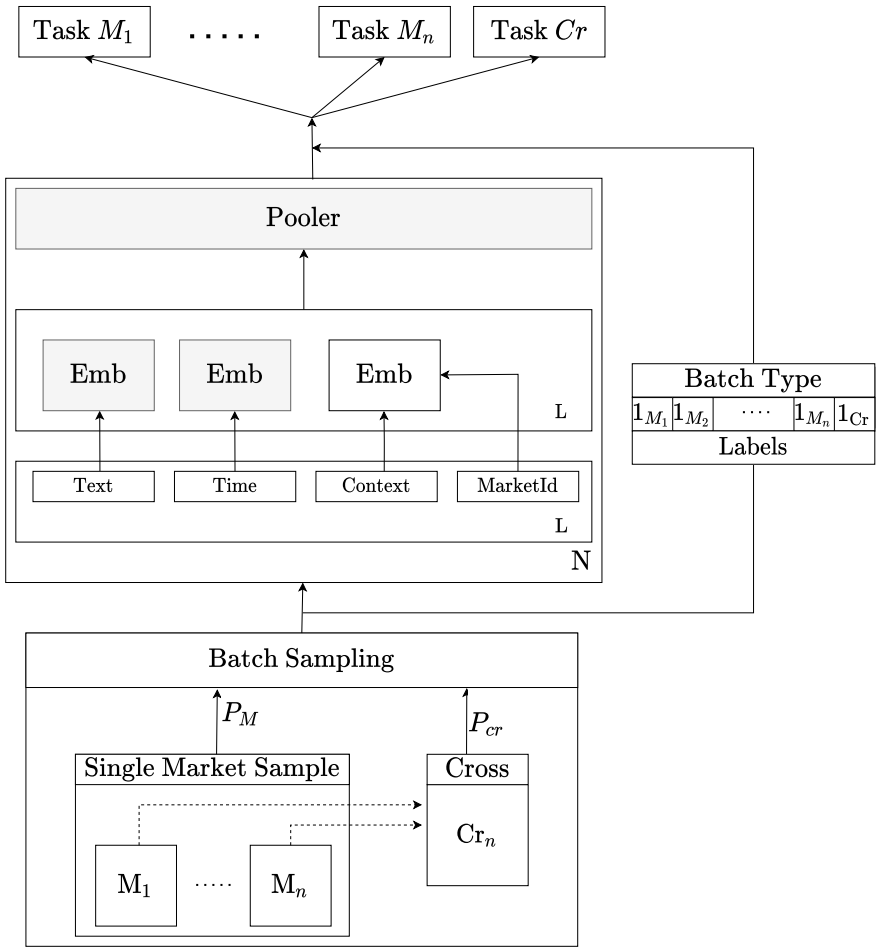


Figure 5.4.5: Multi-task setup. Shaded nodes are shared

denote the list of episodes in E ordered by their cosine similarity with episode q_i (excluding itself) and let $A(.)$ map an episode to its author. The following measures are computed.

Mean Reciprocal Rank: (MRR) The RR for an episode is the reciprocal rank of the first element (by similarity) with the same author. MRR is the mean of reciprocal ranks for a sample of episodes.

$$MRR(Q) = \frac{1}{\kappa}\sum_{i=1}^{\kappa}\frac{1}{\min_j\left(A(r_{ij}) = A(e_i)\right)}$$

Recall@k: (R@k) Following Andrews and Bishop (2019), we define the R@k for an episode e_i to be an indicator denoting whether an episode by the same author occurs within the subset $\langle r_{i1}, \ldots, r_{ik}\rangle$. R@k denotes the mean of these recall values over all the query samples.

$$R@k = \frac{1}{\kappa}\sum_{i=1}^{\kappa}\mathbf{1}_{\{\exists\ j|1\leq j\leq k, A(r_{ij})=A(e_i)\}}$$

Baselines We compare our best model against two baselines. First, we consider a popular short text authorship attribution model (Shrestha et al., 2017) based on embedding each post using character CNNs. While the method had no support for additional attributes (time, context) and only considers a single post at a time, we compare variants that incorporate these features as well. The second method for comparison is invariant representation of users (Andrews and Bishop, 2019). This method considers only one dataset at a time and does not account for graph-based context information. Results for episodes of length 5 are shown in Table 5.5.1

5.6 Analysis

5.6.1 Model and Task Variations

To compare the variants using statistical tests, we compute the MRR of the data grouped by market, episode length, tokenizer, and a graph embedding indicator. This leaves a

Method	BMR		Agora		SR2		SR	
	MRR	R@10	MRR	R@10	MRR	R@10	MRR	R@10
Shrestha et al. (2017) (CNN)	0.07	0.165	0.126	0.214	0.082	0.131	0.036	0.073
+ time + context	0.235	0.413	0.152	0.263	0.118	0.21	0.094	0.178
+ time + context + transformer pooling	0.219	0.409	0.146	0.266	0.117	0.207	0.113	0.205
Andrews and Bishop (2019) (IUR)								
mean pooling	0.223	0.408	0.114	0.218	0.126	0.223	0.109	0.19
transformer pooling	0.283	0.477	0.127	0.234	*0.13*	*0.229*	0.118	0.204
SYSML (single)	*0.32*	*0.533*	*0.152*	*0.279*	0.123	0.21	*0.157*	*0.266*
- graph context	0.265	0.454	0.144	0.251	0.089	0.15	0.049	0.094
-graph context - time	0.277	0.477	0.123	0.198	0.079	0.131	0.04	0.08
SYSML (multitask)	**0.438**	**0.642**	**0.303**	**0.466**	**0.304**	**0.464**	**0.227**	**0.363**
- graph context	0.396	0.602	**0.308**	**0.469**	0.293	0.442	0.214	0.347
- graph context - time	0.366	0.575	0.251	0.364	0.236	0.358	0.167	0.28

Table 5.5.1: Best performing results in **bold**. Best performing single-task results in *italics*. All $\sigma_{MRR} < 0.02$, $\sigma_{R@10} < 0.03$, For all metrics, higher is better. Results suggest single-task performance largely outperforms the state-of-the-art (Shrestha et al., 2017; Andrews and Bishop, 2019), while our novel multi-task cross-market setup offers a substantive lift (**up to 2.5X on MRR and 2X on R@10**) over single-task performance.

small number of samples for paired comparison between groups, which precludes making normality assumptions for a t-test. Instead, we applied the paired two-samples Wilcoxon-Mann-Whitney (WMW) test (Mann and Whitney, 1947). The first key contribution of our model is the use of meta-graph embeddings for context. The WMW test demonstrates that using pretrained graph embeddings was significantly better than using random embeddings ($p < 0.01$). Table 5.5.1 shows a summary of these results using ablations. For completeness of the analysis, we also compare the character and BPE tokenizers. WMW failed to find any significant differences between the BPE and character models for embedding (table omitted for brevity). Many darkweb markets tend to have more than one language (e.g., BMR had a large German community), and BPE allows a shared vocabulary to be used across multiple datasets with very few out-of-vocab tokens. Thus, we use BPE tokens for the forthcoming multitask models.

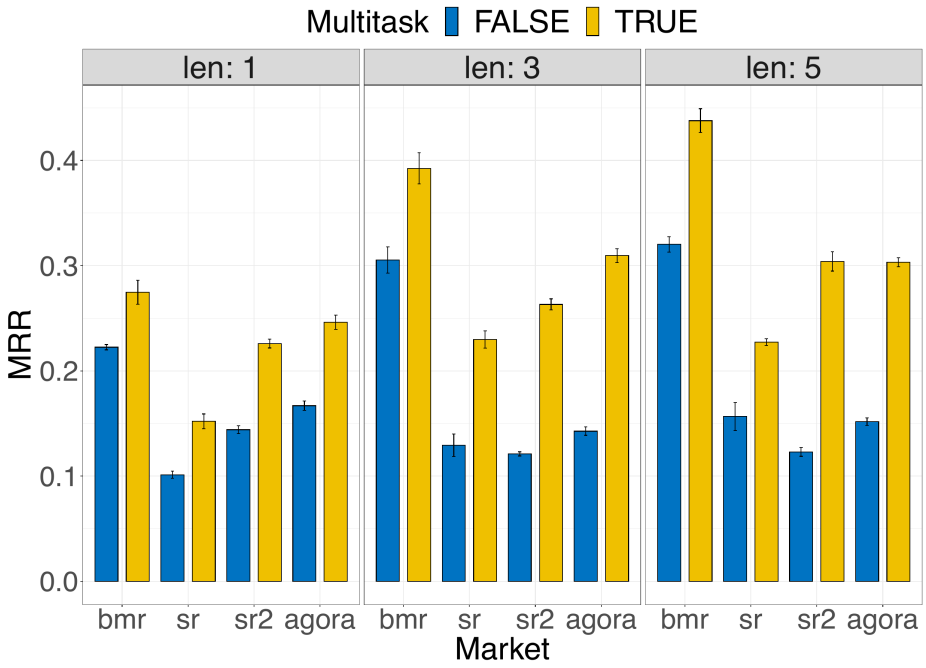


Figure 5.6.1: Drill-down: one-at-a-time vs. multitask.

Multitask Our second key contribution is the multitask setup. Table 5.5.1 demonstrates that SYSML (multitask) outperforms all baselines on episodes of length 5. We further compare runs of the best single task model for each market against a multitask model. Figure 5.6.1 demonstrates that multitask learning consistently and significantly (WMW: $p < 0.01$) improves performance across all markets and all episode lengths.

Metric Learning Recent benchmark evaluations have demonstrated that different metric learning methods provide only marginal improvements over classification (Musgrave et al., 2020; Zhai and Wu, 2019). We experimented with various state-of-the-art metric learning methods (§5.4.3) in the multi task setup and found that softmax-based classification (SM) was the best performing method in 3 of 4 cases for episodes of length 5 (Figure 5.6.2). Across

all lengths, SM is significantly better (WMW: $p < 1e-8$) and therefore we use SM in SYSML.

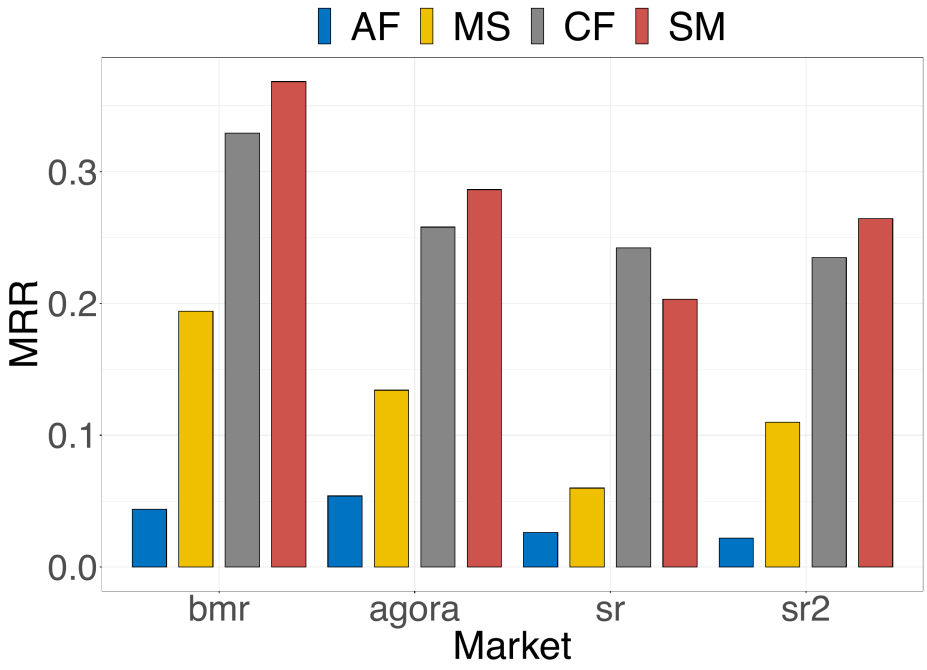


Figure 5.6.2: Task comparison: SM and CF are better performing two methods, with SM better in 3 of 4 cases.

5.6.2 Novel Users

The dataset statistics (Table 5.3.1) indicate that there are users in each dataset who have no posts in the time period corresponding to the training data. To understand the distribution of performance across these two configurations, we compute the test metrics over two samples. For one sample, we constrain the sampled episodes to those by users who have at least one episode in the training period (Seen Users). For the second sample, we sample episodes from the complement of the episodes that satisfy the previous constraint (Novel Users). Figure 5.6.3 shows the comparison of MRR on these two samples against the

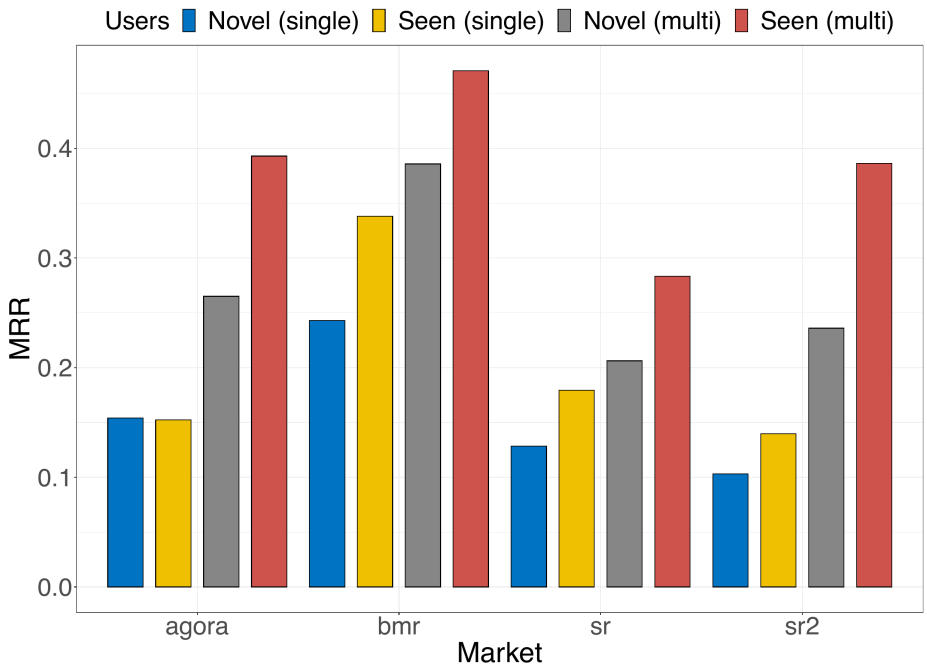


Figure 5.6.3: Lift on the multitask setup across users.

best single task model for episodes of length 5. Unsurprisingly, the first sample (Seen Users) have better query metrics than the second (Novel Users). However, importantly both of these groups outperformed the best single task model results on the first group (Seen Users), which demonstrates that the *lift offered by the multitask setup is spread across all users.*

Episode Length Figure 5.6.4 shows a comparison of the mean performance of each model across various episode lengths. We see that compared to the baselines, SYSML can combine contextual and stylistic information across multiple posts more effectively. Additional results (see appendix), indicate that this trend continues for larger episode sizes.

