The Role of Structure in Building Adaptive Machine Learning

Candidacy proposal

By

Pranav Maneriker,

Graduate Program in Department of Computer Science and Engineering

The Ohio State University

2022

Candidacy Proposal Committee:

Dr. Srinivasan Parthasarathy, Advisor

Dr. Andrew Perrault

Dr. Micha Elsner

Dr. Eric Fosler Lussier

Abstract

The success of neural networks and the advent of specialized hardware such as GPUs has led to the growth of larger models with increasingly large unstructured datasets in machine learning. Curating and assembling a high-quality large dataset is a time-consuming and expensive 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 continue to drift and change over time models need to be periodically retrained to keep up. Graph structures, both implicit and explicit, are ubiquitous in Natural Language Processing. Implicit structures include those derived from language morphology, syntax, and semantics and are 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 proposal, we posit that an abundance of associated structural information is underutilized 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 directions in which to use these structures, viz. augmenting existing text models with structure, extracting structured representations from unstructured text, and structured-enhanced monitoring of the performance of models 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 a state-of-the-art system for phishing URL detection. In this work, we comprehensively analyze 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. We then demonstrate how these models can be more robust by using adversaries constructed from benign URLs using a domain-informed attack scenario. In both fine-tuning and adversarial attacks, the underlying syntax of URLs serves as the structure that enables us to build a robust model.

The second direction of work we study is the role of intrinsic structure in the visualization and analysis of the performance 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 towards meaningful and compliant fairness specifications.

In our third contribution, we study the task of author identification on darkweb forums. 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 AuthorID across time-disjoint task settings. 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.

We hope to explore and expand our existing work in these directions in future work. First, we aim to explore two aspects of augmentation in the stylometry setting - data scale and model explainability. The real-world application for stylometry pertains to content moderation. In these scenarios, models must be able to adapt to large-scale textual corpora. In addition, a human, when making a moderation decision (say,

banning a user), must provide a justification. Existing stylometry explanation techniques rely on general text explainability approaches. We aim to expand on our previous explorations to provide bespoke techniques beyond those that use gradient-based attribution approaches.

Table of Contents

List of Tables

List of Figures

Todo list

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 large 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 the use of 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 still continue to outperform deep learning on a number of tasks including classification and regression (Shwartz-Ziv and Armon, 2022; Grinsztajn et al., 2022). The successes include tasks such as 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 has 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 thesis proposal is to demonstrate that structures, either those present implicitly or derived explicitly, provide opportunities for building improved models over the 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 study of this proposal is the utility of these structures for both formal and natural languages. A formal language refers to a set of strings of symbols that are 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 grammars and their 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 play a direct role in 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, there exist multiple competing formulations 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, either as trees having recursive phrase-structures (constituency grammars) or as general graphs connecting dependent words (dependency grammars). By *explicit* structure, we refer to the locally ascribed or inferred semantics, as well as contextual structure that help 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 choices for structures, 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 proposal, we aim to demonstrate scenarios where it is necessary to take advantage of structure to achieve *adaptive* machine learning systems.

what makes style different from other issues in the llm continuum

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 on another domain. We examine these issues in view of 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 a corresponding challenge with building *adaptable* ML. These are *adversarial testing*, *online monitoring*, and *domain generalization*.

Cite the twitter thread correctly, arguing with the dilemma faced by modern NLP practitioners

Adversarial Testing The standard paradigm of for evaluation of an ML model is using train-validation-test splits to estimate the performance of a model. Common 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 is only an estimate of true performance. Some of these issues can be avoided by 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.

Talk about focusing on temporal splits.

Potentially mention synthetic data?

Online Monitoring Once a model has been deployed and begins to influence decisions being made in the real-world, it is important to monitor it to understand whether it achieve the expected performance. The distribution of data encountered by the model at training time may be different from when it is actually 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 an imperative for auditing the compliance of decision-making ML systems with regulatory requirements. Creating efficient monitoring tools that can establish when a deployed model cannot reach regulatory or performance targets is a difficult task.

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 that has been built for a specific domain may be adapted for use in other, analogous settings, each having their own

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 that we study in this proposal is the setting up of model architectures, training algorithms, and understanding the impact of domain generalization in specific contexts.

1.1 Proposal Statement

In this proposal, we aim to establish structure-based mechanisms usable throughout the stages of development and deployment 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 the design of more adaptable machine learning systems. We focus our work on three specific challenges in this space, adversarial testing, online monitoring, and. domain generalization. Our contributions describe approaches that utilize core ideas to address each of these challenges. 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? Following the deployment of a model, can we quantify whether a model achieve or violates a fairness guarantee while minimizing the number of samples required? Are there procedures that can be used to improve a model built for a specific domain such that it can generalize to a new domain? Can we use information from multiple, related domains to further enhance a model built for a specific domain?* 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

However, as the number of URLs and known phishing pages have 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 model 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 towards meaningful and compliant fairness specifications. This visualization also makes use of the parse trees for fairness metrics that are generated from the grammar used to define them.

1.2.3 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 shot texts on online social media users (Shrestha et al., 2017; Andrews and Bishop, 2019). ur 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 its efficacy.

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 aim to study how different domains and their associated structure correlate with 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. In the context of our aforementioned contribution on stylometry using structure on the darkweb, two questions arise. First, the number of users and posts are limited in the darkweb

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

domain. For example, the total number of forum posts across multiple years on large darkweb forums only number among 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 question - can structure-based methods improve authorship attribution models when the availability of text/authors and size of models is scaled up by multiple orders of magnitude? The second question that we aim to explore is whether these structures are domain-specific, or whether certain meta-structures common across domains can help generalize these to multiple domains. We aim for our analysis to supplement the findings around models that only use text (Barlas and Stamatatos, 2020; Rivera-Soto et al., 2021), potentially providing alternative mechanisms for successful transfer.

1.3.2 Interpretable Stylometric Modeling

Recent work has demonstrated that representation learning models (Andrews and Bishop, 2019; Maneriker et al., 2021a; Rivera-Soto et al., 2021) for authorship attribution can outperform more traditional models that utilize n-gram style features such as TF-IDF and SCAP (Frantzeskou et al., 2007). Despite the improvements offered by these models, a large number of implemented systems continue to use unigram features for authorship attribution and related tasks, eg. state of the art ban evasion systems (Niverthi et al., 2022). A recent preprint (Tyo et al., 2022) questions the magnitude of the claimed gains from these representation learning techniques. Thus, to be able to use these models in practice, it is essential to explain the mechanisms by which authorship attribution models make decisions. Further, prior to the deployment of any representation learning models for content moderation, better explainability is also vital for helping decision makers provide their motivation when enacting bans Finally, the use of authorship attribution can impact the anonymity of individuals. Better explainability can help by providing prescriptive mechanisms for improving the privacy of concerned users. This is an active area of research with interest from organizations such as IARPA (2022). This problem is amplified when considering structure-based models as there are two different modalities that may influence decision making. In the second direction of future work, we hope to make inroads into addressing these challenges using structured *adversarial testing*.

Does this statement add anything useful?

1.3.3 Better Fairness Auditing in Non-IID Settings

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 tracked fairness metric which allows monitoring with arbitrary stopping mechanisms. However, using this inequality comes with the certain assumptions of the data generating process. The most restrictive of these are the assumption 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. For example, the underlying data is unlikely to be from a stationary distribution. Drifts in the distributions are usually categorized based on the causal mechanisms that underlie them: those that arise from changes in the distribution of input features (covariate shift); the support distribution of the true labels (label shift); or changing relationships between the input and output (concept drift) (Huyen, 2022). Additionally, the metric that a user may want to monitor may change over time due to evolving regulatory environments. Finally, the data may be correlated with each other through network structures. We aim to address how recent work

frames some of these challenges and motivate extensions to our system AVOIR(Chapter 3) to address some of these challenges.

reread this once

1.4 Organization

The remainder of the proposal 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 *run time* fairness properties associated with *deployed* machine learning models in Chapter 2. In Chapter 4, we describe how an authorship identification model trained on one domain can be *generalized* to work across domains. Finally, in Chapter 5 we describe the the directions that we aim to focus on in future work. This includes scaling authorship identification models to work across even more domains while utilizing the graphs associated with those domains and improving the interpretability of the associated models. Finally, we describe how fairness monitoring can be made more interactive and work across settings beyond those explored in Chapter 3.

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 tend to 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 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. Recent research has proposed the use of 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 perform a comprehensive analysis of transformer models on the phishing URL detection task. We consider both pre-trained models and end-to-end trained transformer models, with both standard masked language model 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 positives 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).

Fix reference

As mentioned previously, phishing URL are carried out through short-lived and changing URL patterns. Therefore, the machine learning models must be retrained and re-deployed 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 in this work, we propose a threat model and construct an *adversarial testing* scenario to validate models against known threat patterns prior to 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

Background on adversarial testing at some point?

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) and Microsoft's Edge browser includes Windows Defender SmartScreen (Microsoft). 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., 2020). 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.

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

[2]Convolutional Neural Networks

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). 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`).

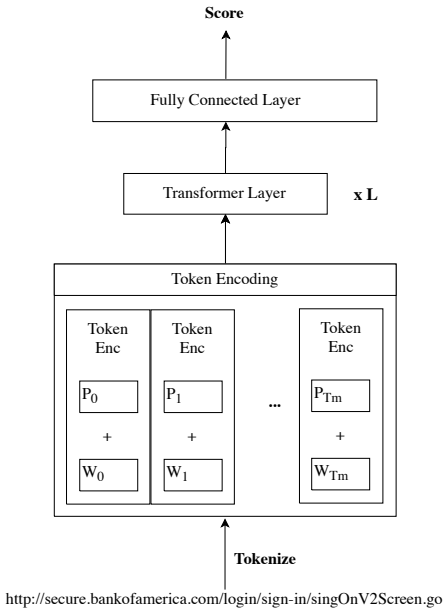


Figure 2.1: URLTran phishing URL detection model.

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

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.

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; 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

Potentially discuss co-variate shift here?

Consider rewording this seciton

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' $\rightarrow$ 'bank-of-america.com'. To implement this, we leveraged a

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

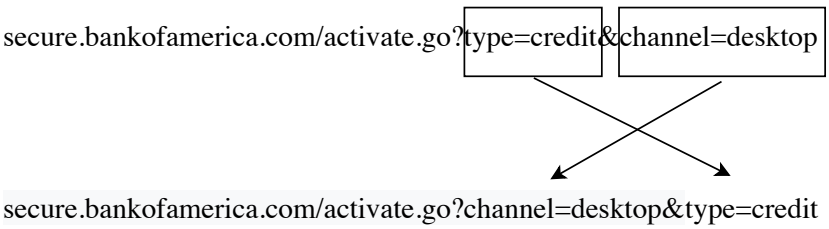


Figure 2.2: An example of parameter reordering

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 $d[i \dots j]$ 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.

Maybe add the dynamic program algorithm here, and talk about the computational complexity?