From Figure 5.6.5, we see that the number of users reduces rapidly as the posts per user decrease. Thus, we limited our analysis to up to 5 posts per episode. For completeness, we also provide additional results for 7 and 9 posts per episode in Table 5.6.1 and 5.6.2

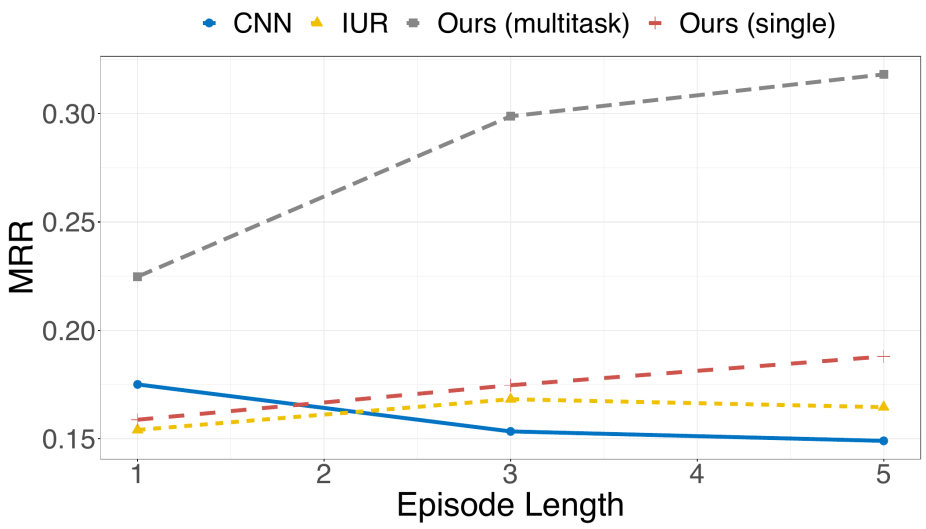


Figure 5.6.4: SYSML is more effective at utilizing multi post stylometric information

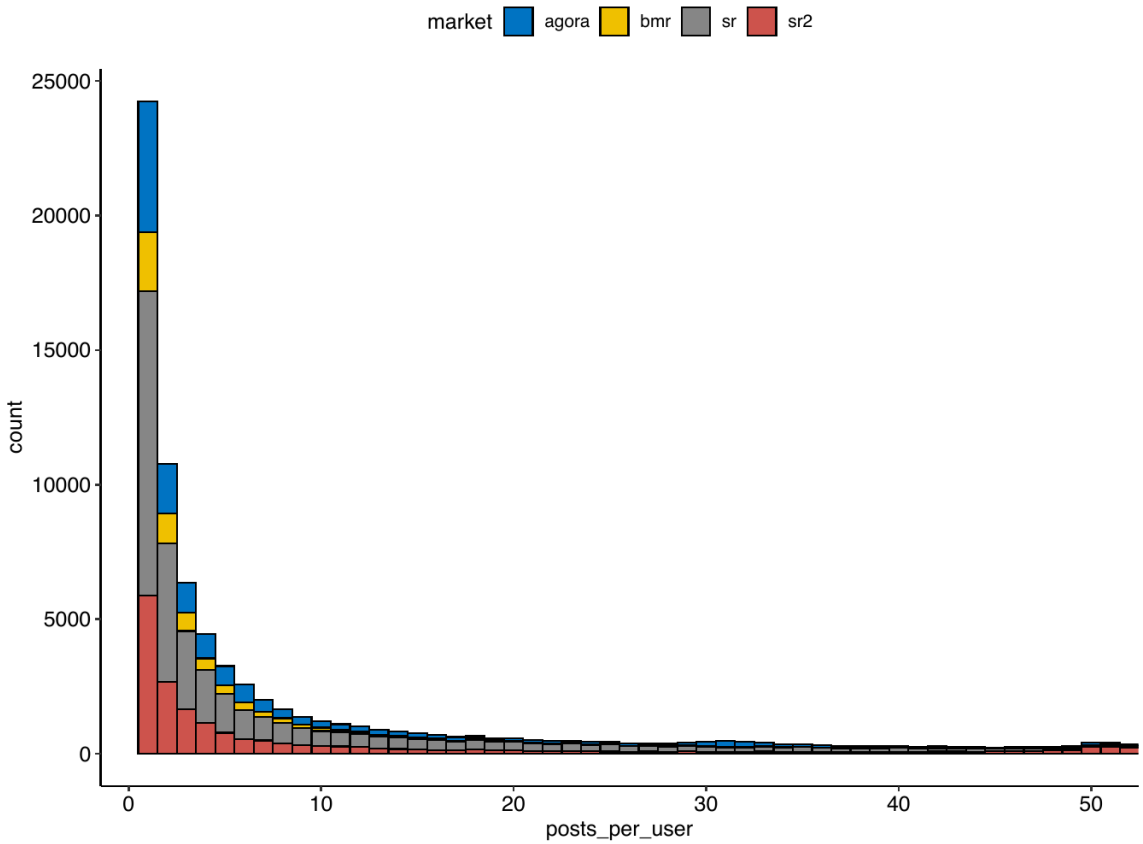


Figure 5.6.5: Frequency of number of posts per user

Method	BMR		Agora		SR2		SR	
	MRR	R@10	MRR	R@10	MRR	R@10	MRR	R@10
SYSML (singletask)	0.305	0.508	0.186	0.32	0.159	0.273	0.14	0.246
SYSML (multitask)	0.484	0.689	0.349	0.519	0.401	0.556	0.292	0.429

Table 5.6.1: Additional results for 7 posts per episode

Method	BMR		Agora		SR2		SR	
	MRR	R@10	MRR	R@10	MRR	R@10	MRR	R@10
SYSML (singletask)	0.264	0.48	0.146	0.249	0.165	0.272	0.194	0.319
SYSML (multitask)	0.4667	0.648	0.357	0.498	0.377	0.522	0.299	0.449

Table 5.6.2: Additional results for 9 posts per episode

respectively. Note that the histogram has some non-smooth bumps at around 10, 50, 100 posts as they act as the minimum number of posts for different levels of forum users. As explained in a previous section, users post on 'newbie' forums until they reach a specific number of posts, leading to these unusual bumps in the histogram. We note that the performance of our methods continues to improve as the posts per episode are increased (at a cost to coverage - number of users studied), though the improvement is higher in the bigger markets as these tend to have a sufficiently large number of individuals with a higher number of total posts.

5.7 Case Study

5.7.1 Qualitative Analysis of Attribution:

In this section, we consider the average (euclidean) distance between each pair of episodes by the same author as a heuristic for stylometric identifiability (SI), where lower average distance corresponds to higher SI and vice versa. Somewhat surprisingly, authors with a

small number of total episodes (< 10) were found at both extremes of identifiability, while the authors with the highest number of episodes were in the intermediate regions, suggesting that SI is not strongly correlated with episode length. Next, we further investigate these groups.

High SI authors: Among the 20 users with the lowest average distance between episodes, a single pattern is prominent. This first group of high SI users are "newbie" users. On a majority of analyzed forums, a minimum number of posts by a user is required before posting restrictions are removed from the user's account. Thus, users create threads on 'Newbie Discussion' subforums. Typical posts on these threads include repeated posting of the same message or numbered posts counting up to the minimum required. As users tend to make all these posts within a fixed time frame, the combination of repeated, similar stylistic text and time makes the posts easy to identify. Exemplar episodes from this "newbie" group are shown in Table 5.7.1.

After filtering these users out, we identified a few more notable high SI users. These include an author on BMR with frequent '£' symbol and ellipses ('...') and an author on Agora who only posted referral links (with an eponymous username 'ReferralLink'). Finally, restricting posts to those made by 200 most frequently posting users (henceforth, T200), we found a user (labeled HSI-Sec[15]) who frequently provided information on security, where character n-grams corresponding to 'PGP', 'Key', 'security' are frequent (Table 5.7.2). Thus, SYSML is able to leverage vocabulary and punctuation-based cues for SI.

Low SI authors: Here, we attempt to characterize the post episode styles that are challenging for SYSML to attribute to the correct author. Seminal work by Brennan and Greenstadt (2009); Brennan et al. (2012) has demonstrated that obfuscation and imitation

[15]pseudonym

Thread	Posts
Spam to 50 & Get out of Noobville	26, 27, 28, 29, 30
Post 30 Times ... To Post Anywhere	7, 8, 9, ...
Spam to 50 ...	46, 47, ..., Yeah 50 Spam!
... use my link ...	[LINK], Here is my ref link [LINK], Try this link [LINK], ...

Table 5.7.1: Examples of highly identifiable posts.

based strategies are effective against text stylometry. We analyze the T200 authors who had high inter-episode distances to ascertain whether this holds true for SYSML. For the least (and third least) identifiable author among T200, we find that frequent word n-grams are significantly less frequent than those for the most identifiable author from this subset (most frequent token occurs ~ 600 times vs. ~ 4800 times for identifiable) despite having more episodes overall. Further, one of the most frequent tokens is the `[QUOTE]` token, implying that this author frequently incorporates other authors' quotes into their posts. This strategy is analogous to the imitation based attack strategy proposed by Brennan et al. (2012). For the second least identifiable T200 author, we find that the frequent tokens have even fewer occurrences, and the special token `[IMAGE]` and its alternatives are among the frequent tokens - suggesting that an obfuscation strategy based on diversifying the vocabulary is effective. Some samples are presented in Table 5.7.2 under LSI-1 and LSI-2.

Gradient-based attribution: To cement our preceding hypotheses, we investigate whether the generated embedding can be attributed to phrases in the input which were mentioned in the previous section. We use Integrated Gradients (Sundararajan et al., 2017), an axiomatic approach to input attribution. Integrated Gradients assign an importance score to each feature which corresponds to an approximation of the integral of the gradient of a model's

Author	Word Importance
HSI-Sec	... 2 cents, anyway ... PGP Key Fingerprint = ... PGP Key Fingerprint ... ... security is NOT retroactive . ... Is it possible for a gpg key to request that)
LSI-1	Check out the link in my sig ... [IMAGE alt=8)] Hey dude, just run a search ... I can not help much ... Im sure if you ask ... German , he may be willing to lend a hand. Good luck freind [IMAGE alt=8)]
LSI-2	[QUOTE] From: ... Just my opinion, I 've done just about everything, ... IMAGE alt=8)] couldnt agree more [QUOTE] From : ... strangely enough, when im in ... I too jabber meaningless jibberish ...
	Negative ■ Neutral □ Positive ■

Table 5.7.2: Integrated Gradient based attribution of posts

output with respect to the input features along a path from some reference baseline value (in our case, all `[PAD]` tokens) to the input feature. In Table 5.7.2, the highlight color corresponds to the attribution importance score for the presented posts. We observed that the attribution scores correspond to our intuitions: HSI-Sec had high importance for security words, LSI-1 had obfuscated posts due to the presence of common image tokens, and LSI-2 had quotes mixed in, lead to misattribution (imitation-like strategy).

5.7.2 Migrant Analysis

To understand the quality of alignment from the episode embeddings generated by our method, we use a simple top-k heuristic: for each episode of a user, find the top-k nearest neighboring episodes from other markets, and count the most frequently occurring user among these (candidate sybil account). Figure 5.7.1 shows a UMAP projection for T200. Users of each market are colored by sequential values of a single hue (i.e., reds - SR2, blues -

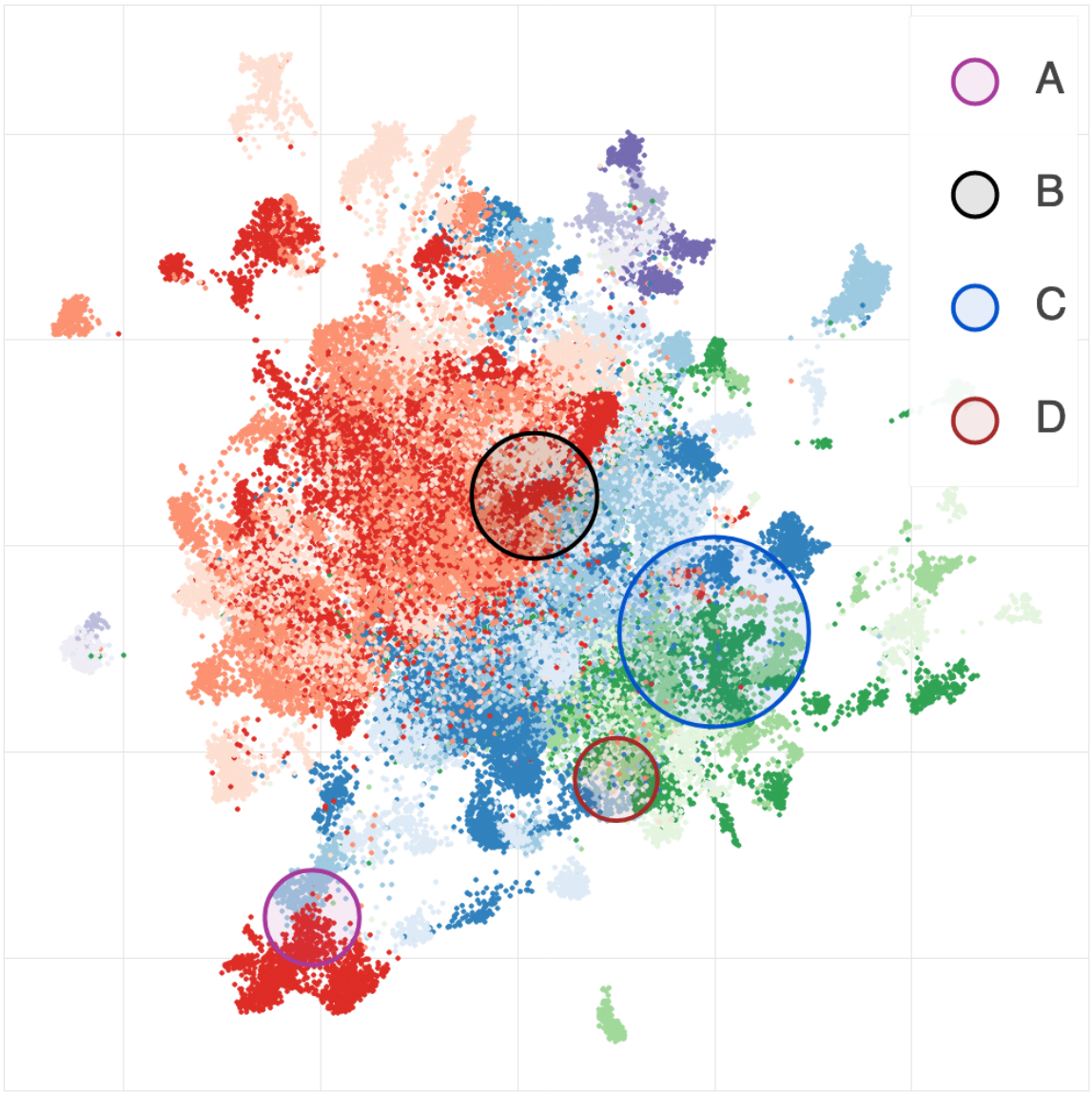


Figure 5.7.1: UMAP visualization of cross dataset embeddings for the top 200 authors, one hue per market. Circles denote the same user in two different markets.

SR, etc.). The circles in the figure highlight the top four pairs of users (top candidate sybils) with a frequent near neighbor from a different market. We find that each of these pairs can be verified as sybil accounts, either by a shared username (A, C, D) or by manual inspection of posted information (B). Note that none of these pairs were pre-matched using PGP - none were present in the high-precision matches. Thus, SYSML is able to identify high ranking sybil matches reflecting users that migrate from one market to another.

5.8 Ethical Considerations

The research conducted in this study was deemed to be *exempt research* by the Ohio State University's Office of Responsible Research Practices, since the forum data is classified as 'publicly available'. Darknet forum data is readily available publicly across multiple markets (Branwen et al., 2015; Munksgaard and Demant, 2016) and we follow standard practices for the darkweb (Kumar et al., 2020) limiting our analysis to publicly available information only. The data was originally collected to study the prevalence of illicit drug trade and the politics surrounding such trades.

Limiting Harm To the best of our knowledge, the collected data does not contain leaked private information (Munksgaard and Demant, 2016). Beyond relying on the exempt nature of the study, we also strive to take further steps for minimizing harms from our research. In accordance with the ACM Code of Ethics and to limit potential harm, we carry out substantial pre-processing (§5.3) to remove links, images, and keys that may contain sensitive information. Towards respecting the privacy of subjects, we do not connect the identity of users to any private information; our method serves only to link users across markets. Further, in this study, we restrict our analysis to darknet markets that have been inactive for several years. The darknet market community has itself taken steps over the past few

years to link identities of trustworthy members across market closure via development of information hubs such as Grams, Kilos, and Recon (Broadhurst et al., 2021). Our efforts aim to understand the formative years that lead towards this centralization.

Inclusiveness Our methods do not attempt to characterize any traits of the users making the posts. Based on our analysis, the datasets contain posts in English, German, and Italian. Thus, our methods may be limited in applicability and biased in performance for languages belonging to these and related Indo-European languages.

Potential for Dual Use Our goal is to understand how textual style evolves on darknet markets and how users on such markets may misuse them for scams and illicit activities. This digital forensic analysis can be put to good use for understanding trust signalling on these markets. We understand the potential harm from dual use; stylometric methods could be used for the identification of users who may not want their identity to be made public, especially when they are subject of hostile governments. We believe that making the information about the existence of such stylometric advances public and providing prescriptions for avoidance techniques (§5.7.1) would aid users who may not know of strategies that they can use to preserve their anonymity. Existing work (Noorshams et al., 2020; Andrews and Bishop, 2019) has already expanded the use of stylometry to the open web. Thus, we have made the analysis of patterns that lower stylometric identifiability one focus of our case study.

5.9 Conclusion

We develop a novel stylometry-based multitask learning approach that leverages graph context to construct low-dimensional representations of short episodes of user activity for authorship and identity attribution. Our results on four different darknet forums suggest that both graph context and multitask learning provides a significant lift over the state-of-the-art.

In the future, we hope to evaluate how such methods can be levered to analyze how users maintain trust while retaining anonymous identities as they migrate across markets. Further, we hope to quantitatively evaluate the migration detection to assess the evolution of textual style and language use on darknet markets. Characterizing users from a small number of episodes has important applications in limiting the propagation of bot and troll accounts, which will be another direction of future work.

Chapter 6: Towards Robust Author Representations

In the previous chapter, we explored the applicability of graph structure in augmenting author identification models. However, we were limited to generalizing across different forums and across time. In this chapter, we will explore the applicability of models trained for author representation learning on large, clear web datasets. We will first quantify whether such models can be used to improve author identification on darkweb forums. Further, we will evaluate the limitations of models trained on large, clear web datasets generalizing across time and demographics. We investigate the research questions using the LUAR model (Rivera-Soto et al., 2021) as the base author representation model, described in the following text. The first part of this work was also presented as a conference talk at the Cambridge Cybercrime Centre's Sixth Annual Cybercrime Conference (Maneriker et al., 2023b).

6.1 Universal Author Representations: Architecture

Figure 6.1.1 shows the architecture for LUAR (Rivera-Soto et al., 2021), the base author representation model used for exploring the research questions in this chapter. For an author A, given all the texts $T_A = \{t_1, \ldots, t_{N_A}\}$ written by the author, each text is tokenized using the tokenizer corresponding to the transformer model used. More details about tokenization strategies for different transformer models were provided in Section 2.4.1.1. An *episode* is created by sampling contiguous windows of l tokens from w texts. These windows are

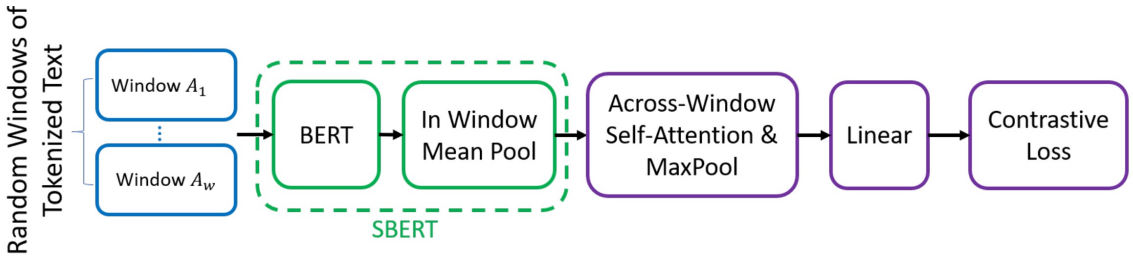


Figure 6.1.1: Architecture for LUAR (Rivera-Soto et al., 2021)

labeled $A_1, \ldots, A_w$. Each individual window is then encoded using a sentence transformer model (Reimers and Gurevych, 2019). A mean pooling operation is applied to the embeddings output by the final layer of the transformer model to generate a single vector representation for each window. An additional mean-pooling operation is used to combine the representations of all windows to generate a single representation for the episode. This representation is then transformed using a linear transformation and the final representation generated from this operation is used as the representation for one *episode* for each author. A supervised contrastive training loss (Khosla et al., 2020) is used a metric learning loss to ensure alignment between the embeddings of multiple episodes of the same author. Further details about the training process and the loss function can be found in the LUAR paper (Rivera-Soto et al., 2021).

6.2 Tracking User Styles across Clear and Dark Web Forums

The code to reproduce the following analysis is available on Github at the following URL: `https://github.com/pranavmaneriker/ccc_darkweb_stylometry`.

Dataset	# Authors	# Posts	# Subforums
Dread	43,629	294,596	382
The Hub	8,243	88,753	62
Reddit-201801	4,413,757	82,531,775	94,945
Reddit-201912	7,439,040	126,992,546	155,864

Table 6.2.1: Dataset statistics prior to preprocessing for comparing LUAR on clear and dark web forums.

6.2.1 Motivation

In chapter 5, we described an architecture utilizing text CNNs (Kim, 2014) for generating the textual component of representations for authorship attribution on darknet forums. Recent work on generalizing authorship representations has focused on a variation of the popular sentence transformer architecture (Reimers and Gurevych, 2019). Specifically, Rivera-Soto et al. (2021) compared the transferability of author representation learning models between Amazon reviews, fanfiction short stores, and Reddit comments. They found that in a zero-shot setting, i.e., without any addition in domain data, the models trained on Reddit data had the highest degree of generalization to new domains. This work motivates us to explore the generalization capabilities of models trained on clear web data to darkweb forums. We explore two research questions. First, can we apply author representation models trained on Reddit forum data directly to Darkweb forums? Second, can we combine data from the darkweb and clear web to build better models?

6.2.2 Datasets

To answer the research questions, we collect datasets from both clear and dark web forums. For the clear web, we sample data from the Pushshift Reddit corpus (Baumgartner et al., 2020). The LUAR model is trained on Reddit data sampled from the same corpus, but

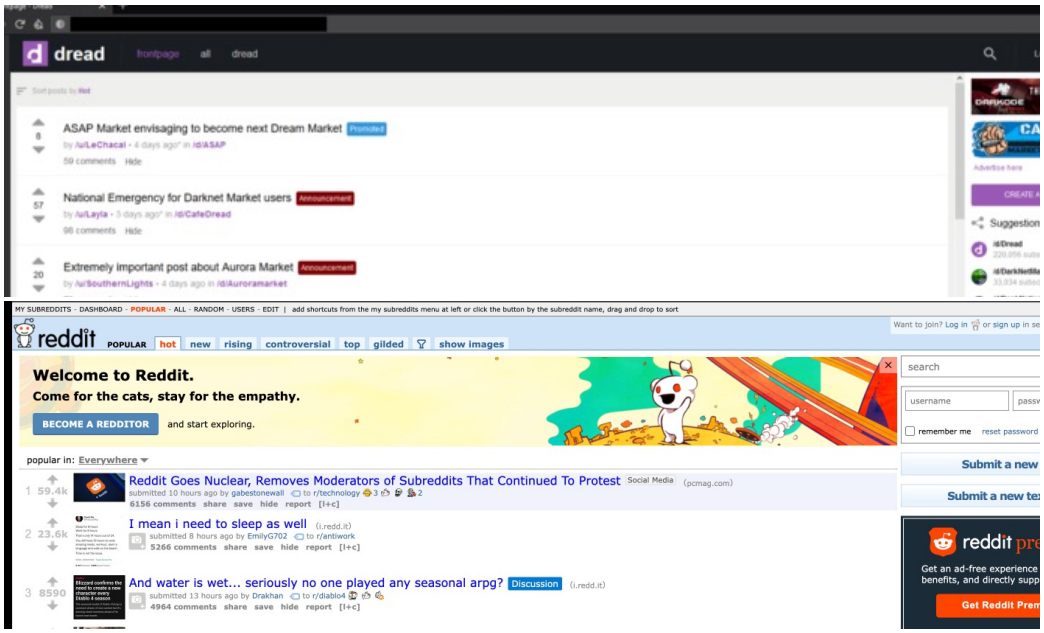


Figure 6.2.1: Dark web market Dread (top) and clear web market Reddit (bottom). Dread image source: Commons (2023)

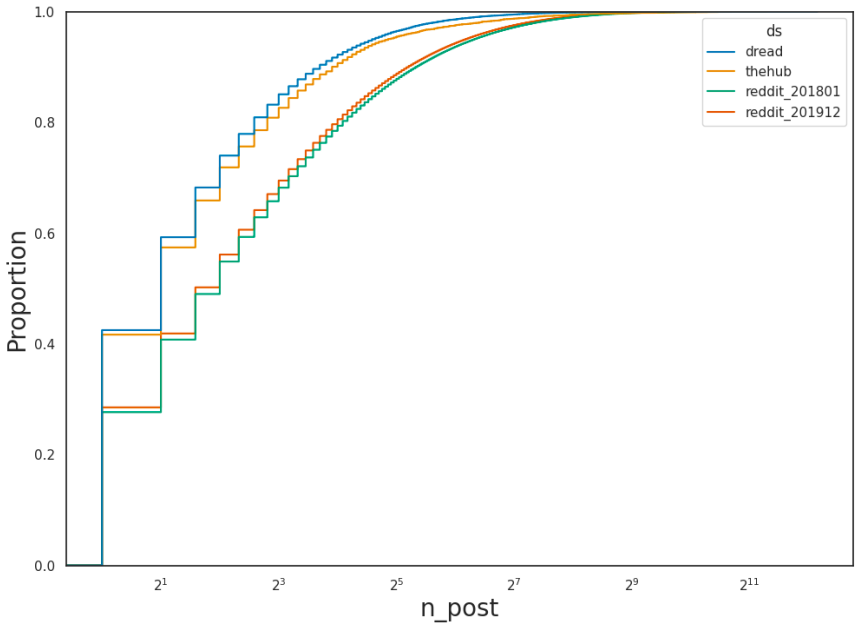
collected between 2015 and 2016. To avoid any overlap with the training data, we sample data from 2018 and 2019. We created two datasets, one for posts sampled from January 2018 and the second for posts sampled from December 2019. Reddit is an 'omni-forum' (Munksgaard and Demant, 2016), where users can participate in a number of subcommunities (subreddit). The similarity between Dread and Reddit is illustrated in Figure 6.2.1. Keeping this in mind, we focus on 'omni-forums' from darkweb data. We collected datasets provided in the CrimeBB collection (Pastrana et al., 2018) and sample data from 'Dread' (the dark web version of Reddit) and 'TheHub'. The data from 'Dread' is collected between February 2018 and January 2020, and 'TheHub' is collected between January 2014 and August 2019. Summary statistics for the unprocessed datasets are provided in Table 6.2.1. As with prior analyses from chapter 5, we assume that each username corresponds to a unique author.

6.2.2.1 Characterizing Author Behaviors

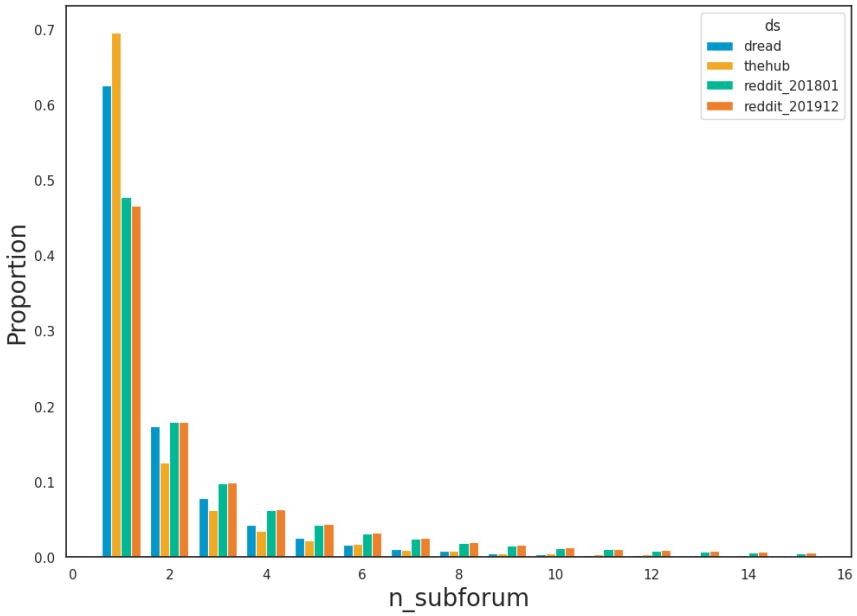
As a first step to understanding the differences in the datasets, we aim to characterize the authors. Figure 6.2.2 provides two figures that capture distributions that characterize the user behaviors. Figure 6.2.2a shows the cumulative distribution function of the number of posts per author. We observe that Reddit has a significantly smaller proportion of authors with only one post. This indicates that there are a larger proportion of authors posting on Darkweb forums with only a single post. This may indicate that users create *throwaway accounts* (Leavitt, 2015) more frequently on the dark web as they desire greater anonymity. Alternatively, this may indicate that the throwaway accounts that exist on Reddit get deleted before they get captured in the intervals in-between Pushshift dataset (Baumgartner et al., 2020) captures. (Changes captured between two consecutive scrapes are not reflected in the dataset). If an author on Reddit deletes their account, all of their posts would be reflected as posts by an author with associated username `[deleted]`. Figure 6.2.2b shows the histogram of the number of subforums a user has posted in. This may indicate that either the authors on Reddit may have interests in diverse topics, or that the granularity of subforums is higher on Reddit. Thus, even from a summary statistics perspective, there are fundamental differences in author behaviors across the clear and dark web forums. However, it is unclear how these differences at the macro level will affect the generalization capabilities of LUAR.

6.2.2.2 Preprocessing and Setup for Author Identification

We divide each dataset temporally into a 70-30 split. That is, the first 70% of each dataset is used for training models for experiments and the final 30% is used for evaluation. The evaluation set is further split into half for constructing a set of query and targets for each author. Thus, there are three splits for each dataset labeled 'train', 'test_query', and



(a) Empirical Cumulative Distribution Function showing the proportion of authors having n_post posts. Reddit has a much smaller proportion of authors with only one post.



(b) Histogram of the number of subforums an author has posted in. Reddit has fewer authors posting on only one subforum.

Figure 6.2.2: User behaviors on Reddit, Dread, and TheHub.

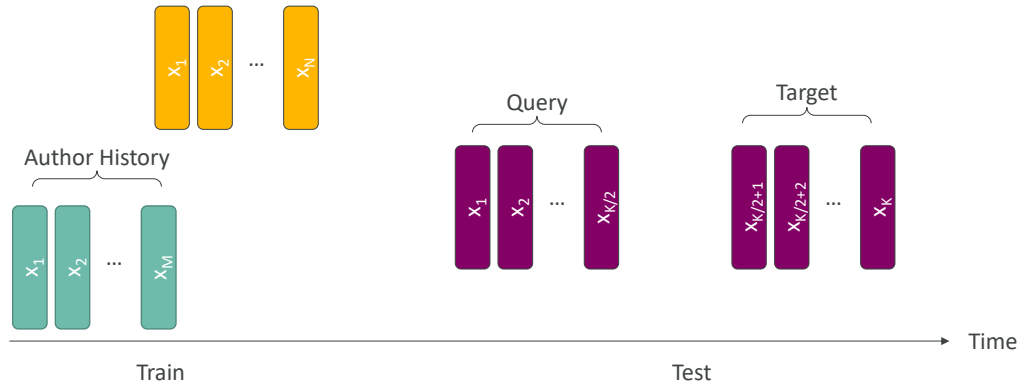


Figure 6.2.3: Setup of splits for the Author Identification task. Each color represents a different author.

'test_target'. The models will be evaluated on their ability to match the representation for an author using an episode from the query set against the corresponding one in the target set. Figure 6.2.3 provides a visual representation of the setup for the author identification task. We filter the posts to include authors with at least 2 posts and a maximum of 1500 posts. Further, to control for the significantly higher number of users in the Reddit datasets, we sample authors from Reddit to ensure that there is an equal number of authors in the Dread dataset and each Reddit dataset. Figure 6.2.4 shows the number of posts and authors in each dataset and split after preprocessing.

6.2.3 Results

For the following results, we set the episode length to 4. We consider sequence lengths (number of tokens sampled per window) at 32 and 64.

6.2.3.1 Zero-shot Transfer

We first evaluate the original LUAR model `LUAR-orig` in a zero-shot setting. That is, we evaluate the model on the test set from the Reddit-201801 and Reddit-201912 datasets,

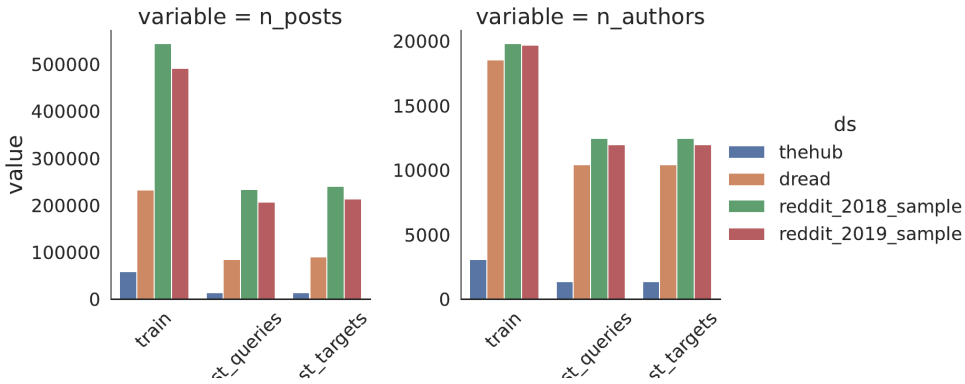


Figure 6.2.4: Number of authors in each dataset after preprocessing.