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

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

$\{3, 6, 12\}$, 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) 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; 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

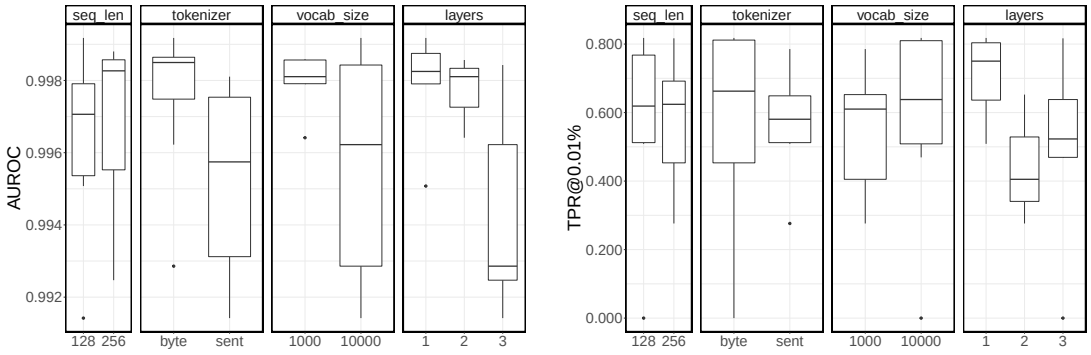
Transformers typically require large amounts of pre-training data (e.g., BERT (Devlin et al., 2019) used a corpus of $\approx$ 3.3 B tokens). However, this data is derived from text articles, which are structured differently from URLs. We trained the URLTran$_C$ model based solely on the URL data found 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.

Some figure font manipulation may be needed

pre-training vs pretraining consistent

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* 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.,



(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

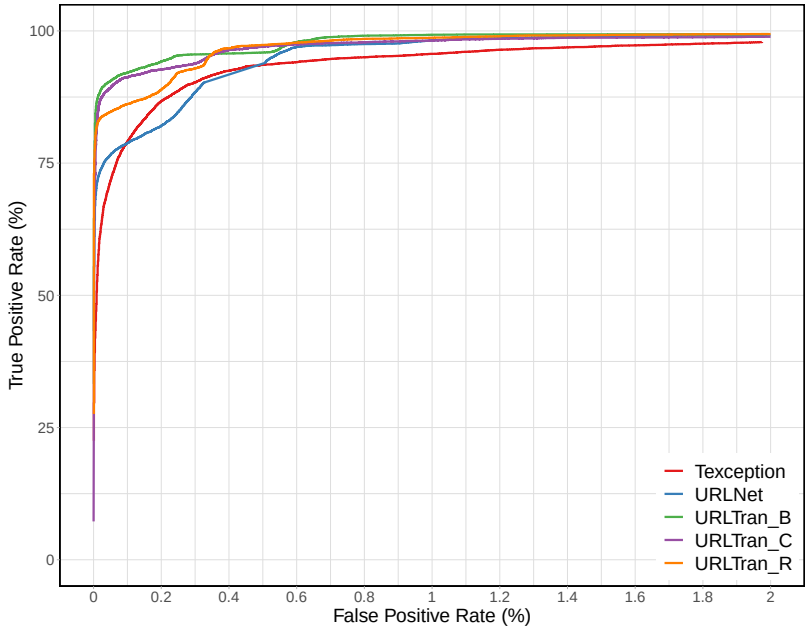


Figure 2.4: Receiver operating characteristic curve indicating the performance of the URLTran and several baseline models zoomed into a maximum of 2% false positive rate.

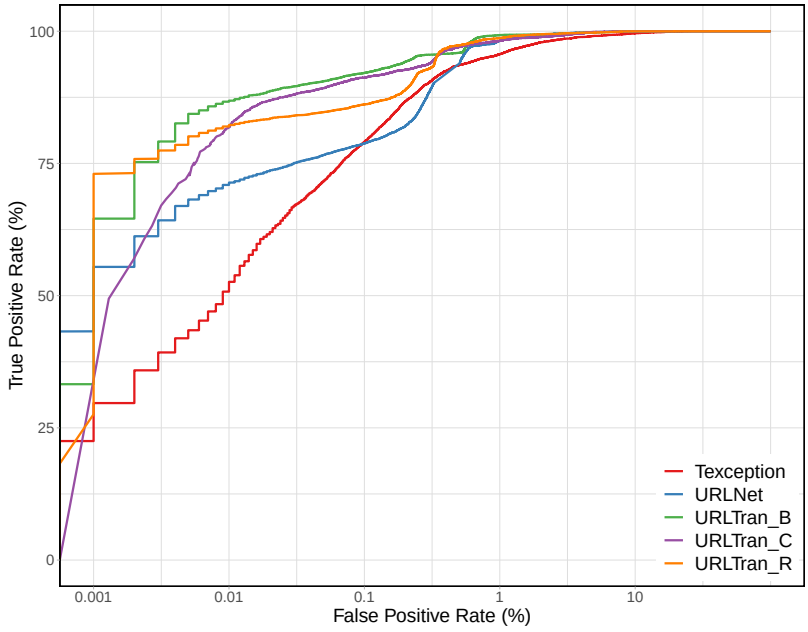


Figure 2.5: Zoomed in receiver operating characteristic curve with a log x-axis.

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.

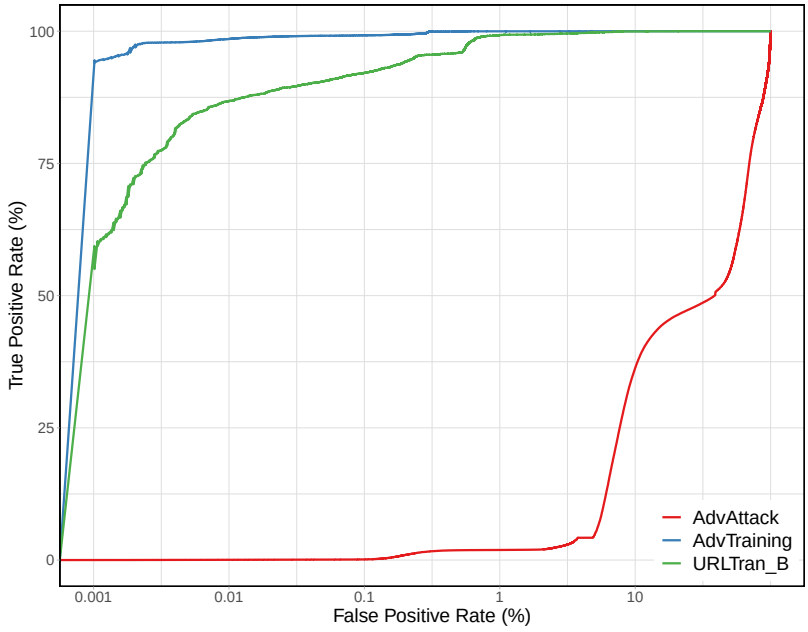


Figure 2.6: ROC curve for URLTran$_B$ when under adversarial attack, and adversarial robustness after augmented training

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 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.

Another place where covariate shift can be referenced.

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.

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.

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.

	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.

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 towards 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 ameliorate the issues with faulty fairness metric design.

3.1 Introduction

The use of advanced analytics and artificial intelligence (AI), along with its many benefits, poses important threats to individuals and broader society at large. (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 produce several sources of risk at each step. To mitigate such risks, organizations need to be able to audit the algorithms to ensure that they are consistent with stated data ethics policies and regulatory requirements.

Governments across the world are wrestling with the implementation of auditing regulation and practices for increasing the accountability of decision processes. Recent examples include the New York City auditing requirements for AI hiring tools (Vanderford, 2022), European data regulation (GDPR 2018), accountability bills 2019; 2021 and judicial reports 2018. In an evolving environment, organizations also aim to augment their understanding of algorithmic audits. 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 audits the behavior of AI algorithms. Such ideas date back to the 1950s (Moore, 1956) but research has largely been sporadic until very recently with the widespread use of AI-based decision making giving rise to the vision of algorithmic auditing (**?**).

Machine learning testing (Zhang et al., 2020) is an avenue that can be used to expose undesired behavior and improve the trustworthiness of machine learning systems. Compared with traditional testing systems, machine learning testing faces fundamental differences. First, the behavior of machine learning systems may change based on the domain and distribution shift of the input training data. Second, machine learning is trained with the intention of generalizing to unobserved data, which is often at odds with general-purpose testing (Zhang et al., 2020). Several works aim to verify the fairness of machine learning systems. Given a specific definition of fairness, Fairtest (Tramèr et al., 2017) and Verifair (Bastani et al., 2019) build a comprehensive framework for investigating fairness in data-driven pipelines. Fairness Testing provides a notion of causal fairness and generates tests to check the fairness of a given decision-making procedure (Galhotra et al., 2017). Fairness-aware programming 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.

Alternative frameworks such as the AI Fairness 360 (Bellamy et al., 2019) provide mechanisms to quantify fairness uncertainty for pre-supported metrics. Uncertainty quantification (Ghosh et al., 2021b; Ginart et al., 2022) provides adaptive guarantees, however, existing work is designed for commonly used outcome metrics such as accuracy, F1-score, etc., rather than for fairness metrics. Justicia (Ghosh et al., 2021a) optimizes uncertainity for fairness metrics estimates using stochaistic SAT solvers but can only be applied to a limited class of tree-based classification algorithms. .

Add references to ICML 2022 paper

We present a framework for *Auditing and Verifying fairness Online through Interactive Refinement* (AVOIR) [6]. AVOIR builds upon the ideas on distributional probabilistic fairness guarantees (Albarghouthi and Vinitsky, 2019; Bastani et al., 2019), generalizing them to real-world data. Prior work has mainly focused on data generated from known distributions. The extensions that we propose in AVOIR allow for the generation of probabilistic guarantees for arbitrary, possibly unknown, decision functions. Further, AVOIR leverages a novel tree-based optimization and visualization that enables a user to choose an a suitable fairness specification that can be guaranteed to make more efficient estimates than prior work. This visualization and optimization is derived using the intrinsic structure of the fairness specification grammar supported by AVOIR. Figure 3.1 shows a summary of the framework, and Table 3.1 provides a summary of the comparison of AVOIR to related prior work.

3.2 Overview

3.2.1 Fairness Criteria

Decision making/scoring functions aim to optimize for a specific metric such as accuracy, precision, F1 score etc. Fairness criteria, on the other hand, quantify the relationship between the outcome metric across multiple subgroups or similar individuals among the population. Formal definitions of fairness focus on observational criteria, i.e., those that can be written down as a

[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.

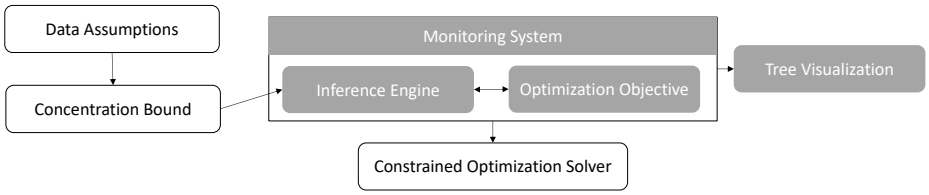


Figure 3.1: Shaded nodes describe our contributions to create the AVOIR framework.