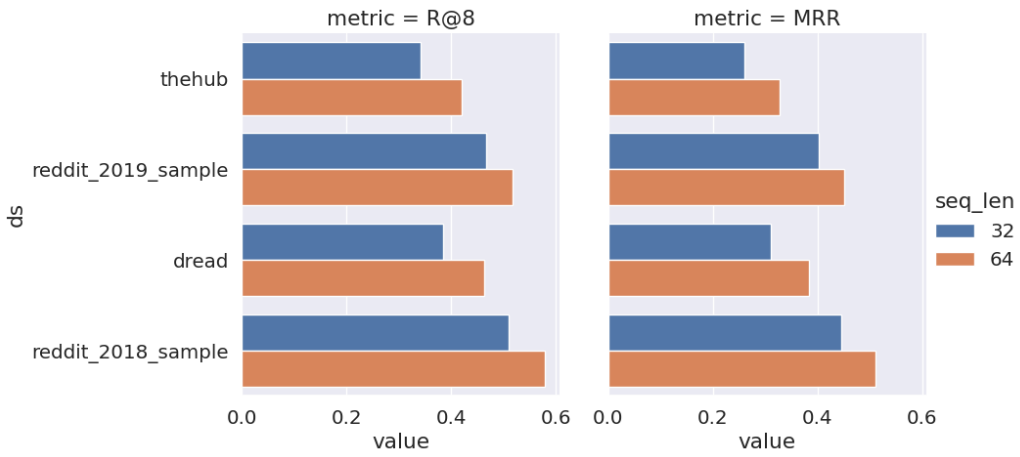


Figure 6.2.5: Zero-shot performance of LUAR on the test set from Reddit-201801, Reddit-201912, Dread, and TheHub. seq_len denotes the number of tokens sampled in each window.

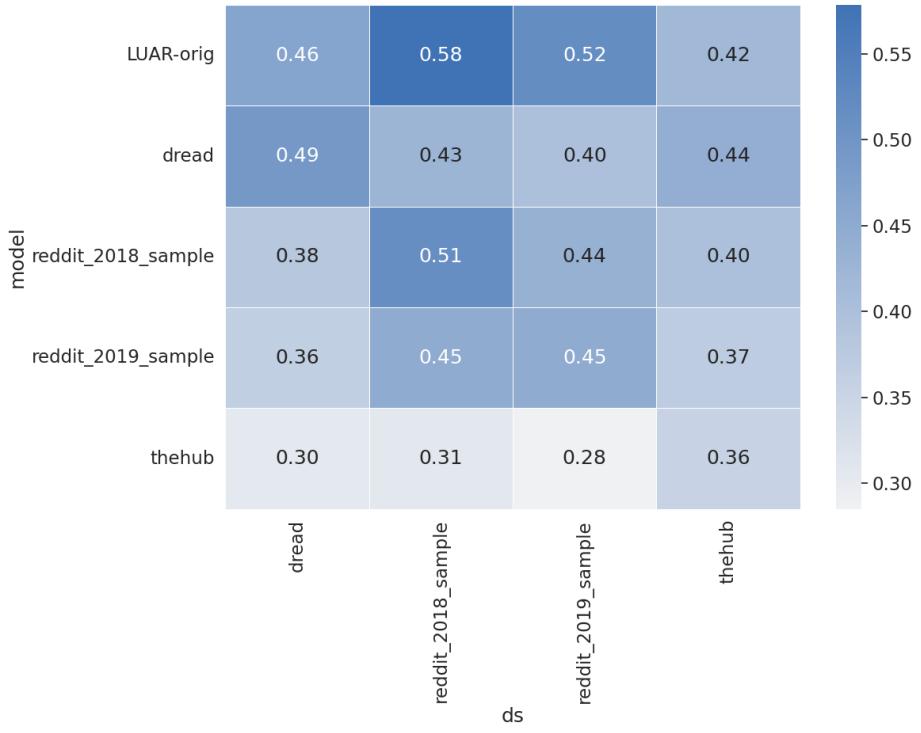


Figure 6.2.6: Heatmap comparing Recall@8 across models. Each row represents the training dataset used for training the LUAR model, while each column represents the test dataset.

and the Dread and TheHub datasets without any additional tuning. Figure 6.2.5 shows the results of this experiment. We evaluate the Recall@8 and Mean Retrieval Rank (MRR) for the LUAR model (see Section 5.5 for definitions of these metrics). The results show that the LUAR model performs well on the Reddit datasets (R@8 > 0.5 for a sequence length of 64), but has a significant drop in performance on the Dread and TheHub datasets. This indicates that a zero-shot transfer of the LUAR model from Reddit to Dread and TheHub is not sufficiently effective. For the remainder of the experiments, we use the sequence length of 64 as it provides the best performance on the Reddit datasets.

6.2.3.2 Darkweb vs Clearweb Models

To better understand the impact of different datasets, we train a separate LUAR model on the training split of each of the datasets. The heatmap in figure 6.2.6 shows the results of this experiment. Each row corresponds to the dataset used to train the LUAR model, and each column corresponds to the test dataset. The first row corresponds to the results from the original LUAR model trained on one year of Reddit data. We find that the Dread-based training dataset is significantly better than the Reddit-based training datasets for generalizing to stylometry on Darkweb data. In fact, the performance of the model trained on Dread generalizes better to TheHub than the model trained on one full year of Reddit data. At the same time, we note that the Dread-based model falls short of the Reddit-based models on the Reddit datasets. This motivates us to try a hybrid approach, where we train the LUAR model on a combination of Reddit and Dread data.

6.2.3.3 Hybrid Dataset: Combining Clear and Dark Web Data

Finally, in Figure 6.2.7, we evaluate the performance of the LUAR model trained on a combination of Reddit and Dread data. The R@8 for this model improves upon the R@8 of the models trained on Reddit or Dread alone in its ability to generalize to TheHub. This supports our hypothesis that combining data from the clear and dark web can lead to better generalization capabilities for author identification models.

6.2.4 Discussion

The results of the experiments show that the LUAR model trained on Reddit data alone does not generalize well to darkweb datasets (Dread and TheHub). However, creating a hybrid dataset by combining data from Reddit and Dread leads to better generalization capabilities. This supports our thesis that combining data from the clear and dark web can

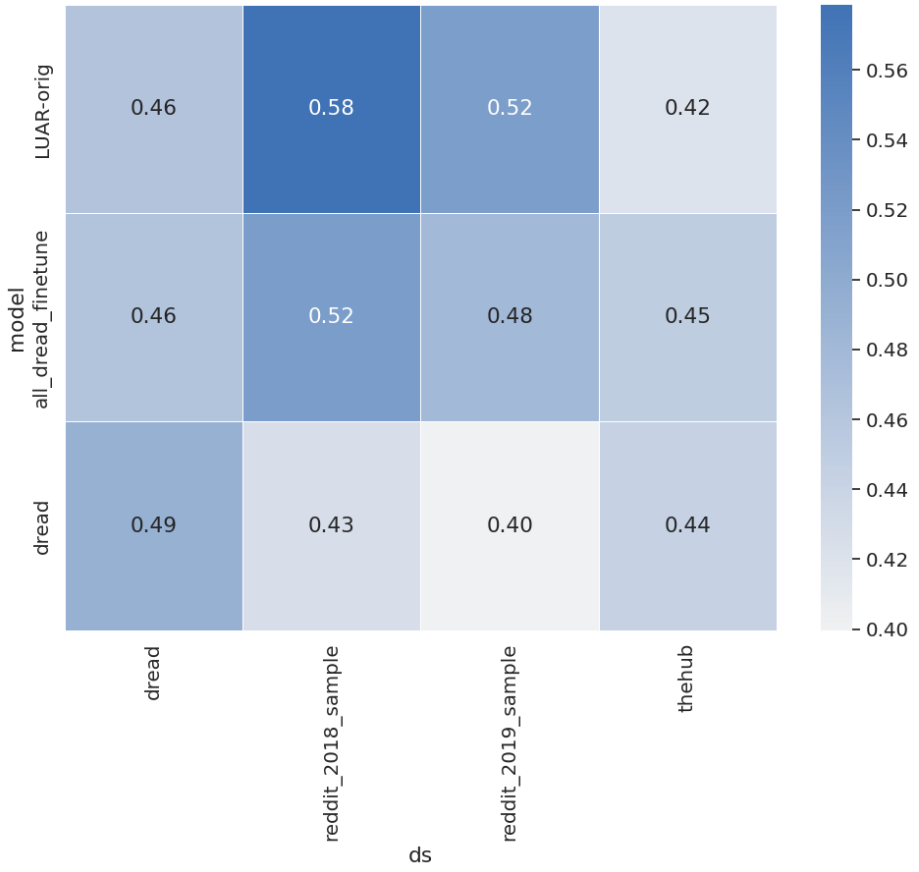


Figure 6.2.7: Heatmap comparing Recall@8 across models with a combined dataset and individual datasets.

lead to better generalization capabilities for author identification models. Further work in this direction could explore the impact of different proportions of clear and dark web data on the generalization capabilities of author identification models.

6.3 Robustness and Generalization within a Domain

An interesting observation from Figure 6.2.5 is that the performance of the LUAR model on the Reddit data is worse on the dataset collected in 2019 compared to the dataset collected in 2018. In light of this observation, in this section, we are concerned with evaluating the impact of temporal data drift, latent author demographic attributes, and their interaction on authorship attribution. We find that both the time elapsed between writing samples and latent demographic attributes can have a significant impact on performance, which we attribute to temporal data shifts. Further, we find that these shifts are more significant for certain groups, notably younger authors whose style evolves over time. This is problematic since these groups may suffer from higher error rates and suffer potential negative outcomes, such as false attributions in forensic applications. Our experiments suggest this degradation in performance is due to fundamental data shifts, rather than model estimation error, which motivates us to propose a recalibration-based mechanism to improve the robustness of authorship attribution models as future work in Chapter 7.

6.3.1 Motivation

The objective of contrastive training is to learn a mapping from an input space—writing samples in our case—to a lower dimensional vector space wherein distance between feature vectors implies a measure of semantic similarity. For example, supervised contrastive learning, which is employed by several models evaluated in this work, uses labels associated with each example to "pull" representations sharing the same semantic label closer together, while

"pushing" representations for examples with different semantic labels further apart (Khosla et al., 2020).

For authorship representation learning, we use author labels as supervision, and we interpret the vector similarity as a measure of the likelihood that two writing samples have the same author. The contrastive training objective can be thought of as attempting to enforce certain *invariances* on the learned representations. In the case of authorship, we desire representations that are invariant to text attributes that exhibit large variance for a single author, such as the particular topics being written about, while capturing stable author features such as writing style, which are more constant over time (Andrews and Bishop, 2019). However, although recent work has successfully improved performance of author representations in downstream tasks, such as social media account linking, for example by training on datasets comprising millions of anonymous authors (Khan et al., 2021), their limitations remain poorly understood. Section 6.2 describes some limitations of these models in their domain generalization capabilities.

As another step to better understand these limitations, in this section we focus on two central questions. First, do author representations capture representations that allow author identification to generalize across time? Through careful experiment design, we find that the degradation in author identification performance may originate from temporally evolving styles rather than model estimation error. Second, do author representations encode systematic biases associated with author demographics? By biases, we refer to the fact that these models may be more likely to incorrectly identify authors from certain demographic groups, which may negatively impact downstream decisions against authors from these groups. For example, authors who are teenagers may be negatively impacted by incorrect moderation decisions as compared to authors who are adults. We answer this question in the

affirmative, finding that authorship attribution performance degrades significantly across demographic groups, including age and gender.

6.3.2 Datasets

We start from the Pushshift Reddit dataset (Baumgartner et al., 2020) generating the different splits. We restricted the time period of the data from January 2015 to November 2019. In each setting, we follow a query/target temporal setup similar to previous work on retrieval-based authorship verification (Andrews and Bishop, 2019; Khan et al., 2021). That is, we first selected a set of users and then include all the posts written by each selected user across two non-overlapping time periods. The author identification task in such a setting takes a user's posts from the query time period as input and outputs a list of users ranked by their likelihood of matching the query user using their posts from the target time period. A positive match requires having a top/high rank for the posts by the same user during the target time period. We constructed multiple such datasets to help quantify the robustness of authorship attribution models. Specifically, we created two types of datasets: TemporalReddit and DemographicReddit.

6.3.2.1 TemporalReddit

In each dataset, we sampled users having between $p_{\min}$ and $p_{\max}$ posts in both the query and the target period. Specifically, we selected query/target temporal pairs with a fixed time difference between them. Suppose the queries span time period $(Q_{\text{start}}, Q_{\text{end}}) = (q_1, q_2)$ and the targets span $(T_{\text{start}}, T_{\text{end}}) = (t_1, t_2)$. We ensured that $\Delta_\tau = q_2 - t_2$ was similar across all datasets. However, we varied q_1 to obtain multiple, non-overlapping datasets. We chose queries from January 2015 to January 2019 (5 query sets, each one month long) and targets from December 2015 to December 2018 and October 2019 with $p_{\min} = 16, p_{\max} = 2000$ with

group	fraction	count
adult	0.52	1575
teenager	0.39	1195
senior	0.02	66
middle-aged	0.05	164

Table 6.3.1: Distribution of demographics for age groups in **DRAge**.

gender	fraction	count
f	0.51	614
m	0.49	586

Table 6.3.2: Distribution of demographics for gender in **DRGender**.

a set of 50k users sampled for each query-target pair. In the following texts, we refer to these datasets as **TRFixed**. A degradation in performance of author identification models over **TRFixed** would suggest that model estimation is not stable to temporal data shifts, and that the model would need to be retrained to maintain performance over time. For the second dataset, we selected a single, fixed query split and multiple target splits. The query period chosen was Jan 2015, and the target periods chosen were Dec 2015–2018, Oct 2019. We sampled a set of 50k users having $p_{\min} = 16, p_{\max} = 2000$ in the query as well as in each target split. We refer to this dataset as **TRVariable**. A degradation in performance in **TRVariable** would indicate that temporal changes of author style lead to a degradation in performance of author identification models.

6.3.2.2 DemographicReddit

We used the RedDust dataset (Tigunova et al., 2020) to collect self-identified demographic attributes associated with Reddit users. Specifically, we investigated two latent demographic attributes (age and gender) and their correlation with author identifiability. We followed the authors' original definition for categorizing users into age groups (13-23: Teenager, 24-45: Adult, 45-65: Middle-Aged, 65+: Senior). To create each of these datasets, we first subset the data to include only those users who lie in the intersection of our subset of the Pushshift dataset with RedDust-Age and RedDust-Gender. Following this, we selected a sequence of consecutive monthly splits from these intersecting users having $p_{min} = 8$ for at least five consecutive months. We restricted the users to be present across all the splits to ensure that the demographics do not change over splits. Furthermore, to control for temporal variation, we only considered query/target pairs from consecutive months, i.e., $T_{start} - Q_{start} = 1$ month. We select a sequence of months that ensure that we can maximize the number of users with known demographics from RedDust. The selected splits under this constraint corresponded to the months of January to May 2019. We sampled 3k users' posts from this period with known age, giving us **DRAge**, and 1.2k users' posts with known gender, **DRGender**, derived from RedDust-Age and RedDust-Gender, respectively. Note that the age in the data was adjusted to reflect the age of the user in January 2019 for all splits.

6.3.2.3 TDReddit

In addition to constructing query/target pairs with a fixed $T_{start} - Q_{start} = 1$ month, we can additionally consider all possible pairs of query/target from the splits collected for creating the DemographicReddit datasets. These (ordered) tuples correspond to ${}^5P_2 = 20$

query/target pairs. The two corresponding demographic datasets are labeled **TDRAge** and **TDRGender** for age and gender, respectively.

6.3.3 Experiments

6.3.3.1 Baselines

We describe the models used and the rationale for the choice of these models in the following text. The first three models correspond to count-based baselines that do not use neural networks for representation learning. The final three models correspond to neural network-based models that use different representation learning model for each window, and these representations are merged in a LUAR like setup (Figure 6.1.1). Similar baselines have been used in other work on author identification (Wang et al., 2023; Soto et al., 2024).

Count-Subreddit (CS): The CS model uses a count based representation of each author, where each entry of the of the author representation vector represents the number of times the author posted on the corresponding subreddit. In particular, we use the CountVectorizer from Pedregosa et al. (2011). Subreddit usually represent a topic of interest, so this vector could be seen as a simple proxy for the topics the author is interested in.

TFIDF-Subreddit (TFS): The TFS model uses TFIDFVectorizer from scikit-learn with the subreddits that an author posts in as the 'term'. This is another compact representation of the topics of interest for an author.

TFIDF-Text (TFT): The TFT model uses all the text posted by an author coverts it into a TF-IDF vector (also using the TFIDFVectorizer) over the text posted by each author as input. This vector captures both topics, as well as some aspects of style such as word usage frequencies.

The hyperparameters for the aforementioned models were chosen using a grid search over the hyperparameter space with a separate split of the query/target data.

SBERT The popular sentence transformer model SBERT (Reimers and Gurevych, 2019) is used to generate the representation for each window of text for an author. The representations from this model may capture the semantics and syntax associated with the text posted by the author.

STEL The STEL model from Wegmann and Nguyen (2021), which is a modular framework to represent linguistic style while controlling for content, is used to generate the representation for each window of text for an author. This model focuses on features that capture writing style only.

LUAR This is the LUAR model from (Rivera-Soto et al., 2021). This model is trained to represent time invariant features of an author's writing style.