Tool	Flexible Fairness Metrics	Adaptive Optimization	Online Monitoring	Visual Refinement	Black-Box Model
(Albarghouthi and Vinitsky, 2019)	✓	✗	✓	✗	✓
(Bellamy et al., 2019)	✓	✗	✗	✗	✓
(Bastani et al., 2019)	✓	✓	✓	✗	✓
(Ghosh et al., 2021b)	✗	✗	✓	✗	✓
(Ginart et al., 2022)	✗	✓	✓	✗	✓
(Ghosh et al., 2021a)	✓	✓	✗	✗	✗
AVOIR (Our work)	✓	✓	✓	✓	✓

Table 3.1: Comparison of AVOIR with prior work.

probability statement involving the joint distribution of the features, sensitive attributes, decision making function, and actual outcome. For example, a decision making function that selects amongst applicants to hire for a specific position may have different hiring rates for majority and minority population groups. Suppose r denotes the return value of the decision function, and s is an indicator denoting whether a candidate belongs to a minority population. An example fairness criterion is the 80%-rule (Zafar et al., 2017) states that

$$\min\left\{\frac{\Pr[r|s]}{\Pr[r|\neg s]}, \frac{\Pr[r|\neg s]}{\Pr[r|s]}\right\} \geq 0.8$$

In our work, we focus on fairness criteria that can be expressed using Bernoulli random variables. This covers a majority of popular fairness criteria (Verma and Rubin, 2018). Section 3.4 contains descriptive analyses and case studies of real world applications of AVOIR on commonly used fairness criteria.

3.2.2 Fairness Specifications

We aim to derive statistical guarantees about fairness criteria based on observed outputs. These guarantees are derived in the form of assertions over specifications. 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. Similar assertions can be made about observed and tracked variables. Specifically, let X be an observed Bernoulli r.v[7]

$$\phi_X := X : (\overline{\mathbb{E}}[X], \epsilon, \delta) \equiv \Pr[|\mathbb{E}[X] - \overline{\mathbb{E}}[X]| \geq \epsilon] \leq \delta \tag{3.1}$$

[7]random variable

where $\overline{\mathbb{E}}[X]$ denotes an empirical estimate of $E[X]$. Given a stream of (observations, outcomes from the decision functions), and a specified threshold probability δ, AVOIR will continue to refine the estimate for a given specification until reaching the failure threshold.

3.2.3 Contributions

3.2.3.1 Probabilistic Guarantees: Optimization for Concentration Bounds

When monitoring the value of a fairness metric in real-world scenarios, a system makes sequential decisions based on observed outcomes. Previous work on verifying fairness improve bounds using better concentration from union bounding Hoeffding inequality (Albarghouthi et al., 2017; Albarghouthi and Vinitsky, 2019) to the Adaptive Hoeffding inequality (Zhao et al., 2016) used by (Bastani et al., 2019)) to reduce the number of samples required to generate low failure probability estimates for a specification. The overall failure probability of an assertion is computed as the sum of failure probabilities of each constituting sub-expression (using the union bound). Proving guarantees for overall uncertainty across multiple groups involves balancing it across groups with differences in the number of observed samples. We derive a method for computing improved probabilistic bounds on point estimates of a model's adherence to a specification, in *provably fewer* iterations in the *online setting*. Prior work does not adequately optimize the union-bounding procedure, leading to weaker bounds. Specifically, they allocate equal failure likelihoods among all sub-expressions of a given expression. This is inefficient as fairness criteria typically involve estimating sub-expressions for terms having different numbers of observations (minority and majority groups). Such an estimator can tolerate a higher failure probability with lower change in the confidence interval. For example, consider Bernoulli r.vs $X_{1,2}$ for which we derive concentration guarantees $\Pr[|\mathbb{E}[X_i] - \overline{\mathbb{E}}[X_i]| \geq \epsilon_i] \leq \delta_i$ after t_i observations. From the Hoeffding inequality, $\delta = 2e^{-2t\epsilon^2}$. We can claim stronger guarantees for X_2 if $t_2 > t_1$ as the failure probability is lower at the same concentration. This observation enables us optimize over sub-expressions to provide stronger overall concentration for compound expressions. Adaptive versions of these inequalities also have similar patterns (see Figure 3.2a). Section 3.3 provides further details on how this allocation can be optimized.

3.2.3.2 Inference Engine for Arbitrary Metrics

One mechanism to support monitoring is to construct a custom domain specific language (DSL, see Figure 3.2b) parser and perform edit operations on the extracted syntax tree. Our implementation takes a different route - we implemented the entire DSL using python. This provides an easy mechanism to integrate AVOIR into existing python ML pipelines. We build our framework as a library for specifying fairness criteria as decorators over python functions. That is, each non-terminal expressible in the grammar corresponds to a unique python object. We construct classes to represent each of these objects and overload the appropriate operators such that any statement in the python version of the DSL corresponds to a unique statement in the modified grammar. The classes themselves contain code to carry out the inference necessary to compute the failure probabilities and optimization setup required for any fairness metric implemented in the grammar. Assuming that some concentration bound on the sum/mean of the decision functions is available, AVOIR supports tracking for any group fairness metric. We demonstrate AVOIR using the Adaptive Hoeffding bound (Zhao et al., 2016) which applies to sub-gaussian random variables. Further, we will open-source our implementation. Prior work either does not have open-source implementations available (Albarghouthi and Vinitsky, 2019) or the available implementations only provide examples

for specific metrics (Bastani et al., 2019) and require tedious manual optimization and implementation efforts to generalize.

3.2.3.3 Visualization as a Tool for interactive refinement

There are a plethora of fairness criteria and subtle changes in their definition can change the implications on decision making (Castelnovo et al., 2021). Thus, practitioners need support when selecting, designing, and guaranteeing fairness for deployed machine learning algorithms. We develop an application for visually analyzing adherence to a specification for any data/model. The application uses a novel visualization to allow users to understand how specification violations arise by using the context from surrounding violations.

3.3 AVOIR Framework

3.3.1 Language Specification

We describe AVOIR's Domain Specific Language (DSL) used for specifying fairness metrics. Concrete examples of implemented specifications in AVOIR's DSL are provided in Section 3.4. We focus on binary decision making functions; their outputs can be characterized by Bernoulli r.v.s. Note that for such a Bernoulli r.v. X, $\mathbb{E}[X] = \Pr[X = 1]$ and hereafter, these are used interchangeably. We start with the grammar used by prior work and enhance the grammar to simplify the expressions used for common fairness specifications. Figure 3.2b describes the full grammar. We modified the grammar to include two additional operations. First, we added a `given` argument to the expectation term, 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 used for group fairness specification. Additionally, we add binary comparison operators $<, >, ==, ! =$, which further simplifies the process of writing specifications.

3.3.2 Propagating Bounds

Generating the bounds for a specification requires propagating guarantees from elementary subexpressions. Assuming that observed values for each `ETerm` correspond to an underlying random variable X, A probabilistic guarantee ϕ_X consists of an empirical estimate $\overline{\mathbb{E}}[X]$, a concentration bound ϵ_X, and a failure probability δ_X, such that $\Pr[|\mathbb{E}[X] - \overline{\mathbb{E}}[X]| \geq \epsilon_X] \leq \delta_X$. We refer to expressions of this form as *elementary* subexpressions. A fairness specification will typically consist of multiple such elementary expressions, denoted as *compound* expressions. For compound expressions, we must infer the implied guarantees that can be provided, with corresponding constraints. Each inference rule corresponds to a derivation in the DSL grammar. Inference rules have preconditions and postconditions that follow the general expression

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

where ϕ denotes a claim for a subexpression, ψ for a <spec>, $\overline{\mathbb{E}}$ and ϵ are the mean and concentration terms associated with a subexpression claim, C denotes a constraint. For example, consider starting with $X : (\overline{\mathbb{E}}[X], \epsilon_X, \delta_X)$, $Y : (\overline{\mathbb{E}}[Y], \epsilon_Y, \delta_Y)$

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

i.e., $X \pm Y : \big(\overline{\mathbb{E}}[X] \pm \overline{\mathbb{E}}[Y], \epsilon_X + \epsilon_Y, \delta_X + \delta_Y\big)$. Some derivations also lead to rules that require constraints. For instance, assume $X : (\overline{\mathbb{E}}[X], \epsilon_X, \delta_X), \overline{\mathbb{E}}[X] > c$. Then we have $\Pr[X < \overline{\mathbb{E}}[X] - \epsilon_X] > 1 - \delta$ If we add the constraint that $\overline{\mathbb{E}}[X] - \epsilon_X \geq c$, we have $\Pr[X < c] > 1 - \delta$, thus,

$$\begin{aligned}X : (\overline{\mathbb{E}}[X], \epsilon_X, \delta_X) \implies \psi \equiv X > c : (T, \delta_X) \\ \text{under the constraint } \{\overline{\mathbb{E}}[X] - \epsilon_X \geq c\}\end{aligned}$$

The full set of inference rules required for the DSL is provided in the appendix (Figure 3.5). The implementation in AVOIR follows these rules but can be extended to other rule inference templates that support the DSL. We note that these rules extend the ones implemented by VeriFair (VF)[8] (Bastani et al., 2019) with constraints that enable the optimizations required in AVOIR (see Appendix 3.8).

3.3.3 Optimizing Bounds

3.3.3.1 AVOIR Algorithm

The pseudocode for the optimization procedure in AVOIR is described in the appendix (Algorithm 1). The input to the algorithm is the reporting threshold probability Δ and a specification ψ. We then infer a symbolic optimization problem is inferred corresponding to the failure probabilities and constraints derived from concentration bounds. At each step, the `OBSERVE(X)` function is called with new observation of every *elementary* subexpression and observed output. The running mean and counts of observations are updated. The final optimization problem `OPT` corresponding to each specification is a nonlinear constrained optimization problem. We use the COIN-OR implementation of IPOPT (Wächter and Biegler, 2006), accessed though the Pyomo (Hart et al., 2011) interface to solve this problem at each step. If a solution is successfully found for `OPT`, the algorithm terminates, with the estimate for the specification having reached the required threshold. If no solution is found, the estimates continue to be updated with $\delta_i = \Delta$ for each *elementary* subexpression. The main intuition behind the algorithm is to create a confidence sequence corresponding to the estimates at each time step. The `OPT` corresponding to a specification:

$$\begin{aligned}&\min_{\delta_i} \sum_{i=1}^{n} \delta_i \\ &\text{s.t. } g_k(\delta_{1,\ldots,n}, \overline{\mathbb{E}}[X_1], \ldots, \overline{\mathbb{E}}[X_n]) \leq \epsilon_k \\ &0 \leq \delta_i \leq 1\end{aligned} \tag{3.2}$$

where g_k and ϵ_k are the functions/bounds derived using the transformations carried out through the DSL inference rules (further details in Appendix 3.8.2).

[8] Verifair

Definition 1. *For $\delta \in (0,1)$, a $(1-\delta)$ confidence sequence is a sequence of confidence sets, usually intervals $(\mathrm{CI_t})_{t=1}^{\infty}$, say $\mathrm{CI_t} := (\mathrm{L_t}, \mathrm{R_t}) \subseteq \mathbb{R}$ satisfying a uniform convergence guarantee. After observing the tth unit, we calculate an updated confidence set $\mathrm{CI_t}$ for an unknown quantity of interest 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 a r.v. $X \in \{0,1\}$ such that

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

can be used in AVOIR. Here, $\overline{\mathbb{E}}_t[X]$ denotes the empirical estimate of $\mathbb{E}[X]$ after the t^{th} observation. For the purpose of comparison with previous work (eg., VF), we use the Adaptive Hoeffding Inequality (Zhao et al., 2016), which will be referred to as AIN hereafter.