6.3.4 Evaluation and Discussion

6.3.4.1 TemporalReddit

method Q/T	CS	TFS	TFT	SBERT	STEL	LUAR
15-1/15-12	0.237	0.260	0.192	0.232	0.135	0.779
16-1/16-12	0.265	0.284	0.212	0.251	0.148	0.796
17-1/17-12	0.278	0.297	0.224	0.262	0.137	0.733
18-1/18-12	0.283	0.303	0.232	0.281	0.171	0.770
19-1/19-10	0.316	0.343	0.235	0.290	0.143	0.760

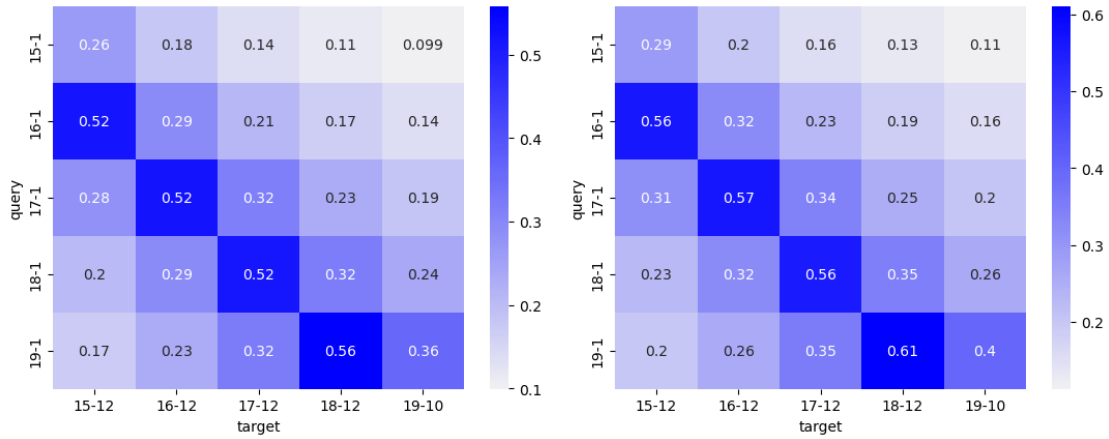
Table 6.3.3: Recall@8 results across different models on **TRFixed**. The leftmost column represents the Query/Target period.

Table 6.3.3 shows the evaluation results of recall@8 for **TRFixed**. LUAR significantly outperforms all other models across all fixed time periods. As expected, the TFIDF and count based models, which do not require explicit 'training' for data in a specific time period, have relatively stable author identification performance over time. The results from the SBERT/STEL models, which are not trained without controlling the time periods for the training dataset, also show a similar trend. As LUAR is trained on data from 2015-2016, if the author representations were overfit to identify authors in this time period, we would expect a significant drop in performance for the later time periods. The last column on the right of Table 6.3.3 shows that this is not the case, and LUAR's performance is relatively stable across time. Thus, for a consistent query-target time difference, LUAR is able to maintain a high level of performance across time periods, indicating that model retraining is not necessary to maintain performance over time.

Next, using **TRVariable** we consider temporal variation across a fixed set of users across time. Figure 6.3.1 shows heatmaps for the recall@8 results for each model across different query/target time periods for a fixed set of authors. We see that across all the methods, there is a temporal degradation in performance, which could potentially be caused by temporally evolving styles (STEL), topics of interest (Count/TFIDF) based results or semantics (SBERT). However, from Figure 6.3.2, we see that when normalized for diagnal, degradation in performance is the least for LUAR. This indicates that the author representation features captured by LUAR are more stable across time compared to the other models.

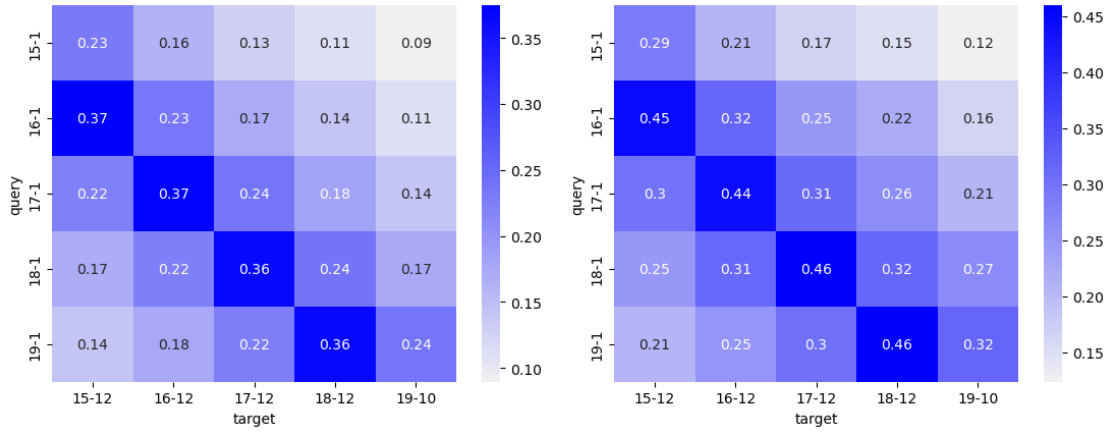
6.3.4.2 DemographicReddit

First, we consider the results for the demographic splits controlled for temporal variation. Figure 6.3.3 shows the recall@8 results for age groups across different models on **DRAge**.



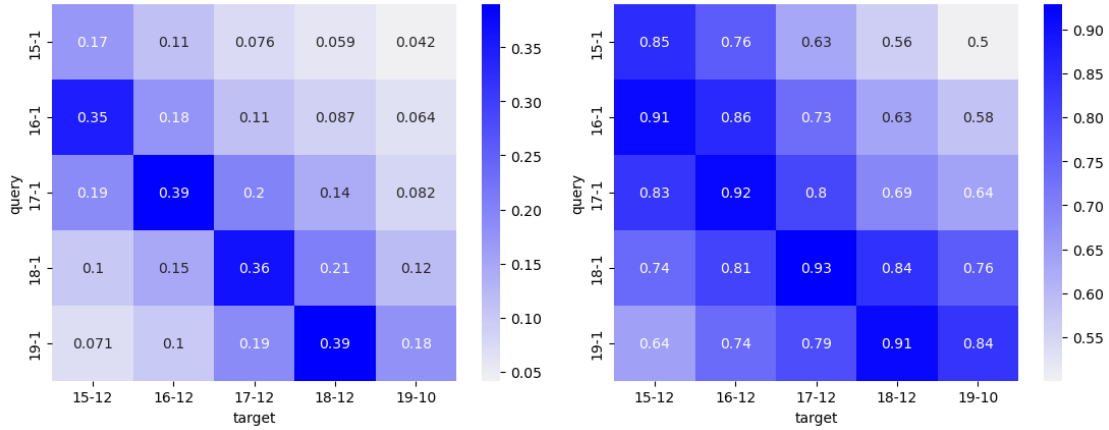
(a) CS

(b) TFS



(c) TFT

(d) SBERT



(e) STEL

(f) LUAR

Figure 6.3.1: Recall@8 results on **TRVariable**

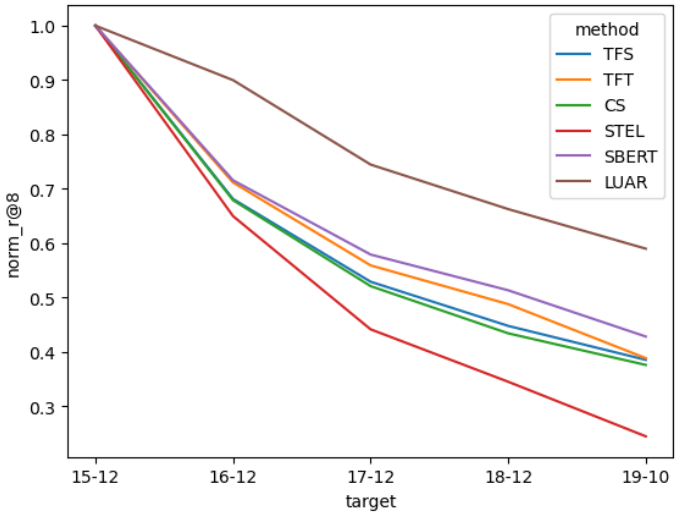


Figure 6.3.2: Target results for the earliest query split (15-1). We compare normalized recall@8 across all methods for **TRVariable**.

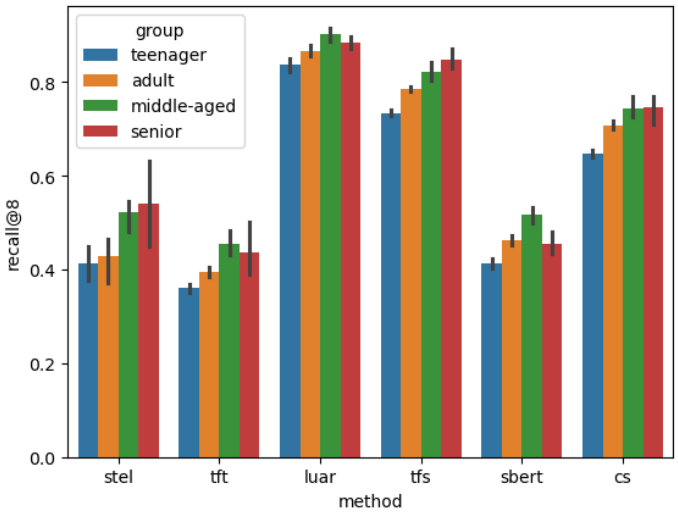


Figure 6.3.3: Overall results on **DRAge** split by age group.

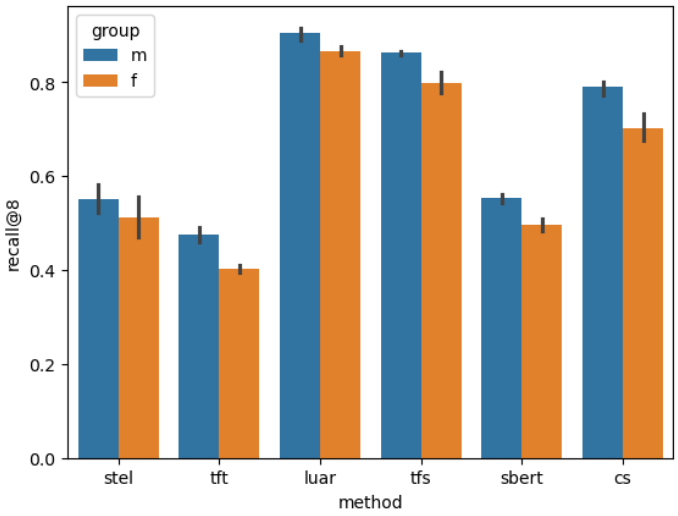


Figure 6.3.4: Overall results on **DRGender** split by each group (bottom).

We note that all models consistently underperform on (self-identified) teenage users. Further, from Figure 6.3.4, we note that all models consistently underperform on (self-identified) female users.

Next, we consider **TDRAge** and **TDRGender**. We focus on LUAR, as it is the best performing model across all the datasets. The results from Figure 6.3.5 that across all $|T_{\text{start}} - Q_{\text{start}}|$, (self-identified) teenage authors are consistently the least identifiable. Further, from Figure 6.3.6, We note that across all $|T_{\text{start}} - Q_{\text{start}}|$ and all models, authors self identifying as female are consistently the least identifiable.

6.3.4.3 Analysis

From the results with TemporalReddit, we can conclude that model estimation is unlikely to be a cause for the degradation in performance over time. However, changing author interests and writing styles do contribute to the degradation in performance. Drilling down into the specifics, we find that the degradation in performance is more significant for certain

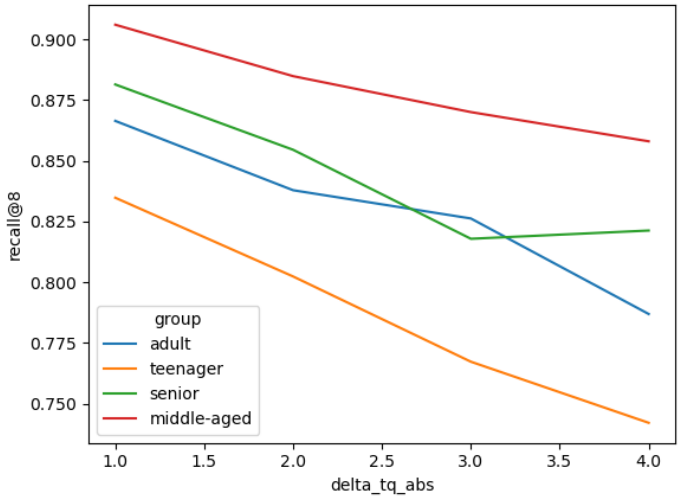


Figure 6.3.5: Overall results on **TDRAge**. The x-axis denotes the absolute difference in the query and target start time, i.e., $|T_{\text{start}} - Q_{\text{start}}|$.

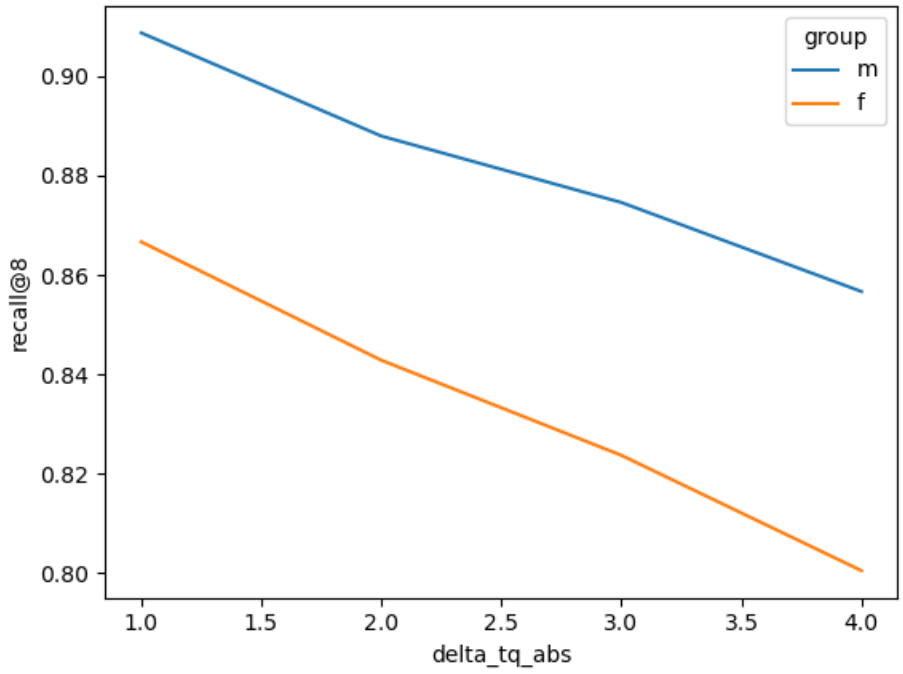


Figure 6.3.6: Overall results on **TDRGender**. The x-axis denotes the absolute difference in the query and target start time, i.e., $|T_{\text{start}} - Q_{\text{start}}|$.

groups, notably younger authors and female authors. Our results support our hypothesis that author representations encode systematic biases with respect to demographics. These biases get further exacerbated across time. Care must be taken when deploying authorship attribution models in forensic applications, as the models may suffer from higher error rates and potential negative outcomes, such as false attributions, for certain the aforementioned groups.

6.3.5 Limitations

Tigunova et al. (2020) carefully extracted self-identified demographic traits while excluding posts made by users on subreddits involving gamin/roleplaying as authors tend to adopt different personas in such subreddits. However, our analysis presupposes the correctness of their demographic extraction process. The demographic traits are self-identified, and there is no way to verify the accuracy of these traits. This limitation must be noted when interpreting the results from the TemporalReddit and DemographicReddit datasets.

Chapter 7: Conclusions and Future Work

In this preceding chapters, we described structures that help models be more adaptive at the three different stages of their lifecycle. Specifically, we used implicit and explicit structures along with *adversarial testing*, *runtime monitoring*, and *domain adaptation* to build more adaptive machine learning systems. We now describe some directions for extending these techniques for future work.

7.1 Large Scale Structure-aware Authorship Attribution

In this extension, the goal is to test the generalizability of our findings related to the improvements offered by utilizing forum graph structures on the darkweb (Chapter 5). Recent work on cross-domain authorship attribution using text has determined that certain domains (eg. Reddit) are more useful for training authorship attribution models that generalize to other domains (Barlas and Stamatatos, 2020; Rivera-Soto et al., 2021). Specifically, Rivera-Soto et al. (2021) demonstrate that in the source domain, diversity in the expressed topics and larger number of unique users play a role in explaining better transfer to target domains. However, this work does not utilize the additional structure and context present in different domains. We posit that these graphs can provide information orthogonal to that which is already present in the text. In this direction of future work, we aim to demonstrate that even in scenarios with an abundance of text/users (over 100k users/1M text posts), these graph

structures help improve authorship identification within a single domain, and also help better generalize across domains. Chapter 5 shows that this is true in the complementary setting with fewer users ($\approx$ 1-10k) In the remainder of this section, we first describe the choices that are involved in constructing graph structures and their similarities across domains. We then describe some preliminary work on utilizing these structures for authorship attribution on Reddit and share our preliminary findings. To conclude this section, we describe additional directions that we hope to explore, including the associated datasets and techniques.

7.1.1 Graphs in Authorship Attribution

Prior to the prevalence of neural network approaches, seminal work in computational authorship attribution (Stamatatos, 2009) often used syntax-based features including, part-of-speech tags, phrase structures, and syntactic error-based features, among others. These features may also improve neural authorship attribution models but are not the graphs focal to our analysis. Instead, we center our work on online content platforms and identify the organizational structures that they use. A multitude of platforms that involve individuals posting content online have associated graph structures. We focus on structures common to platforms that face potential challenges with content moderation where authorship identification has the potential to play an important role. While individual platforms may differ, there exists a shared, underlying, meta-structure, which can be used to identify patterns that aid in stylometric analyses across domains. In Figure 7.1.1 we show a metagraph for a specific platform but note that the the thread-comment-user structure is shared across other platforms as well. Additional nodes in the metagraph are specific to the platform of interest. In the following section, we focus on a specific platform - Reddit - and utilize its structure to provide preliminary evidence for its potential.

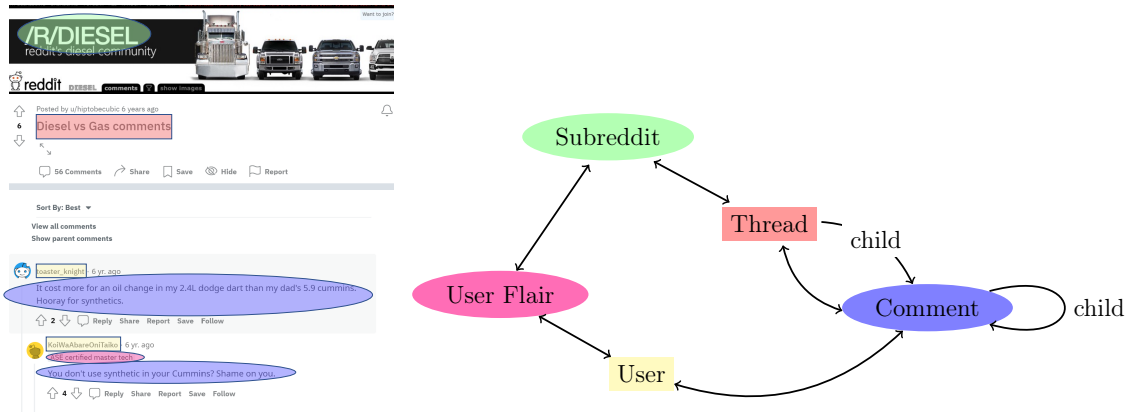


Figure 7.1.1: Metagraph of Reddit used for preliminary analysis.

7.1.2 Preliminary Analysis: Reddit Graph-aware Authorship Identification

7.1.2.1 Dataset

We collected data from a snapshot of Reddit comments over one month (Aug 2016) from the collection released by Baumgartner et al. (2020). Figure 7.1.1 describes the metagraph corresponding to the graph that we construct from this snapshot. We use directed edges to distinguish comments that are direct descendants of the parent comment/thread. Note that the raw data collected includes only the comments posted within the month; thus, for certain comments, the text corresponding to the direct parent thread/comment may be unavailable as they are posted in the previous month. These comments could lead to disconnected components in the graph. To ensure that the graph is connected, we add a bidirectional edge from each comment to the thread where it was posted. Table 7.1.1 shows summary statistics about the graph constructed for this dataset.

Entity	Approximate Count
Subreddits	63,000
Authors	3,250,000
Comments	70,000,000
Nodes	79,100,000
Edges	300,000,000

Table 7.1.1: Dataset used for preliminary analysis of graphs for authorship attribution on Reddit.

7.1.2.2 Goals

We aim to evaluate our proposed approaches for author identification in a retrieval based setup commonly used in this setting in prior work (Andrews and Bishop, 2019; Rivera-Soto et al., 2021; Khan et al., 2021; Maneriker et al., 2021a). That is, we get a sample query text(s) and must retrieve the nearest author from a collection of target texts. The query and target texts that the methods are evaluated on are each collected from a different time periods compared to the training set. This ensures that the embeddings for specific authors are robust to temporal shifts. Ensuring this is particularly challenging in the graph setting as there are multiple mechanisms to construct graphs across different time periods. There may be authors who post in both the training, test query, and test target time periods. In chapter 5, we describe one strategy where we only use subforum embeddings. While they do provides improvements (Sec 5.5), here we propose alternative mechanisms to incorporate further structural information. It is not possible to connect the graphs across different time periods as using the same node to denote the author in disjoint time periods would lead to a trivial embedding solution (unique node embedding for the author). One strategy to deal with the temporal challenge is to use separate graphs for the training and testing time periods, aligning node embeddings across them. This corresponds to the vertex nomination problem

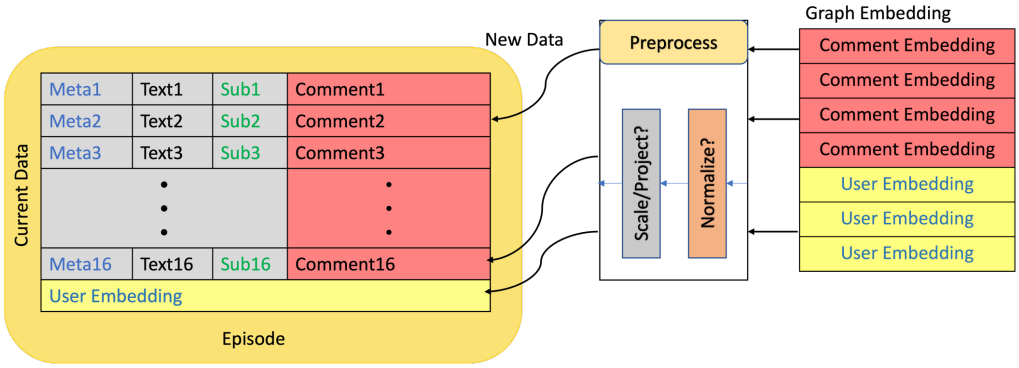


Figure 7.1.2: Structure-based Author Identification Embedding

across multiple graphs, and strategies include using orthogonal procrustes (Agterberg et al., 2020)) for alignment. Alternatively, node attributes such as text, time, and label could be used to construct heterogeneous, attributed graphs. In the latter construction, certain inductive representation learning techniques for large graphs may be applied (Hamilton et al., 2017; Xu et al., 2020). We will now describe our initial explorations with these approaches and follow with potential future directions.

7.1.2.3 Methods

Structure-based

In the first approach, which we designate as the *structure-based* approach, we use separately generate embeddings from the structure of the Reddit graph, use a separate neural network to embed text/metadata in each episode, and then fuse the two embeddings to generate a structure-aware embedding. The graph embedding may need to be transformed/scaled before it is combined with the embeddings of non-graph features to ensure that their magnitudes are comparable. Thus, we transform the embeddings prior to fusing them. Figure 7.1.2 demonstrates the structure-based embedding approach for author identification.

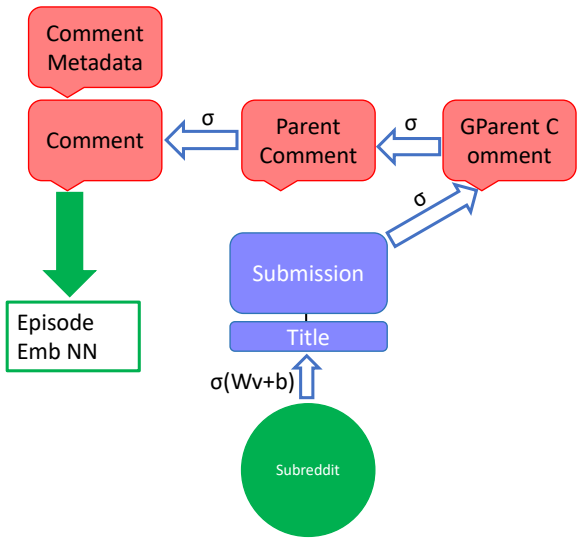


Figure 7.1.3: Context based Author Identification Embedding

Context-based The second approach, designated as the *context-based* approach uses the structure of the Reddit graph to collect surrounding context for each post prior to embedding it. We systematically add the context from parent nodes ((grand)parent comment, thread, subreddit), each of which are individually embedded using a shared text embedding neural network. Each layer of the architecture adds some context before applying a non-linear transformation. Figure 7.1.3 provides a visual representation of the transformations.

Preliminary Results

We test these approaches using SVDs of the adjacency matrix for structure-aware embeddings (à la Agterberg et al. (2020)) and text CNNs for embedding texts. Figure 7.1.4 demonstrates that these directions have the potential for improving author identification even on a large scale dataset.

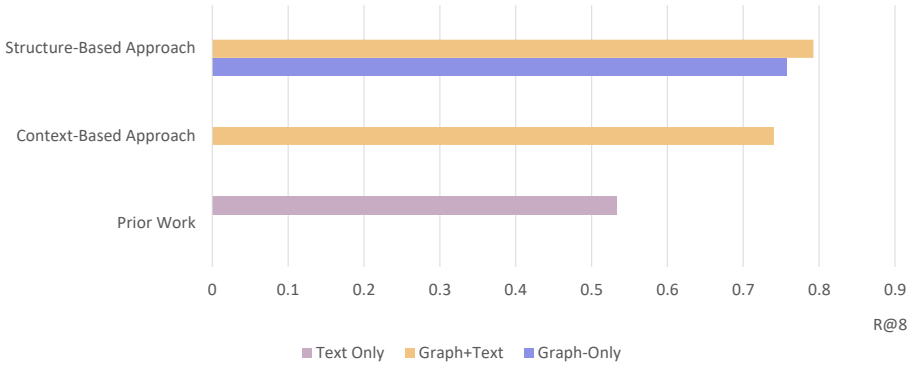


Figure 7.1.4: Results on author identification with preliminary approaches on a validation split. R@8 = recall at 8.

7.1.3 Future Directions

The preliminary results are promising and demonstrate improvements over the text-based baseline. In future work, we aim to explore the impact of alternative structure embeddings, including GNN based (Veličković et al., 2018; Hamilton et al., 2017) and anonymous walk-based embeddings (Ivanov and Burnaev, 2018; Wang et al., 2020b). In particular, successes from scaling GraphSAGE-based (Hamilton et al., 2017; Ying et al., 2018) approaches motivate their potential use in a hybrid fashion, where both the structure and text embeddings may be fused more effectively. Additionally, the results in Figure 7.1.4 are evaluated on a specific validation dataset. Additional experiments need to be carried out to test whether the different stages (embedding, alignment) are affected by temporal shifts. Finally, we aim to test the impact of using structure and context to help in better adaptation across different domains. Similar to our study of domain adaptation for text embeddings (from Chapter 6), we would aim to test whether structure-aware embedding would adapt to authorship identification datasets from Dread and Twitter.

7.2 Fairness through Conformal Prediction

The standard formulation for conformal prediction (Theorem 4) provides a score-based mechanism to control for miscoverage in the prediction sets. However, when dealing with fairness, we may want to reason over guarantees over other expressions that involve the inputs, labels, and predictions. Such guarantees can be considered as a form of thresholded risk Control. In the conformal setting, the Learn Then Test (Angelopoulos et al., 2021a) and conformal risk control (Angelopoulos et al., 2024) frameworks provide a way to control for model risks for specific classes of risk functions. However, the Learn Then Test framework requires an IID assumption over the calibration data. Therefore, in the following section, we will describe how conformal risk control can be used for providing fairness guarantees. Note that we will describe the split CP version of each of these frameworks.

7.2.1 Conformal Risk Control

First, we define the framework of conformal risk control (Angelopoulos et al., 2024). We start from a calibration set $\mathcal{D}_{\text{calib}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$, with exchangeability assumed for $\mathcal{D}_{\text{calib}} \cup \{(X_{n+1}, Y_{n+1})\}$. We have feature vectors $X_i \in \mathcal{X}$ and labels/outcomes $Y_i \in \mathcal{Y}$. The guarantee provided by conformal prediction sets C relates to miscoverage

$$\Pr\left[Y_{n+1} \notin C(X_{n+1})\right] \leq \alpha$$

Instead, consider any bounded loss function L that is monotonically non-increasing with the increasing size of $C(X_{n+1})$. Examples of such functions include accuracy, F1-score, and false negative rate. Conformal Risk Control provides a mechanism to construct prediction sets C that provide guarantees over a bounded loss function $L \in (-\infty, B]$:

$$\mathbb{E}\left[L(C(X_{n+1}), Y_{n+1})\right] \leq \alpha$$

Suppose we have a trained model $\hat{f} : \mathcal{X} \to \mathcal{Z}$, where $\mathcal{Z}$ is the space of raw model outputs which is used to create a function $C_\lambda(.)$ that constructs prediction sets for a given λ. Where larger λ correspond to more conservative prediction sets. For example, for the TPS method discussed in Chapter 4, this would be $C_\lambda(X) = \{y : 1 - \hat{f}(X)_y \leq \lambda\}$. Conformal risk control provides a threshold $\hat{\lambda}$ such that $E[C_{\hat{\lambda}}] \leq \alpha$ for a given $\alpha < B$

7.2.2 Fairness through Conformal Risk Control

We can use conformal risk control to modify the outputs of a model to provide guarantees on any monotone loss function. We provide one example below, constructing a guarantee for an expression capturing predictive parity.

We will build a risk control guarantee for a predictive parity like term. Consider a binary classification task where the label y_1 corresponds to acceptance and y_0 corresponds to rejection. Consider two groups g_0 and g_1. Define

$$L_i(C_{\lambda_i}(X), Y) = \mathbf{1}[y_1 \notin C_{\lambda_i}(X) \cap X \in g_i], i = 0, 1$$

This loss function captures the points where the prediction set excludes acceptance for each group. With increasing λ, C_λ will include additional classes. The possible values of $C_\lambda(X)$ as λ increases, if the acceptance class y_1 has higher score would be $\emptyset, \{y_0\}, \{y_0, y_1\}$, which corresponds to loss values of $1, 1, 0$ respectively. If the rejection class y_0 has a higher score, the possible values would be $\emptyset, \{y_1\}, \{y_0, y_1\}$, which corresponds to loss values of $1, 0, 0$ respectively. In both cases, the loss function is monotonically non-increasing with the size of the prediction set. Thus we can use conformal risk control over this set. From risk control, we would get

$$\hat{\lambda}_0 : E[L_0(C_{\hat{\lambda}_0}(X), Y)] \leq \alpha_0$$

and

$$\hat{\lambda}_1 : E[L_1(C_{\hat{\lambda}_1}(X), Y)] \leq \alpha_1$$

Since L is an indicator function, these guarantees corresponds to $\Pr[y_1 \notin C_{\hat{\lambda}_i}(X) \cap X \in g_i] \leq \alpha_i$. From this risk control and monotonicity of L, we can set $\hat{\lambda} = \max\{\hat{\lambda}_1, \hat{\lambda}_2\}$, and can provide a predictive parity-like guarantee for the model:

$$-\alpha_1 \leq \Pr[y_1 \notin C_{\hat{\lambda}}(X) \cap X \in g_0] - \Pr[y_1 \notin C_{\hat{\lambda}}(X) \cap X \in g_1] \leq \alpha_0$$

Thus, given a calibration set and required thresholds α_0, α_1, we can use conformal risk control prediction sets to provide a fairness guarantee for the outputs produced by any black-box model.

This example shows the utility of conformal risk control to provide fair predictions. We leave the exploration of the full gamut of fairness definitions and possible loss functions as future work

7.3 Towards More Robust Stylometry

In chapters 5 and 6, we study the robustness of authorship attribution models across time and domains. The methods we propose in chapter 5 require retraining a model with additional context provided from graph-based representations. As seen in chapter 6, the representations learned by models trained on one domain do not always generalize well to other domains, or even within the same domain. The robustness of these models across time and demographics varies. We propose two potential directions of future work that are aimed at building more robust authorship attribution models viz. *recalibration* and *conformal prediction*.

7.3.1 Recalibration

A model is said to be well calibrated if the score predicted by the model is a good estimate of the true probability of the event. Specifically, consider a binary classification model that predicts a score s for a given input. For a well-calibrated model,

$$\Pr(Y = 1|S = s) = s$$

The outputs of neural networks are not guaranteed to be well-calibrated (Guo et al., 2017). Exact calibration is difficult to achieve and measure, so binned versions of calibration are often used. Reliability diagrams are used to visualize the calibration of a model, while the expected calibration error (ECE) is used to measure the calibration of a model. This reliability diagram plots the empirical probability of the event against the predicted score across bins. For a well-calibrated model, the empirical probability should match the predicted score, i.e, lie on the line $y = x$. The expected calibration error is defined as

$$\text{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|$$

where B_m is the set of examples in bin m, n is the total number of examples, $\text{acc}(B_m)$ is the accuracy of the model in bin m, and $\text{conf}(B_m)$ is the confidence of the model in bin m. Methods such as temporal scaling, isotonic regression, and Platt scaling have been used to recalibrate the outputs of neural networks (Guo et al., 2017) to achieve calibration.