Theorem 1. *The sequence of estimates generated by AVOIR form a confidence set.*

The proof follows from the fact that AVOIR always estimates using a failure probability higher than that which is provided by AIN, and hence applying a union bound ensures that the estimates are a confidence set. The full proof is provided in Appendix 3.10.

Corollary 1.1. *The estimates for the overall specification ψ form a confidence sequence converging to $\psi : (b, \Delta), b \in \{T, F\}$.*

Proof. We initialize the main specification with the required failure probability Δ. The termination condition requires $\sum \delta_i \leq \Delta$. From Theorem 1 we can infer that the confidence sequence corresponding to the termination achieves the required threshold Δ, and therefore, is valid. □

3.3.3.2 Improvements over Baseline

3.3.3.2.1 Concrete Example Consider a Bernoulli r.v R 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 positive decision for the majority and minority classes, respectively. Suppose we aim to estimate $\psi := X - Y < \epsilon_T$

We demonstrate the improvements possible using our approach by instantiating this example with data. Suppose we want the upper bound of the failure probability $\Delta = 0.1$ for the specification. Consider a set of observations such that $\overline{\mathbb{E}}[X] = 0.8, n_X = 1550$ and $\overline{\mathbb{E}}[Y] = 0.5, n_Y = 310$. Figure 3.3a 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 from section 3.2.3.1 about allocating higher failure probability to terms with the majority of observations. The optimization problem inferred by AVOIR:

$$\begin{aligned} &\min_{\delta_X, \delta_Y} \ \delta_X + \delta_Y \\ &\text{s.t. } \epsilon_X + \epsilon_Y \leq \overline{\mathbb{E}}[X] - \overline{\mathbb{E}}[Y] - \epsilon_T \\ &0 \leq \delta_{X,Y} \leq 1 \end{aligned} \tag{3.4}$$

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

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 such term in the specification. This simplification is carried out using the assumption $A_\delta := \delta_i = \delta_j$ for all *elementary* subexpressions. As we do not make this assumption, we can prove the following key theorem.

Theorem 2. *Given a threshold probability Δ for a specification ψ, let the stopping time for AVOIR be $\mathcal{T}$ and stopping time with the A_δ assumption be $\mathcal{T}^+$. Then $\mathcal{T} \leq \mathcal{T}^+$*

See Appendix 3.11 for the proof.

3.3.4 Visualization for Interactive Refinement

Using our specification framework as a backend, we built an interactive application for analysis and refinement of specifications provided in our grammar. Specifically, fairness specifications can be naturally parsed into a tree because of the structure of the grammar. Each node of the tree represents some sub-expression in the syntax tree of the overall specification. These nodes allow a user of AVOIR to interactively audit and tune the specification definition. To create the visualization, we use Vega (Satyanarayan et al., 2015), a declarative JSON-based visualization grammar. We log the estimates during runs of AVOIR and then output the grammar in a tabular JSON-format that contains a row for each grammar element and its associated evaluations. This tabular data is used by our Vega specification to produce the visualizations. 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. 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 interactiving with and deciding how to refine a specification We provide a detailed example of how these interactions can help AVOIR users choose an appropriate fairness metric in Section 3.4.

3.4 Case Studies

The following text describes two real-world scenarios for AVOIR. We implement both Verifair and our optimized bounds within the AVOIR framework, denoted as AVOIR-VF and AVOIR-OB, respectively. An important case study on the COMPAS dataset can be found in Appendix 3.14.

3.4.1 Rate My Profs

In this section, we provide a detailed black-box machine learning model (ML) based case study on a real-world dataset. In this case study, we use the rate my professors (RMP) dataset released by 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) 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:

[9]https://simpletransformers.ai/

(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 3.3b shows that AVOIR-OB[10] can provide a guarantee in $\mathbf{2.5}\%$ 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.

3.4.2 Adult Income

In this case study we use the Adult income dataset (Kohavi, 1996) which has been used frequently in prior fairness-related work. The historical dataset labels individuals from the 1994 census as having a *high-income* ($> 50,000$ a year) or not ($\leq 50,000$ a year). In this case study, we look at a column of data as a black-box measurement (internally, we use a materialized view, details in Appendix 3.14.1).

US Federal laws mandate against race and sex based discrimination. Thus, the specification we start our analysis with is a group fairness property that monitors the difference of the proportions of individuals with sex recorded as male that have a high income to females that have a high income should be less than 0.5. In addition, we ensure that the difference between individuals with race marked as white and those without should have a difference of less than 0.5. 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 materialized view cannot achieve the required bound. We can then use our iterative refinement visualization tool to analyze different components of the specification. A developer would first interact with the left subtree of the specification. Due to paucity of space, this visualization is presented in Appendix 3.14.2. The plot for the corresponding data is shown in Figure 3.4a shows that guarantees cannot converge under the threshold with the given number of data samples. The developer can now choose to either reduce the guarantee (i.e. reduce δ) or increase the threshold. Next, analyzing the right subtree, the race group fairness term can be guaranteed to be under the threshold (Figure 3.4b). Using this information, the developer can make an intelligent decision to increase the threshold on the group fairness for term for sex. Suppose they increase it to 0.55 and rerun the analysis. OB is able provide a guarantee at this threshold within 870 steps, whereas VF can provide it at 960 steps, demonstrating a relative improvement of about $\mathbf{10.35}\%$. Additionally, the optimal δ split across the terms are $\approx (0.135, 0.36 * 10^4)$ which is far from the equal split allocated by VF. The reason for this split is because increasing the threshold for the first time provides the optimizer with additional legroom to better distribute the failure probabilities between the two terms.

[10] OB = Optimized Bounds

3.5 Discussion

The case studies presented in the previous section demonstrate the ability of our tools to provide vital context when deciding how to refine a model or fairness specification. Although this contextual information makes decisions easier, it is not always clear how one should alter a specification in light of a violation and its relevant context. To assist in these decisions, we are currently examining ways work to suggest edits that are likely to achieve the desired intent of a developer. Using our visual analysis tool for refinement, we can gather edits from developers and then use that data to learn iterative changes to the syntax tree of the specification. In addition to improving the usability of our tools for making fairness specification refinements, we also envision a more scalable framework. Our case studies look at a single model with respect to a single dataset. However, real-world deployment of machine learning often contain many clients with models that may differ. We take it as future work to study the efficient monitoring of machine learning behavior with respect to a fairness specification in a distributed context, enabling horizontal scalability. We believe techniques such as decoupling the observation of data and the reporting results from the monitoring of the results are promising and can lead to the desired scalability.

3.6 Conclusion

We present the AVOIR framework for easily defining and monitoring fairness specifications online and aids in the interactive refinement of specifications. AVOIR is easy to integrate within modern database systems but can also serve as a standalone system evaluating whether blackbox machine learning models are meeting 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. To assist in refinement of specifications, we develop an interactive visual analysis application within AVOIR. Through case studies, we demonstrate that AVOIR can provide users with insights that contribute directly to refinement decisions. Our framework builds the foundation for further improvements to the fairness specification, auditing and verification workflow. We plan to extend this work to provide intelligent specification refinement suggestions and support distributed machine learning settings. We also plan to explore the use of AVOIR for fair ranking problems and tailored database integration.

3.7 Reproducibility

To enhance the reproducibility of our work, on the theoretical side, all proofs (with the necessary assumptions) are provided in the appendix. Specifically, proofs for the inference engine are in Appendix 3.8, and proofs for the correctness of bounds are provided in Appendix 3.10. Theorem 2, which shows how AVOIR improves over prior work is proved in Appendix 3.11. To reproduce the results of the case studies in the paper, each case study is encapsulated inside a Jupyter notebook. These notebooks are attached along with the source code for AVOIR. In addition, all datasets used for generating results for the case studies are also attached in the submitted supplementary documentation. Finally, the model weights used for the *RateMyProfs* study for exact reproduction are provided in a dropbox folder hosted at https://www.dropbox.com/sh/n5o4vswnkxv34zr/AABthgLMaYL3MuA0KC39Z1G8a?dl=0.

3.8 Inference Rules

In Figure 3.5, we provide a set of rules that can be used to determining the constraints and guarantees associated with 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 curly brackets $\{\}$.

The proof of correctness for each inference rule starts from the assumptions above the horizontal line and derives the assertions below. These proofs ise ideas similar to those in (Bastani et al., 2019). We reproduce the proofs in Appendix 3.8.1 here for completeness. Note that the assertions in the base case (elementary subexpressions) can be arrived at by applying AIN.

3.8.1 Inference rules with Constraints

In Section 3.3.2 we provided the proofs for $X \pm Y$, $X > c$. In the following text we provide the proofs for the remainder of the inference rules.

3.8.1.0.1 Product Starting with $\phi_X = X : (\overline{\mathbb{E}}[X], \epsilon_X, \delta_X)$, $\phi_Y = Y : (\overline{\mathbb{E}}[Y], \epsilon_Y, \delta_Y)$. First, from union bound, both of these hold true with probability at least $1 - \delta_X - \delta_Y$. Then,

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

We have

$$\begin{aligned}
|\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y] - \mathbb{E}[XY]| &= |\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y] - \mathbb{E}[X]\mathbb{E}[Y]| && \text{(as } X, Y \text{ bernoulli)} \\
&= |\overline{\mathbb{E}}[X]\overline{\mathbb{E}}[Y] - \overline{\mathbb{E}}[X]\mathbb{E}[Y] + \overline{\mathbb{E}}[X]\mathbb{E}[Y] - \mathbb{E}[X]\mathbb{E}[Y]| \\
&= |\overline{\mathbb{E}}[X](\overline{\mathbb{E}}[Y] - \mathbb{E}[Y]) + \mathbb{E}[Y](\overline{\mathbb{E}}[X] - \mathbb{E}[X])| \\
&\leq |\overline{\mathbb{E}}[X]||(\overline{\mathbb{E}}[Y] - \mathbb{E}[Y])| + |\mathbb{E}[Y]||(\overline{\mathbb{E}}[X] - \mathbb{E}[X])| \\
&\leq |\overline{\mathbb{E}}[X]|\epsilon_Y + |\mathbb{E}[Y]|\epsilon_X \\
&\leq |\overline{\mathbb{E}}[X]|\epsilon_Y + (|\overline{\mathbb{E}}[Y]| + \epsilon_Y)\epsilon_X \\
&= |\overline{\mathbb{E}}[X]|\epsilon_Y + |\overline{\mathbb{E}}[Y]|\epsilon_X + \epsilon_X\epsilon_Y
\end{aligned}$$

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

3.8.1.0.2 Inverse/Inverse constr. Assume $X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right)$ and $\overline{\mathbb{E}} - \epsilon > 0$. Instead, in the constrained case, we start with only the prior assumption i.e., $X : \left(\overline{\mathbb{E}}, \epsilon, \delta\right)$ 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,

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

where the last step follows from the previous derivation and if $\mathbb{E}[X] - \epsilon_X > 0$. The latter condition enforces that the sign of the inequality does not change. VF adds this as a precondition; we add it as a post-constraint.