For the authorship attribution task, we would aim to recalibrate the similarity scores for the representations predicted by the model so that the score is a good estimate of the true probability of the event. We set up this problem as a binary classification task, sampling a matching pair of episodes (same author) from the query and target, and a fixed number of non-matching pairs (negative samples) for each author. For the purpose of this discussion, we set the number of negative samples to 5. Figure 7.3.1 shows the density plot for the pairwise

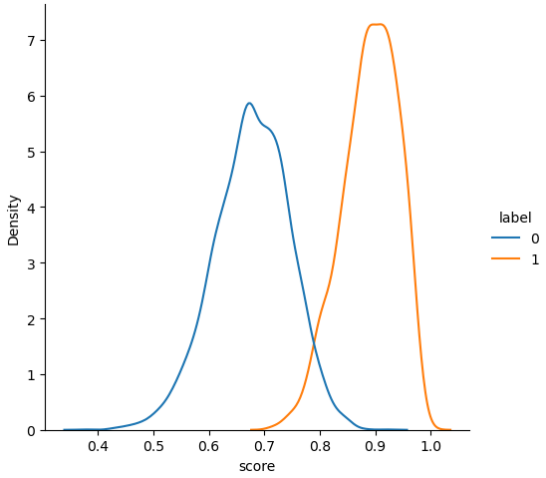


Figure 7.3.1: Density plot for pairwise cosine similarity for authors from the **TRVariable** dataset. 1 corresponds to matching pairs, 0 corresponds to non-matching pairs.

cosine similarity for authors from the **TRVariable** dataset. The density plot shows that the scores are separable, but it is not clear if the scores are well-calibrated. Thus, the score could be used as a proxy for the confidence of the model in its prediction. Other proxies may include the magnitude of the representation (Novoselov et al., 2023), or estimates of error from sampling subsets of windows from an episode. In Figure 7.3.2, we see that the neither the cosine similarity (cos) nor the magnitude sum (mag) are well-calibrated. Using a logistic regression-based recalibration method, we can get a well-calibrated score for the model.

Next, we test whether this calibration displays temporal degradation. From Figure 7.3.3 and Table 7.3.1, we see that the ECE is stable for a longer period but does degrade over a longer time span. These initial experiments demonstrate that recalibration can help produce a model with more temporally robust predictions. Further future work on recalibration may be able to achieve more robust predictions across different domains and demographics.

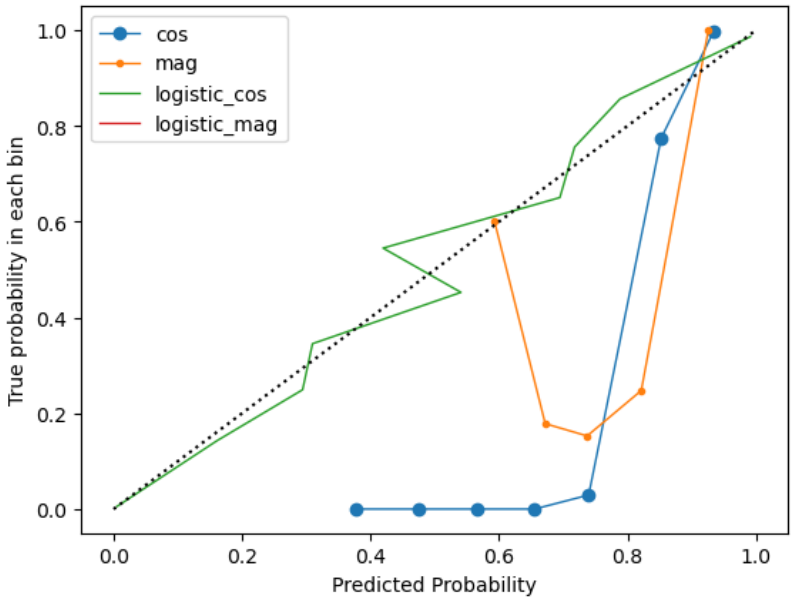


Figure 7.3.2: Reliability diagram for the LUAR model trained on the **TRVariable** dataset for 2015. The x-axis represents the predicted score, and the y-axis represents the empirical probability of the event. The dashed line represents perfect calibration.

ECE	year
0.003675	2015
0.003591	2016
0.003645	2017
0.005089	2018
0.007548	2019

Table 7.3.1: Measuring Temporal degradation of ECE for the recalibrated LUAR model across **TRVariable** splits.

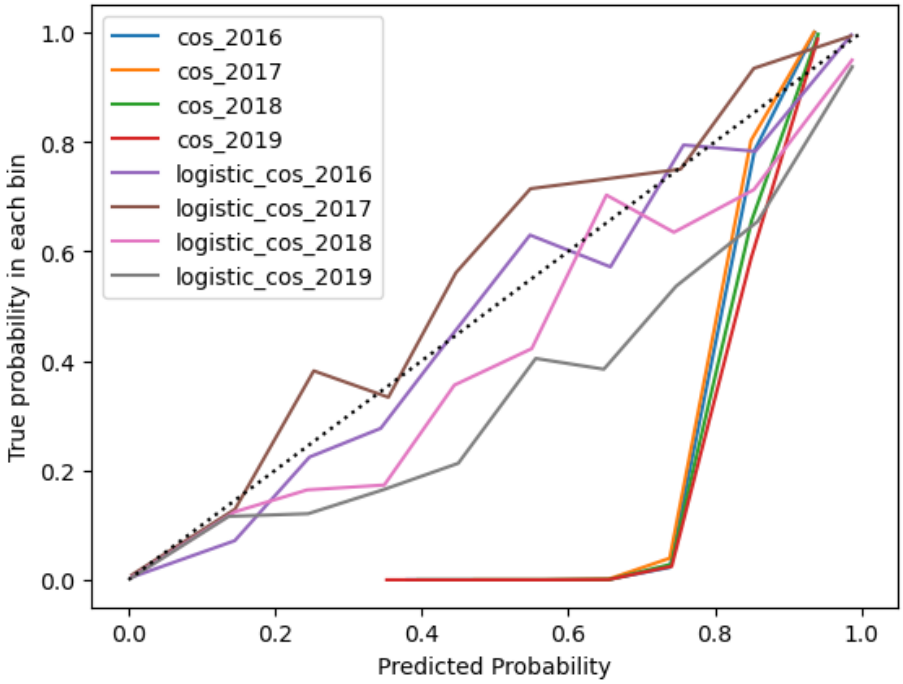


Figure 7.3.3: Plots for the reliability diagrams recalibrated LUAR model across **TRVariable** splits.

7.3.2 Conformal Prediction

In our work on extending runtime monitoring to graph structured data (Chapter 4), we discussed the approach of prediction sets generated through conformal prediction allowing control of the miscoverage rate. We speculate that this may be a candidate solution for providing guarantees on the performance of authorship attribution models. Notably, unlike the recalibration-based setup that models the task as a binary classification problem, conformal prediction could be a candidate for a more general framework for providing guarantees directly in the retrieval setting.

Specifically, a modified version of Adaptive Prediction Sets (Romano et al., 2020) could be used to add up the scores associated with the predictions until the coverage is achieved. Even if the threshold scores need to be recomputed, using formal guarantees provided by (Vovk et al., 2005), we can ensure that the coverage achieved is within a certain ϵ of the desired coverage. Further, with the framework of risk control (Angelopoulos et al., 2024), we can provide guarantees on the performance of the model for specific classes of risk functions. For the case of retrieval, since we are interested in measures such as recall and MRR. In future work, the risk control framework may be a candidate for achieving guaranteed performance for these measures.

Bibliography

Joshua Agterberg, Youngser Park, Jonathan Larson, Christopher White, Carey E Priebe, and Vince Lyzinski. 2020. Vertex nomination, consistent estimation, and adversarial modification. *Electronic Journal of Statistics*, 14(2):3230–3267.

Aws Albarghouthi, Loris D'Antoni, Samuel Drews, and Aditya V Nori. 2017. Fairsquare: probabilistic verification of program fairness. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA):1–30.

Aws Albarghouthi and Samuel Vinitsky. 2019. Fairness-aware programming. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 211–219.

Nicholas Andrews and Marcus Bishop. 2019. Learning invariant representations of social media users. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1684–1695, Hong Kong, China. Association for Computational Linguistics.

Anastasios N Angelopoulos and Stephen Bates. 2021. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*.

Anastasios N Angelopoulos, Stephen Bates, Emmanuel J Candès, Michael I Jordan, and Lihua Lei. 2021a. Learn then test: Calibrating predictive algorithms to achieve risk control. *arXiv preprint arXiv:2110.01052*.

Anastasios N Angelopoulos, Stephen Bates, et al. 2023. Conformal prediction: A gentle introduction. *Foundations and Trends® in Machine Learning*, 16(4):494–591.

Anastasios Nikolas Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. 2024. Conformal risk control. In *The Twelfth International Conference on Learning Representations*.

Anastasios Nikolas Angelopoulos, Stephen Bates, Michael Jordan, and Jitendra Malik. 2021b. Uncertainty sets for image classifiers using conformal prediction. In *International Conference on Learning Representations*.

Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine bias. In *Ethics of Data and Analytics*, pages 254–264. Auerbach Publications.

Abolfazl Asudeh, H. V. Jagadish, Julia Stoyanovich, and Gautam Das. 2019. Designing fair ranking schemes. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 1259–1276. ACM.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR*.

Rina Foygel Barber, Emmanuel J Candès, Aaditya Ramdas, and Ryan J Tibshirani. 2021. Predictive inference with the jackknife+. *The Annals of Statistics*, 49(1):486–507.

Rina Foygel Barber, Emmanuel J. Candès, Aaditya Ramdas, and Ryan J. Tibshirani. 2023. Conformal prediction beyond exchangeability. *The Annals of Statistics*, 51(2):816 – 845.

Georgios Barlas and Efstathios Stamatatos. 2020. Cross-domain authorship attribution using pre-trained language models. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 255–266. Springer.

Osbert Bastani, Xin Zhang, and Armando Solar-Lezama. 2019. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages*, 3(OOPSLA):1–27.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. In *Proceedings of the international AAAI conference on web and social media*, volume 14, pages 830–839.

R. K. E. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilović, S. Nagar, K. Natesan Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, K. R. Varshney, and Y. Zhang. 2019. Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development*, 63(4/5):4:1–4:15.

Tim Berners-Lee, Roy T. Fielding, and Larry M Masinter. 2005. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986.

Alex Biryukov, Ivan Pustogarov, Fabrice Thill, and Ralf-Philipp Weinmann. 2014. Content and popularity analysis of tor hidden services. In *2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, pages 188–193. IEEE.

Aaron Blum, Brad Wardman, Thamar Solorio, and Gary Warner. 2010. Lexical feature based phishing URL detection using online learning. *Proceedings of the Workshop on Artificial Intelligence and Security*, 1(1):1–37.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir P. Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher R'e, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the

opportunities and risks of foundation models. *ArXiv*.

Gwern Branwen, Nicolas Christin, David Décary-Hétu, Rasmus Munksgaard Andersen, StExo, El Presidente, Anonymous, Daryl Lau, Delyan Kratunov Sohhlz, Vince Cakic, Van Buskirk, Whom, Michael McKenna, and Sigi Goode. 2015. Dark net market archives, 2011-2015. `https://www.gwern.net/DNM-archives`.

Michael Brennan, Sadia Afroz, and Rachel Greenstadt. 2012. Adversarial stylometry: Circumventing authorship recognition to preserve privacy and anonymity. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):1–22.

Michael Robert Brennan and Rachel Greenstadt. 2009. Practical attacks against authorship recognition techniques. In *IAAI*.

Roderic Broadhurst, Matthew Ball, Chuxian Jiang, Joy Wang, and Harshit Trivedi. 2021. Impact of darknet market seizures on opioid availability. *Broadhurst R, Ball, M, Jiang, CX, et al*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. 2009. Building classifiers with independency constraints. In *2009 IEEE International Conference on Data Mining Workshops*, pages 13–18. IEEE.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Alessandro Castelnovo, Riccardo Crupi, Greta Greco, and Daniele Regoli. 2021. The zoo of fairness metrics in machine learning. *arXiv preprint arXiv:2106.00467*.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

Alexandra Chouldechova. 2017. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163.

Weibo Chu, Bin B Zhu, Feng Xue, Xiaohong Guan, and Zhongmin Cai. 2013. Protect sensitive sites from phishing attacks using features extractable from inaccessible phishing urls. In *2013 IEEE international conference on communications (ICC)*, pages 1990–1994. IEEE.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Jase Clarkson. 2023. Distribution free prediction sets for node classification. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 6268–6278. PMLR.

Wikimedia Commons. 2023. File:dread screenshot.png — wikimedia commons, the free media repository. [Online; accessed 7-March-2024].

US Congress. 2023. H.r.3369 - ai accountability act.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1107–1116, Valencia, Spain. Association for Computational Linguistics.

A Feder Cooper, Emanuel Moss, Benjamin Laufer, and Helen Nissenbaum. 2022. Accountability in an algorithmic society: relationality, responsibility, and robustness in machine learning. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 864–876.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 4690–4699. Computer Vision Foundation / IEEE.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2022. Turl: Table understanding through representation learning. *ACM SIGMOD Record*, 51(1):33–40.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

William Dieterich, Christina Mendoza, and Tim Brennan. 2016. Compas risk scales: Demonstrating accuracy equity and predictive parity. *Northpointe Inc*, 7(4).

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1723–1732, Beijing, China. Association for Computational Linguistics.

Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 135–144. ACM.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018a. Hotflip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36.

Mohammadreza Ebrahimi, Mihai Surdeanu, Sagar Samtani, and Hsinchun Chen. 2018b. Detecting cyber threats in non-english dark net markets: A cross-lingual transfer learning approach. In *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 85–90. IEEE.

The U.S. EEOC. 1979. Uniform guidelines on employee selection procedures.

Abeer ElBahrawy, Laura Alessandretti, Leonid Rusnac, Daniel Goldsmith, Alexander Teytelboym, and Andrea Baronchelli. 2019. Collective dynamics of dark web marketplaces. *ArXiv*

preprint, abs/1911.09536.

Yujie Fan, Yiming Zhang, Yanfang Ye, and Xin Li. 2018. Automatic opioid user detection from twitter: Transductive ensemble built on different meta-graph based similarities over heterogeneous information network. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 3357–3363. ijcai.org.

Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 259–268.

Avi Feller, Emma Pierson, Sam Corbett-Davies, and Sharad Goel. 2016. A computer program used for bail and sentencing decisions was labeled biased against blacks. it's actually not that clear. *The Washington Post*, 17.

Tao-Yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pages 1797–1806. ACM.

Gemma Galdon Clavell, Mariano Martín Zamorano, Carlos Castillo, Oliver Smith, and Aleksandar Matic. 2020. *Auditing Algorithms: On Lessons Learned and the Risks of Data Minimization*, pages 265–271. Association for Computing Machinery.

Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness testing: testing software for discrimination. In *Proceedings of the 2017 11th Joint meeting on foundations*

of software engineering, pages 498–510.

J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Sujata Garera, Niels Provos, Monica Chew, and Aviel D Rubin. 2007. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode*, pages 1–8.

Bishwamittra Ghosh, Debabrota Basu, and Kuldeep S Meel. 2021a. Justicia: a stochastic sat approach to formally verify fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7554–7563.

Soumya Ghosh, Q Vera Liao, Karthikeyan Natesan Ramamurthy, Jiri Navratil, Prasanna Sattigeri, Kush R Varshney, and Yunfeng Zhang. 2021b. Uncertainty quantification 360: A holistic toolkit for quantifying and communicating the uncertainty of ai. *arXiv preprint arXiv:2106.01410*.

Tony Ginart, Martin Jinye Zhang, and James Zou. 2022. Mldemon: Deployment monitoring for machine learning systems. In *International Conference on Artificial Intelligence and Statistics*, pages 3962–3997. PMLR.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

Google. 2007. Making the world's information safely accessible.

Kyle Gorman and Steven Bedrick. 2019. We need to talk about standard splits. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2786–2791, Florence, Italy. Association for Computational Linguistics.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. 2022. Why do tree-based models still outperform deep learning on tabular data? *arXiv preprint arXiv:2207.08815*.

Piotr Grzybowski, Ewa Juralewicz, and Maciej Piasecki. 2019. Sparse coding in authorship attribution for Polish tweets. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 409–417, Varna, Bulgaria. INCOMA Ltd.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Saket Gurukar, Priyesh Vijayan, Srinivasan Parthasarathy, Balaraman Ravindran, Aakash Srinivasan, Goonmeet Bajaj, Chen Cai, Moniba Keymanesh, Saravana Kumar, Pranav Maneriker, Anasua Mitra, and Vedang Patel. 2022. Benchmarking and analyzing unsupervised network representation learning and the illusion of progress. *Transactions on Machine Learning Research*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

William L Hamilton. 2020. *Graph representation learning*. Morgan & Claypool Publishers.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29.

William E Hart, Jean-Paul Watson, and David L Woodruff. 2011. Pyomo: modeling and solving mathematical programs in python. *Mathematical Programming Computation*, 3(3):219–260.

HelpNetSecurity. 2019. Phishing attacks at highest level in three years.

GE Hinton, JL McClelland, and DE Rumelhart. 1986. Distributed representations. in the pdp research group (eds.), parallel distributed processing: Explorations in the microstructure of cognition, volume 1. foundations (pp. 77-109).

Dennis Hirsch, Tim Bartley, Arvind Chandrasekaran, Srinivasan Parthasarathy, Piers Turner, Devon Norris, Keir Lamont, and Christina Drummond. 2020. Corporate data ethics: Data governance transformations for the age of advanced analytics and ai (final report). In *Appeared at the Privacy Law Scholars Conference (also available as SSRN Abstract: 3828239)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Shifu Hou, Yanfang Ye, Yangqiu Song, and Melih Abdulhayoglu. 2017. Hindroid: An intelligent android malware detection system based on structured heterogeneous information network. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 1507–1515. ACM.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Steven R Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. 2021. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080.

Zhiting Hu, Bowen Tan, Russ R Salakhutdinov, Tom M Mitchell, and Eric P Xing. 2019. Learning data manipulation for augmentation and weighting. In *Advances in Neural Information Processing Systems*, pages 15764–15775.

Kexin Huang, Ying Jin, Emmanuel Candes, and Jure Leskovec. 2023. Uncertainty quantification over graph with conformalized graph neural networks. *Advances in Neural Information Processing Systems*, 36:26699–26721.

Yongjie Huang, Jinghui Qin, and Wushao Wen. 2019. Phishing url detection via capsule-based neural network. In *2019 IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, pages 22–26. IEEE.

HuggingFace. 2019. Transformers - state-of-the-art natural language processing for pytorch and tensorflow 2.0.

Chip Huyen. 2022. *Designing Machine Learning Systems*. " O'Reilly Media, Inc.".

Federal Bureau of Investigation. 2019. 2019 internet crime report.

Sergey Ivanov and Evgeny Burnaev. 2018. Anonymous walk embeddings. In *International conference on machine learning*, pages 2186–2195. PMLR.

Disi Ji, Padhraic Smyth, and Mark Steyvers. 2020. Can i trust my fairness metric? assessing fairness with unlabeled data and bayesian inference. *Advances in Neural Information Processing Systems*, 33:18600–18612.

Justin Johnson, Agrim Gupta, and Li Fei-Fei. 2018. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228.

Armand Joulin, Édouard Grave, Piotr Bojanowski, and Tomáš Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

Patrick Juola. 2006. Authorship attribution. *Found. Trends Inf. Retr.*, 1(3):233–334.

B Justice Srikrishna. 2018. A free and fair digital economy: Protecting privacy, empowering indians.

Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. 2020. Learning and evaluating contextual embedding of source code. In *International Conference on Machine Learning*.

Moniba Keymanesh, Tanya Berger-Wolf, Micha Elsner, and Srinivasan Parthasarathy. 2021. Fairness-aware summarization for justified decision-making. *arXiv preprint arXiv:2107.06243*.

Aleem Khan, Elizabeth Fleming, Noah Schofield, Marcus Bishop, and Nicholas Andrews. 2021. A deep metric learning approach to account linking. In *Proceedings of the 2021 Conference*

of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5275–5287.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. 2017. Inherent Trade-Offs in the Fair Determination of Risk Scores. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 43:1–43:23, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457.

Ron Kohavi. 1996. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *KDD*, volume 96, pages 202–207.

Ramnath Kumar, Shweta Yadav, Raminta Daniulaityte, Francois R. Lamy, Krishnaprasad Thirunarayan, Usha Lokala, and Amit P. Sheth. 2020. edarkfind: Unsupervised multi-view learning for sybil account detection. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 1955–1965. ACM / IW3C2.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. *Advances in neural information processing systems*, 30.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*.

Hung Le, Quang Pham, Doyen Sahoo, and Steven CH Hoi. 2018. Urlnet: Learning a url representation with deep learning for malicious url detection. *arXiv preprint arXiv:1802.03162*.

Alex Leavitt. 2015. " this is a throwaway account" temporary technical identities and perceptions of anonymity in a massive online community. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 317–327.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7:195–212.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Pranav Maneriker, Codi Burley, and Srinivasan Parthasarathy. 2023a. Online fairness auditing through iterative refinement. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1665–1676.

Pranav Maneriker, Yuntian He, Scott Duxbury, Dana Haynie, and Srinivasan Parthasarathy. 2023b. Following the trail: Tracking user styles on clear and dark web forums. Conference Presentations.

Pranav Maneriker, Yuntian He, and Srinivasan Parthasarathy. 2021a. SYSML: StYlometry with Structure and Multitask Learning: Implications for Darknet forum migrant analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6844–6857, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pranav Maneriker, Jack W Stokes, Edir Garcia Lazo, Diana Carutasu, Farid Tajaddodianfar, and Arun Gururajan. 2021b. Urltran: Improving phishing url detection using transformers. In *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)*, pages 197–204. IEEE.

Henry B Mann and Donald R Whitney. 1947. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*,

pages 50–60.

Haspelmath Martin. 2017. The indeterminacy of word segmentation and the nature of morphology and syntax. *Folia linguistica*, 51(s1000):31–80.

James Martin. 2014. *Drugs on the dark net: How cryptomarkets are transforming the global trade in illicit drugs*. Springer.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Meta. 2016. Pytorch - from research to production.

Microsoft. 2023. Microsoft defender smartscreen.

Tomas Mikolov, Kai Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space. In *ICLR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 220–229.

E.F. Moore. 1956. Gedanken-experiments on sequential machines. *Automata Studies, Princeton University Press*, 129-153.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Rasmus Munksgaard and Jakob Demant. 2016. Mixing politics and crime–the prevalence and decline of political discourse on the cryptomarket. *International Journal of Drug Policy*, 35:77–83.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. 2020. A metric learning reality check.

Fatemeh Nargesian, Abolfazl Asudeh, and H. V. Jagadish. 2021. Tailoring data source distributions for fairness-aware data integration. *Proc. VLDB Endow.*, 14(11):2519–2532.

Jerzy Neyman and Egon Sharpe Pearson. 1933. Ix. on the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231(694-706):289–337.

Helen Nissenbaum. 1996. Accountability in a computerized society. *Science and engineering ethics*, 2(1):25–42.

Nima Noorshams, Saurabh Verma, and Aude Hofleitner. 2020. TIES: temporal interaction embeddings for enhancing social media integrity at facebook. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 3128–3135. ACM.

Sergey Novoselov, Galina Lavrentyeva, Vladimir Volokhov, Marina Volkova, Nikita Khmelev, and Artem Akulov. 2023. Investigation of different calibration methods for deep speaker embedding based verification systems. In *International Conference on Speech and Computer*, pages 159–168. Springer.

Central Digital Office and Data. 2021. Algorithmic transparency standard.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

European Parliament. 2018. 2018 reform of eu data protection rules. *European Commission*.

Sergio Pastrana, Daniel R Thomas, Alice Hutchings, and Richard Clayton. 2018. Crimebb: Enabling cybercrime research on underground forums at scale. In *Proceedings of the 2018 World Wide Web Conference*, pages 1845–1854.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Yongfang Peng, Shengwei Tian, Long Yu, Yalong Lv, and Ruijin Wang. 2019. A joint approach to detect malicious url based on attention mechanism. *International Journal of Computational Intelligence and Applications*, 18(03).

Geoffrey K Pullum and Barbara C Scholz. 2002. Empirical assessment of stimulus poverty arguments. *The linguistic review*, 19(1-2):9–50.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Francisco Rangel, Paolo Rosso, Moshe Koppel, Efstathios Stamatatos, and Giacomo Inches. 2013. Overview of the author profiling task at pan 2013. In *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, pages 352–365. CELCT.

Abhinav Rastogi, Raghav Gupta, and Dilek Hakkani-Tur. 2018. Multi-task learning for joint language understanding and dialogue state tracking. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 376–384, Melbourne, Australia. Association for Computational Linguistics.

Alec Ratdford, Karthik Narasimhan, Tim SSalimans, and Ilya Sutskeyver. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

F. Ren, Z. Jiang, and J. Liu. 2019. A bi-directional lstm model with attention for malicious url detection. In *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 300–305.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Rafael A. Rivera-Soto, Olivia Elizabeth Miano, Juanita Ordonez, Barry Y. Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. Learning universal authorship representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 913–919, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Yaniv Romano, Matteo Sesia, and Emmanuel Candes. 2020. Classification with valid and adaptive coverage. *Advances in Neural Information Processing Systems*, 33:3581–3591.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *ArXiv preprint*, abs/1706.05098.

Sebastian Ruder, Parsa Ghaffari, and John G Breslin. 2016. Character-level and multi-channel convolutional neural networks for large-scale authorship attribution. *ArXiv preprint*, abs/1609.06686.

David Reinsel-John Gantz-John Rydning, J Reinsel, and J Gantz. 2018. The digitization of the world from edge to core. *Framingham: International Data Corporation*, 16.

Mauricio Sadinle, Jing Lei, and Larry Wasserman. 2019. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114(525):223–234.

Doyen Sahoo, Chenghao Liu, and Steven C. H. Hoi. 2017. Malicious URL Detection using Machine Learning: A Survey. *arXiv preprint arXiv:1701.07179*, 1(1):1–37.

Arvind Satyanarayan, Ryan Russell, Jane Hoffswell, and Jeffrey Heer. 2015. Reactive vega: A streaming dataflow architecture for declarative interactive visualization. *IEEE transactions on visualization and computer graphics*, 22(1):659–668.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152.

Michael Lee Scott. 2000. *Programming language pragmatics*. Morgan Kaufmann.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1715–1725.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop, NeurIPS 2018*.

Prasha Shrestha, Sebastian Sierra, Fabio González, Manuel Montes, Paolo Rosso, and Thamar Solorio. 2017. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 669–674, Valencia, Spain. Association for Computational Linguistics.

Ravid Shwartz-Ziv and Amitai Armon. 2022. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90.

Leslie N Smith. 2017. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE.

Anders Søgaard, Sebastian Ebert, Jasmijn Bastings, and Katja Filippova. 2021. We need to talk about random splits. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1823–1832, Online. Association for Computational Linguistics.

Kacper Sokol and Peter Flach. 2020. Explainability fact sheets: a framework for systematic assessment of explainable approaches. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 56–67.

Rafael Alberto Rivera Soto, Kailin Koch, Aleem Khan, Barry Y. Chen, Marcus Bishop, and Nicholas Andrews. 2024. Few-shot detection of machine-generated text using style representations. In *The Twelfth International Conference on Learning Representations*.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556.

Chenkai Sun, Abolfazl Asudeh, H. V. Jagadish, Bill Howe, and Julia Stoyanovich. 2019. Mithralabel: Flexible dataset nutritional labels for responsible data science. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 2893–2896. ACM.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Xiao Hui Tai, Kyle Soska, and Nicolas Christin. 2019. Adversarial matching of dark net market vendor accounts. In *Proceedings of the 25th ACM SIGKDD International Conference on*

Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019, pages 1871–1880. ACM.

Farid Tajaddodianfar, Jack W. Stokes, and Arun Gururajan. 2020. Texception: A character/word-level deep learning model for phishing url detection. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2857–2861.

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Philip S Thomas, Bruno Castro da Silva, Andrew G Barto, Stephen Giguere, Yuriy Brun, and Emma Brunskill. 2019. Preventing undesirable behavior of intelligent machines. *Science*, 366(6468):999–1004.

Sebastian Thrun. 1998. Lifelong learning algorithms. *Learning to Learn*.

Anna Tigunova, Paramita Mirza, Andrew Yates, and Gerhard Weikum. 2020. Reddust: A large reusable dataset of reddit user traits. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6118–6126.

Florian Tramèr, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Jean-Pierre Hubaux, Mathias Humbert, Ari Juels, and Huang Lin. 2017. Fairtest: Discovering unwarranted associations in data-driven applications. In *2017 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 401–416.

Richard Vanderford. 2022. New york's landmark ai bias law prompts uncertainty. *WSJ*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *2018 ieee/acm international workshop on software fairness (fairware)*, pages 1–7. IEEE.

Vladimir Vovk. 2012. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490. PMLR.

Vladimir Vovk. 2015. Cross-conformal predictors. *Annals of Mathematics and Artificial Intelligence*, 74:9–28.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. 2005. *Algorithmic learning in a random world*, volume 29. Springer.

Andreas Wächter and Lorenz T Biegler. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57.

Andrew Wang, Cristina Aggazzotti, Rebecca Kotula, Rafael Rivera Soto, Marcus Bishop, and Nicholas Andrews. 2023. Can authorship representation learning capture stylistic features? *Transactions of the Association for Computational Linguistics*, 11:1416–1431.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. 2018. Cosface: Large margin cosine loss for deep face recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5265–5274. IEEE Computer Society.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. 2022. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019a. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.

Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R. Scott. 2019b. Multi-similarity loss with general pair weighting for deep metric learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 5022–5030. Computer Vision Foundation / IEEE.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. 2020b. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations*.

Anne L Washington. 2018. How to argue with an algorithm: Lessons from the compas-propublica debate. *Colo. Tech. LJ*, 17:131.

Ian Waudby-Smith, David Arbour, Ritwik Sinha, Edward H Kennedy, and Aaditya Ramdas. 2021. Time-uniform central limit theory, asymptotic confidence sequences, and anytime-valid causal inference. *arXiv preprint arXiv:2103.06476*.

Anna Wegmann and Dong Nguyen. 2021. Does it capture STEL? a modular, similarity-based linguistic style evaluation framework. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7109–7130, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Max Welling and Thomas N Kipf. 2016. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*.

Jonathan Woodbridge, Hyrum S Anderson, Anjum Ahuja, and Daniel Grant. 2018. Detecting homoglyph attacks with a siamese neural network. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 22–28. IEEE.

Lingfei Wu, Yu Chen, Kai Shen, Xiaojie Guo, Hanning Gao, Shucheng Li, Jian Pei, and Bo Long. 2021. Graph neural networks for natural language processing: A survey. *arXiv preprint arXiv:2106.06090*.

Lingfei Wu, Peng Cui, Jian Pei, Liang Zhao, and Le Song. 2022. Graph neural networks. *Graph Neural Networks: Foundations, Frontiers, and Applications*, pages 27–37.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang,

Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, abs/1609.08144.

Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. 2020. Inductive representation learning on temporal graphs. *ArXiv*, abs/2002.07962.

Tom Yan and Chicheng Zhang. 2022. Active fairness auditing. In *International Conference on Machine Learning*, pages 24929–24962. PMLR.

Ke Yang, Julia Stoyanovich, Abolfazl Asudeh, Bill Howe, H. V. Jagadish, and Gerome Miklau. 2018. A nutritional label for rankings. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1773–1776. ACM.

Suleiman Y Yerima and Mohammed K Alzaylaee. 2020. High accuracy phishing detection based on convolutional neural networks. In *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)*, pages 1–6. IEEE.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. 2017. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970. PMLR.

Soroush Zargarbashi, Simone Antonelli, and Aleksandar Bojchevski. 2023. Conformal prediction sets for graph neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 12292–12318. PMLR.

Andrew Zhai and Hao-Yu Wu. 2019. Classification is a strong baseline for deep metric learning. In *30th British Machine Vision Conference 2019, BMVC 2019, Cardiff, UK, September 9-12, 2019*, page 91. BMVA Press.

Jie M. Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, pages 1–1.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yiming Zhang, Yujie Fan, Wei Song, Shifu Hou, Yanfang Ye, Xin Li, Liang Zhao, Chuan Shi, Jiabin Wang, and Qi Xiong. 2019a. Your style your identity: Leveraging writing and photography styles for drug trafficker identification in darknet markets over attributed heterogeneous information network. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 3448–3454. ACM.

Zhuosheng Zhang, Hai Zhao, Kangwei Ling, Jiangtong Li, Zuchao Li, Shexia He, and Guohong Fu. 2019b. Effective subword segmentation for text comprehension. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(11):1664–1674.

Shengjia Zhao, Enze Zhou, Ashish Sabharwal, and Stefano Ermon. 2016. Adaptive concentration inequalities for sequential decision problems. *Advances in Neural Information Processing Systems*, 29.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76.

Zvelo. 2020. The rise of single-use phishing urls and the need for zero-second detection.