3.8.1.0.3 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.

3.8.2 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 aforementioned inference rules, we can derive the overall $\delta_T = \sum_i \delta_i$, along with a set of (say) K constraints

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

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[th] inequality subexpression (i.e. having 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 3.2, having n optimizaiton 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.

3.9 Concentration bounds

The adaptive Hoeffding inequality (Zhao et al., 2016; Bastani et al., 2019).

Theorem 3. *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 random variable on $\mathbb{N} \cup \{\infty\}$ such that $\Pr[\mathcal{T} < \infty] = 1$, and let

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

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

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

.

Theorem 3 provides a mechanism for choosing the stopping time using arbitrary methods for a fixed δ. Note that in general, any adaptive concentration inequality suffices; we use the Hoeffding inequality that does not depend on the empirical variance but is frequently used in scenarios dealing with bounded rvs. However, we use confidence intervals to visualize the evolution of sub-expressions (and overall specification) over the sequence of observations. For doing 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

$$\Pr\left[\exists n, S_t \geq f(t)\right] \leq \delta$$

Corollary 4.1. *For any $\delta > 0$,*

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

for any stopping time $\mathcal{T}$ if and only if

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

Proof. Follows directly from applying Theorem 4 to Theorem 3. □

Intuitively, Theorem 3 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, as long as the underlying distribution remains unchanged, the bound will hold prior to and after the stopping rule is enforced. Theorem 4.1 implies that once we choose an optimal bound for each subexpression, we can extend the bounds derived using Theorem 3 to following observations with the guarantees for the subexpressions still holding true.

3.10 Confidence Sequences

In this section, we show that the estimates generated from AVOIR form a confidence set (Theorem 1). First, we assume the existence of a concentration sequence for the mean of each elementary subexpression (eg., Theorem 3 can provide one). That is, we need a function $\epsilon(t, \delta)$ such that

$$\Pr[\forall t \geq 1, |\overline{\mathbb{E}}_t[X] - \mathbb{E}[X]| \leq \epsilon(t, \delta_X)] \geq 1 - \delta_X. \tag{3.5}$$

For convenience of exposition, we denote such adaptive inequality functions as AIN. For any AIN to be usable with AVOIR, we require $\epsilon(t, \delta)$ 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. For example, the adaptive Hoeffding inequality (Theorem 3) follows this assumption. Second, we assume that, except in degenerate cases, AVOIR terminates (see corollary 4.2 for termination criteria). We now prove Theorem 1.

Proof. First, we will prove that the estimates for *elementary* subexpressions are a confidence sequence. Following this, using the inference rules from Appendix 3.8, we will show that the estimates for every compound expression are also a confidence sequance.

3.10.0.0.1 Elementary subexpressions Consider a specification ψ consisting of *elementary* subexpressions $X_1, \ldots, X_n$. At stopping time $\mathcal{T}$, let

$$\phi^{\mathcal{T}}_{X_i} := X_i : (\overline{\mathbb{E}}_{\mathcal{T}}[X_i], \epsilon(\mathcal{T}, \delta_{X_i}), \delta_{X_i}) \tag{3.6}$$

be the stopping time estimates. Then, from the termination criterion, a solution to the optimization problem `OPT` exists, i.e,

$$\Delta \geq \sum_i \delta_{X_i} \tag{3.7}$$

The sequence of bounds claimed by AVOIR are

$$\epsilon_{X_i}(t) = \begin{cases} \epsilon(\Delta, t), & t < \mathcal{T}, \\ \epsilon(\delta_{X_i}, t), & t \geq \mathcal{T} \end{cases} \tag{3.8}$$

From equation 3.7 and the optimization constraint $\delta_i \in [0, 1]$ we have $\Delta \geq \delta_{X_i}$. From the non-decreasing behavior of AIN

$$\epsilon(\Delta, t) \leq \epsilon(\delta_i, t) \tag{3.9}$$

Now

$$\begin{aligned}
&\Pr[\forall t \geq 1, |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| \leq \epsilon_{X_i}(t)] \\
&\quad = 1 - \Pr[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)] \\
&\quad = 1 - \Pr\left[\bigcup_{t \geq 1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)\right\}\right] \\
&\quad = 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 \bigcup_{t \geq \mathcal{T}} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon_{X_i}(t)\right\}\right] \\
&\quad \text{(associativity of } \cup\text{)} \\
&\quad = 1 - \Pr\left[\bigcup_{t=1}^{\mathcal{T}-1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\Delta, t)\right\} \cup \bigcup_{t \geq \mathcal{T}} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\right\}\right] \\
&\quad \text{(From 3.8)} \\
&\quad = 1 - \Pr\left[\bigcup_{t=1}^{\mathcal{T}-1} \left\{|\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))]\right\} \cup \right. \\
&\quad \left. \bigcup_{t \geq \mathcal{T}} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\right\}\right] \text{ (Using 3.9)} \\
&\quad = 1 - \Pr\left[\bigcup_{t=1}^{\mathcal{T}-1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| \in (\epsilon(\Delta, t), \epsilon(\delta_{X_i}, t))]\right\} \cup \bigcup_{t \geq 1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\right\}\right] \\
&\quad \text{(Rearranging)} \\
&\quad \geq 1 - \Pr\left[\bigcup_{t \geq 1} \left\{|\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\right\}\right] \\
&\quad = 1 - \Pr\left[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_i] - \mathbb{E}[X_i]| > \epsilon(\delta_{X_i}, t)\right] \\
&\quad \geq 1 - \delta_{X_i}
\end{aligned}$$

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

3.10.0.0.2 Compound subexpressions Consider a non-specification compound <ETerm> C_j consisting of *elementary* subexpressions with indices $_j = \{\{j_1, j_2, \ldots, j_M\}\}$ as the decision r.v.s,

i.e, $X_{j_1} \ldots, X_{j_M}$. Note that $_j$ is a multiset as the same expression could occur multiple times within C_j. At stopping time $\mathcal{T}$,

$$\phi^{\mathcal{T}}_{C_j} : (\overline{\mathbb{E}}_{\mathcal{T}}[C_j], \delta_{C_j}, \epsilon_{C_j}) \tag{3.10}$$

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

$$\overline{\mathbb{E}}_t[C_j], \delta_{C_j}(t), \epsilon_{C_j}(t) = \text{INFER}(\phi^{\mathrm{t}}_{\mathrm{X_{j_1}}}, \ldots, \phi^{\mathrm{t}}_{\mathrm{X_{j_M}}}) \tag{3.11}$$

the values inferred at time step t, where INFER denotes the inference rules. Now,

$$\begin{aligned}
&\Pr[\exists t \geq 1, |\mathbb{E}[C_j] - \overline{\mathbb{E}}[C_j]| > \epsilon_{C_j}(t)] \\
&\quad \leq \Pr\left[\bigcup_{i=1}^{M} \exists t \geq 1, \neg\phi^t_{X_{j_i}}\right] && \text{(From 3.11)} \\
&\quad \leq \sum_{i \in_j} \Pr\left[\exists t \geq 1, \neg\phi^t_{X_{j_i}}\right] && \text{(union bound)} \\
&\quad = \sum_{i \in_j} \Pr\left[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_{j_i}] - \mathbb{E}_t[X_{j_i}| > \epsilon_{X_{j_i}}(t)\right] && \text{(definition of } \phi^t_{X_{j_i}}\text{)} \\
&\quad \leq \sum_{i \in_j} \delta_{X_{j_i}} && \text{(elementary subexpressions)} \\
&\quad \leq \delta_{C_j} && \text{(applying 3.11 for } t = \mathcal{T}\text{)}
\end{aligned}$$

Therefore $\epsilon_{C_j}(t)$ defines a $1 - \delta_{C_j}$ confidence sequence for $\mathbb{E}[C_j]$

A similar proof can be constructed for any <spec>. Consider any specification ψ_k. Let

$$\psi^t_k : (\hat{b}_{\psi_k}(t), \delta_{\psi_k}(t)) \tag{3.12}$$

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 $_k = \{\{k_1, \ldots, k_D\}\}$. 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

$$\hat{b}_k(t), \delta_{\psi_k}(t) = \text{INFER}(\phi^{\mathrm{t}}_{\mathrm{X_{k_1}}}, \ldots, \phi^{\mathrm{t}}_{\mathrm{X_{k_D}}}) \tag{3.13}$$

We have

$$\begin{aligned}
&\Pr[\exists t \geq 1, b_k \notin \hat{b}_k(T)] \\
&\quad \leq \Pr\left[\bigcup_{i=1}^{D} \exists t \geq 1, \neg\phi^t_{X_{k_i}}\right] && \text{(From 3.13)} \\
&\quad \leq \sum_{i \in_k} \Pr\left[\exists t \geq 1, \neg\phi^t_{X_{k_i}}\right] && \text{(union bound)} \\
&\quad = \sum_{i \in_j} \Pr\left[\exists t \geq 1, |\overline{\mathbb{E}}_t[X_{k_i}] - \mathbb{E}_t[X_{k_i}| > \epsilon_{X_{k_i}}(t)\right] && \text{(definition of } \phi^t_{X_{j_i}}\text{)} \\
&\quad \leq \sum_{i \in_j} \delta_{X_{k_i}} && \text{(elementary subexpressions)} \\
&\quad \leq \delta_{\psi_k} && \text{(applying 3.11 for } t = \mathcal{T}\text{)}
\end{aligned}$$

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

3.11 Optimality

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 3.2. 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 terminate at step $\mathcal{T}^+$, but may find a feasible solution at an earlier step, i.e. $\mathcal{T} \leq \mathcal{T}^+$.

□

Corollary 4.2. *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.

- 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 $C_k > c$), $\overline{\mathbb{E}}[C_k] \neq c$, $C_k \neq c$

□

3.12 Implementation

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[11] 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 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.

[11] https://streamlit.io/

Algorithm 1 AVOIR Algorithm

Input: Δ, ψ ▷ Δ, Specification
Output: 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. 3.5)
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

3.12.1 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 3.6 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.

3.13 AVOIR in Database Setting

In the database literature researchers (**?**), 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 (**???**), 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[12] 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.

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

3.14 Additional Case Studies

3.14.1 Materialized views

A materialized view is constructed by querying the dataset to select for employees of the federal government. We simulate the materialized view using a pandas[13] dataframe wrapped in a python class to monitor updates and run AVOIR for any monitored specification.

3.14.2 Interaction through Vega

Figure 3.6 shows a subtree of the specification visualized through Vega. A developer analyzing this spec can click on the top pink node to see the evolution of the sex fairness part of the specification and superimpose the threshold. The threshold is set to be evaluated with every 5 new data points added to the materialized view. Clicking on the corresponding element in the right subtree, the developer can see Figure 3.4b.

3.14.3 COMPAS Risk Assessment via Materialized Views

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 a software released by Northpointe and widely used across the United States for making sentencing decisions. In 2016, **?** 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 a period of 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 (**?**) strongly rejecting the claims made by the ProPublica article; instead, they claimed that **?** made several statistical and technical errors in the report. In this case study, we use AVOIR to study the claims made by the two aforementioned reports. First, we start with the data released by ProPublica and load it into a pandas-simulated DB. We then create a materialized view that corresponds to the preprocessing steps used in the publicly available ProPublica analysis notebook[14]. We look at "Sample A" (**?**), 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 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 **?**, `recid` is an indicator for re-offending within 2 years of first arrest, and a 10%-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 in a sequential,

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

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

online fashion, the assertion for violation of the specification cannot be made with the required failure guarantee.

By analyzing the components using the visualization tool, one can glean that AVOIR is unable to optimize since the lower FPR in the denominator (FPR for Caucasian individuals) increasing the overall variance and limiting 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

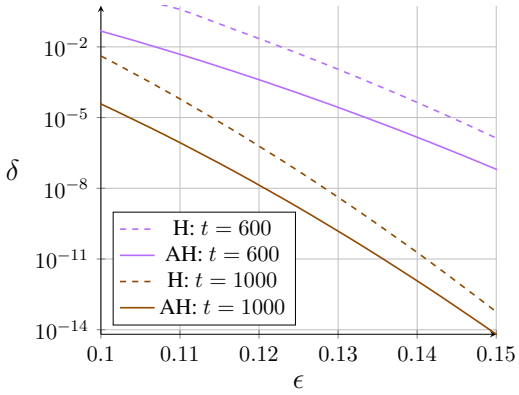
we find that indeed, the specification is violated with a confidence of over $1-\Delta = 0.9$ probability, and AVOIR can detect this violation within about half the number of available assessments (3350 steps) when run in an online setting. Figure 3.7a demonstrates the progression of the tracked expectation term. Thus, if deployed with the corrected specification, AVOIR would be able to alert Northpointe of a violation/potentially-biased decision making tool.

The Northpointe report (**?**) makes several claims about the shortcomings of this analysis, but one of the primary claims is that **?** used an analysis based on "Model Errors" rather than "Target Population Errors". In Fairness metric terms, this refers to the difference between a False Positive Rate (FPR) balance vs False Discovery Rate (FDR) balance, i.e. balancing for predictive parity over predictive equality. In probabilistic terms, the difference amounts to comparing $P(\hat{Y} = 1|Y = 0, g = 1, 2)$ (FPR) vs $P(Y = 0|\hat{Y} = 1, g = 1, 2)$ (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 run reproduce the corresponding preprocessing steps and run both versions (numerator and denominator being Caucasian) versions of the corresponding specification under the same setup as earlier. We find that despite the point estimate being within the required threshold, neither version can be guaranteed with the required confidence with the given data. Due to paucity of space, we describe only one of the two variants with the corresponding figure (Figure 3.7b).

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

We note that the Northpointe report (**?**) does not provide confidence intervals for their claim either. 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 both academic and journalistic publications (**??**). Seminal work by **?** proved the impossibility of simultaneously guaranteeing certain combinations of fairness metrics. While AVOIR cannot solve this problem, its usage can help provide explicit guarantees on defined metrics. The specification grammar also provides a simple mechanism for independent replication of claims. We conclude this case study by noting that AVOIR lends itself to successful analysis that is not possible with the Verifair implementation available online.



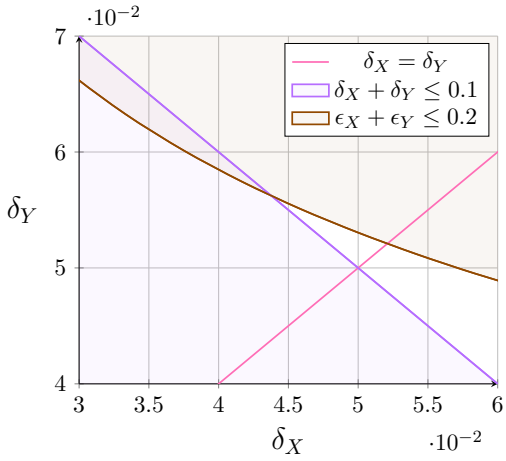
(a) At the same concentration ϵ, lower failure probability δ for the majority class.

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

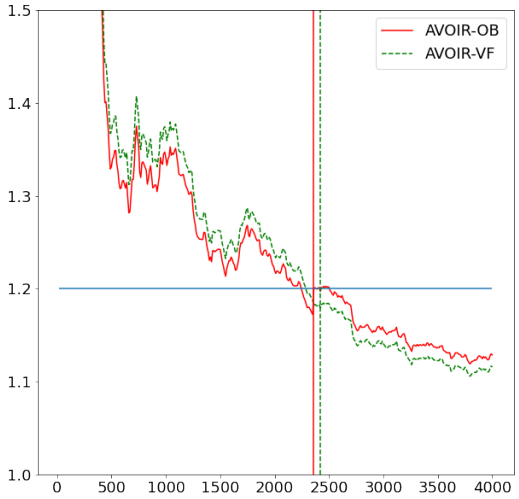
$\langle ETerm \rangle$ ::= $\mathbb{E}[\langle E \rangle]$
| $\mathbb{E}[\langle E \rangle | \langle E \rangle]$
| c $\in \mathbb{R}$
| $\langle ETerm \rangle$ $\{+, -, \times, \div\}$ $\langle ETerm \rangle$

(b) $\langle E \rangle$ refers to pure expressions and $\langle comp-op \rangle$ is any arbitrary comparison operator $\in \{>, <, =, \neq\}$.

Figure 3.2: *(Left)* Failure probability of Bernoulli r.v. being concentrated around its mean for different n. H = (online) Hoeffding, AH = Adaptive Hoeffding. *(Right)* Modified Grammar for Specification.

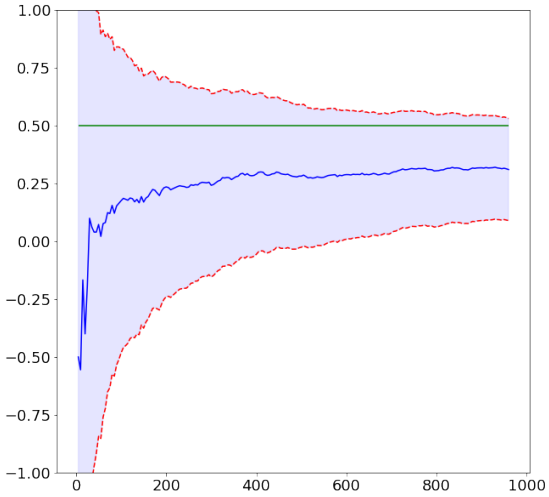


(a) No solution exists with additional constraint $A_\delta : \delta_X = \delta_Y = \Delta/2$ - common assumption in prior work.

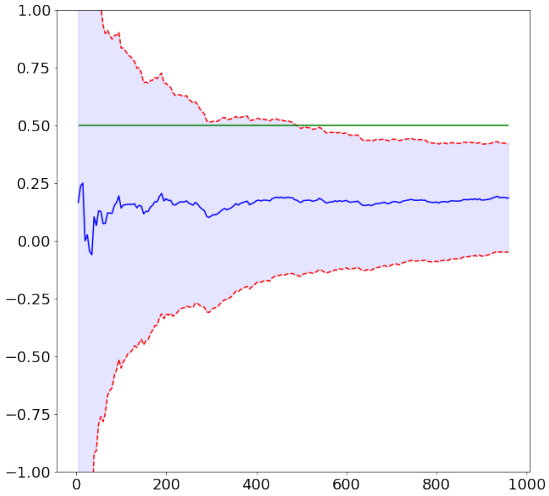


(b) Bounds for first half of a gender-fairness specification generated by AVOIR-OB and AVOIR-VF.

Figure 3.3: *(Left)* AVOIR finds a solution for a *theoretical* scenario with $\delta_X + \delta_Y \leq \Delta$ under constraint $\epsilon_X + \epsilon_Y \leq \epsilon_T$ *(Right)* For *RateMyProfs*, a real-world dataset, the vertical lines show the step at which the methods can provide a guarantee of failure for the upper bounds with $\Delta <= 0.05$.



(a) Group fairness for sex. Difference in ratio of high income earners in left subtree for initial specification.



(b) Group fairness for race (difference in ratio of high income earners) in right subtree for initial specification.

Figure 3.4: *(Left)* Red dotted lines the upper bound of the value cannot be guaranteed to be under the threshold at the specified failure probability. *(Right)* Guarantee possible with given data.

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

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

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

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

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

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

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

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

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

Figure 3.5: 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).

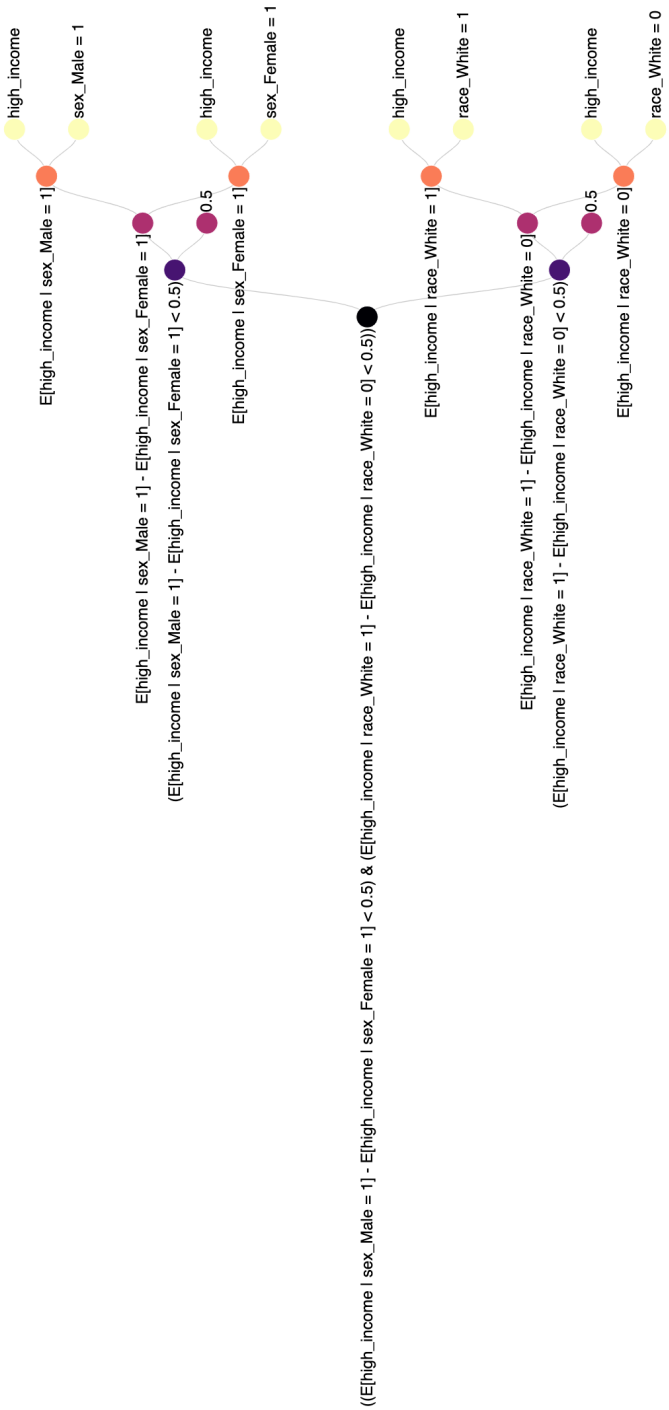
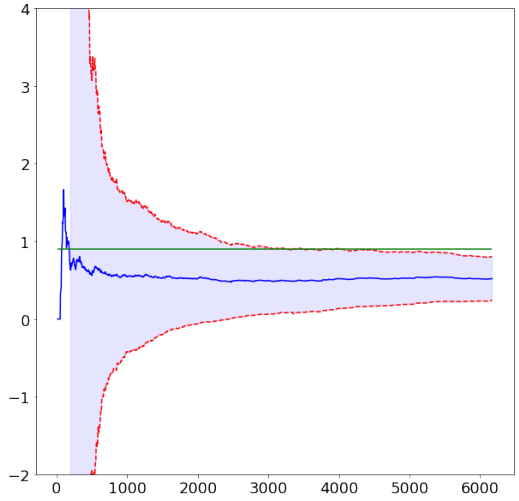
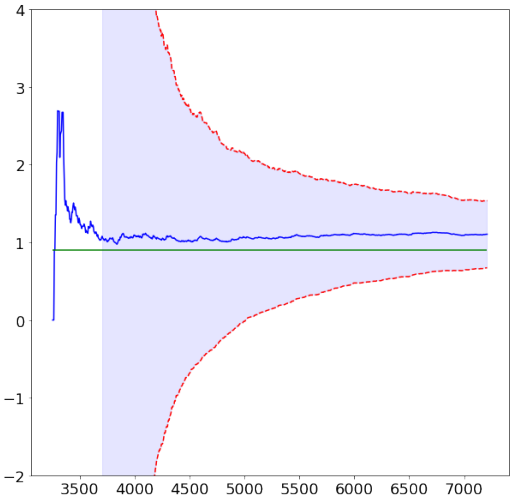


Figure 3.6: Tree of initial specification before refinement in the adult income dataset.



(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 3.7: COMPAS dataset case study.

Chapter 4: Stylometry on the Darkweb

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. 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 darknet 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.

4.1 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.

4.2 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).

TODO: Weird DBLP cites.

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[15] 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 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.

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

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).

4.3 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 4.1. 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 4.1: 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.

4.4 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 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.

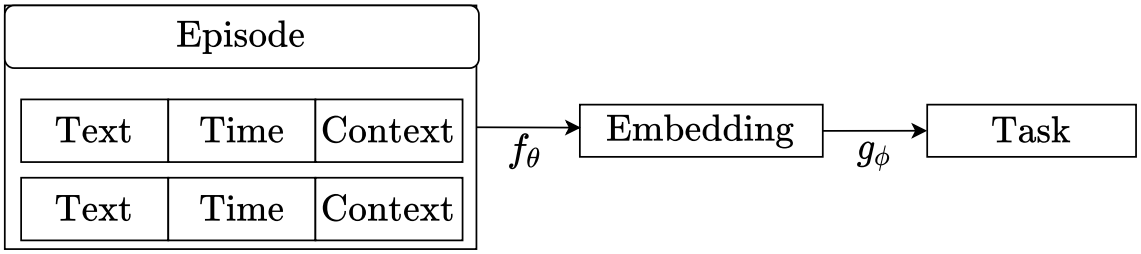


Figure 4.1: Overall SYSML Workflow.

4.4.1 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,

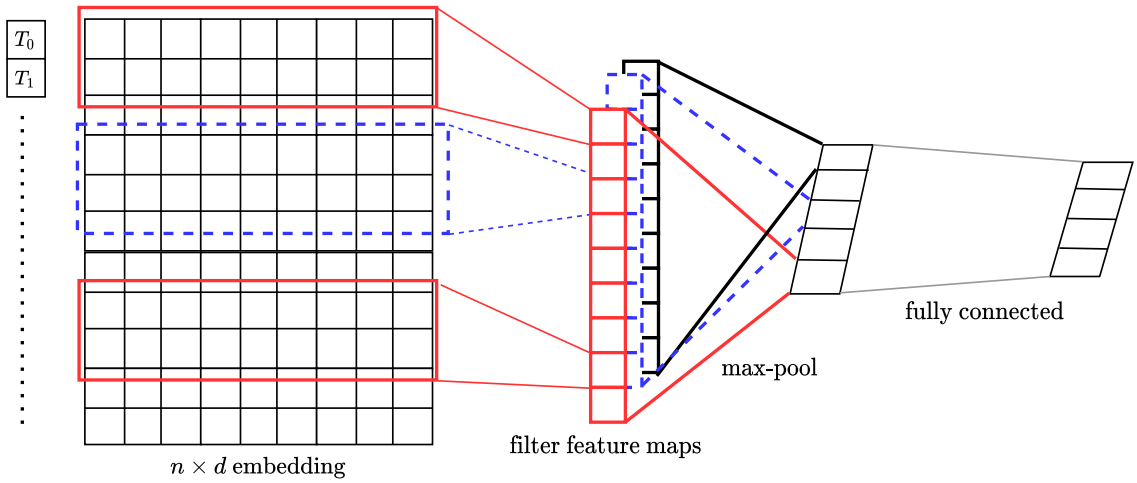


Figure 4.2: Text Embedding CNN (Kim, 2014).

we use f sliding window filters, with filters sized $F = \{2, 3, 4, 5\}$ 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 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 3 (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 $\phi(v) : V \to T_V$, $\psi(e) : E \to T_E$, 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

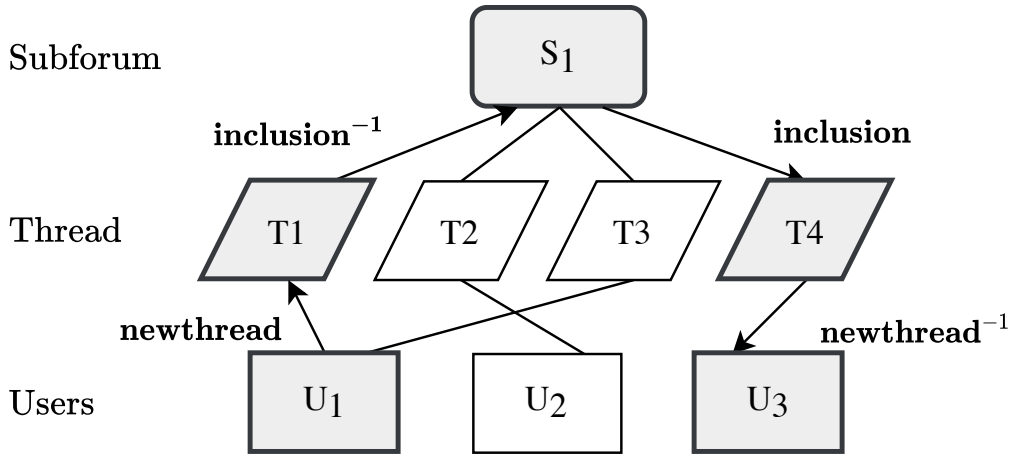


Figure 4.3: An instance of meta-path 'UTSTU' in a subgraph of the forum graph.

can incorporate specific semantic relationships into node embeddings. For example, Figure 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 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).

4.4.2 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., 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

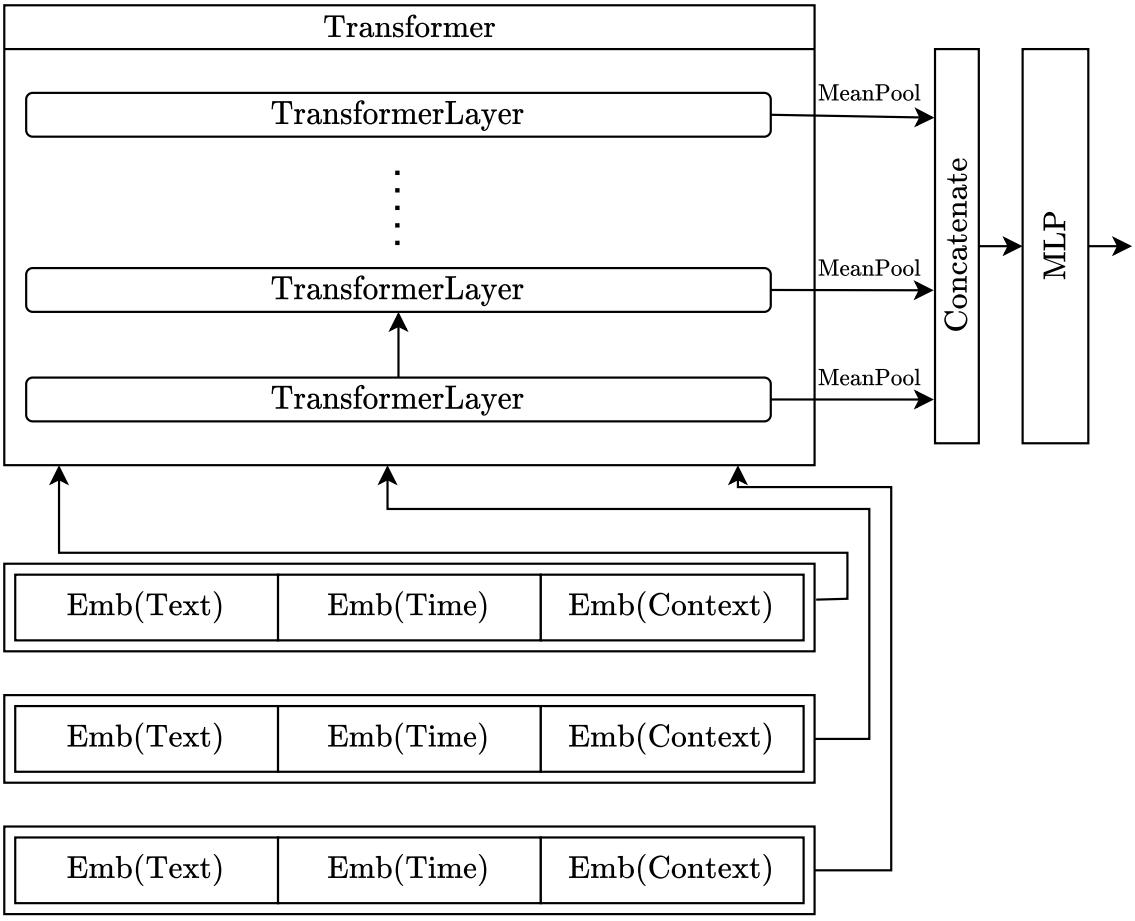


Figure 4.4: Architecture for Transformer Pooling.

brevity (Figure 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 $f_\theta : \{(t, \tau, c)\}^L \to \mathbb{R}^E$.

4.4.3 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., 2019).

4.4.4 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 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.

4.4.5 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 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: $\mathcal{L} = \mathop{\mathbb{E}}_{\mathcal{T}_i \sim \mathcal{T},\, \mathcal{E} \sim \mathcal{T}_i} [\mathcal{L}_i(\mathcal{E})]$.

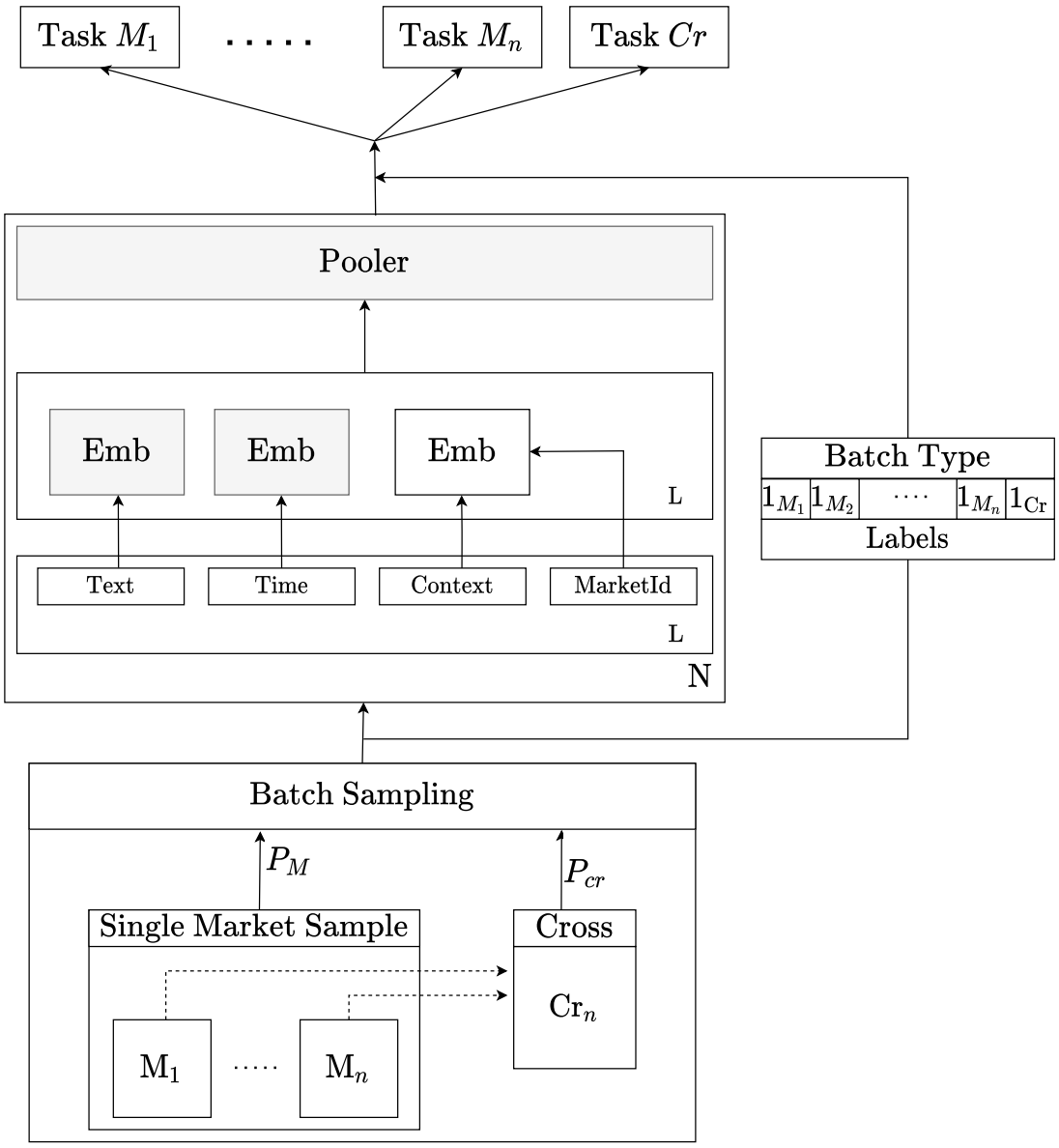


Figure 4.5: Multi-task setup. Shaded nodes are shared

4.5 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 $E = \{e_1, \dots e_n\}$ and let $Q = \{q_1, q_2, \dots q_\kappa\} \subset E$ be the sampled subset. We computed the cosine similarity of the query episode embeddings with all episodes. Let $R_i = \langle r_{i1}, r_{i2}, \dots r_{in} \rangle$ 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 $\langle r_{i1}, \dots, r_{ik} \rangle$. R@k denotes the mean of these recall values over all the query samples.

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

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 4.2

4.6 Analysis

4.6.1 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 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 (**?**). 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 4.2 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

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 4.2: 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.

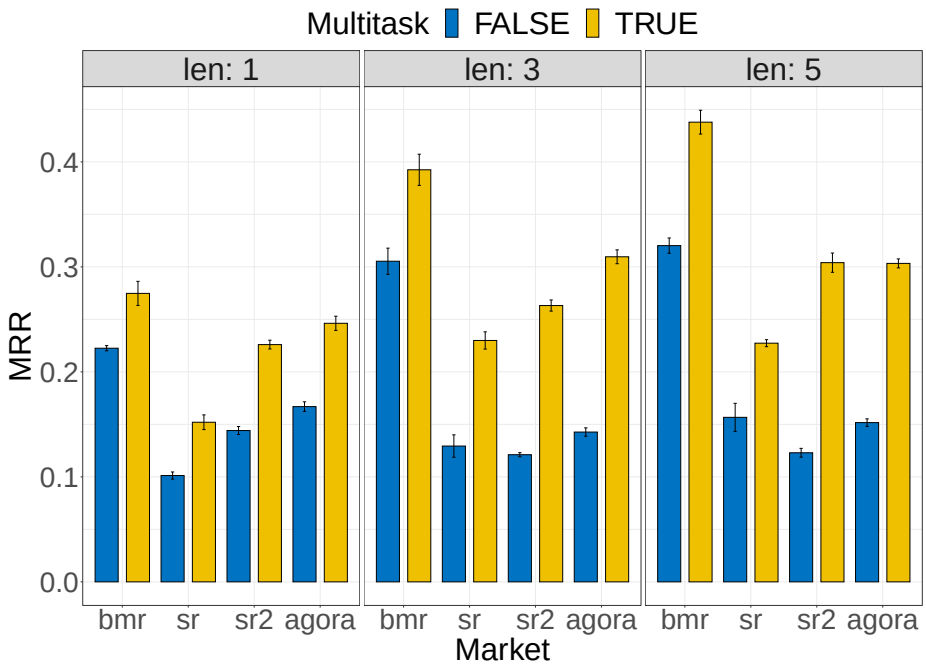


Figure 4.6: Drill-down: one-at-a-time vs. multitask.

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.

Multitask Our second key contribution is the multitask setup. Table 4.2 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 4.6 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 (**??**). We experimented with various state-of-the-art metric learning methods (§4.4.3) 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 4.7). Across all lengths, SM is significantly better (WMW: $p < 1e - 8$) and therefore we use SM in SYSML.

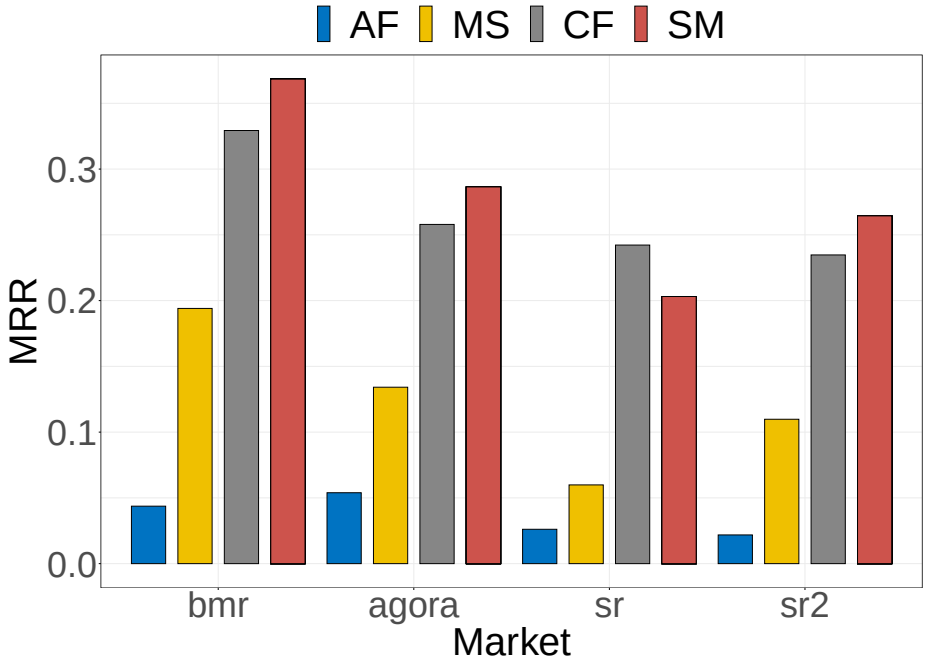


Figure 4.7: Task comparison: SM and CF are better performing two methods, with SM better in 3 of 4 cases.

4.6.2 Novel Users

The dataset statistics (Table 4.1) 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

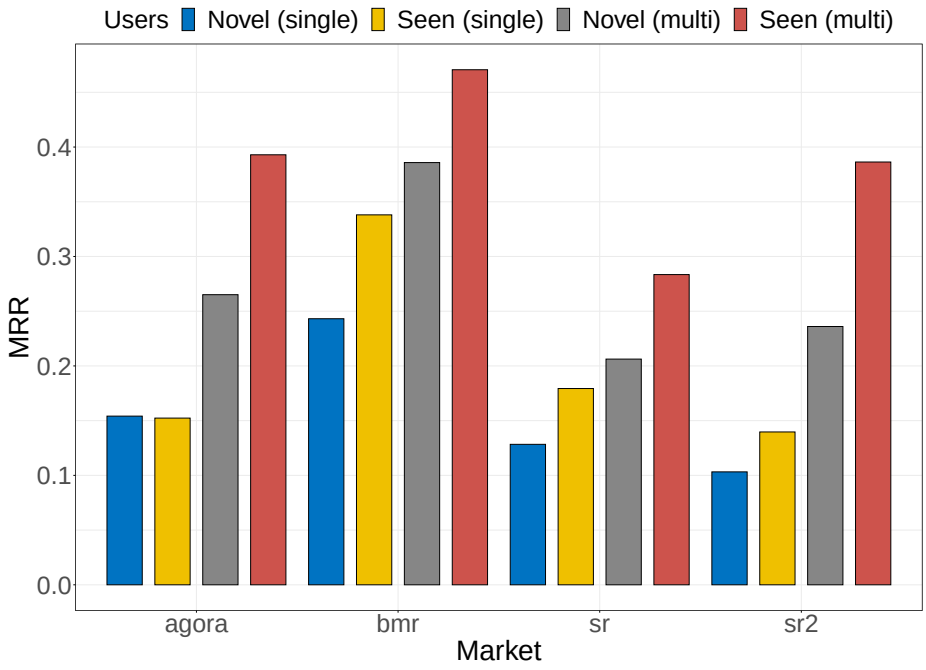


Figure 4.8: Lift on the multitask setup across users.

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 4.8 shows the comparison of MRR on these two samples against the 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 4.9 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.

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 4.3: Additional results for 7 posts per episode

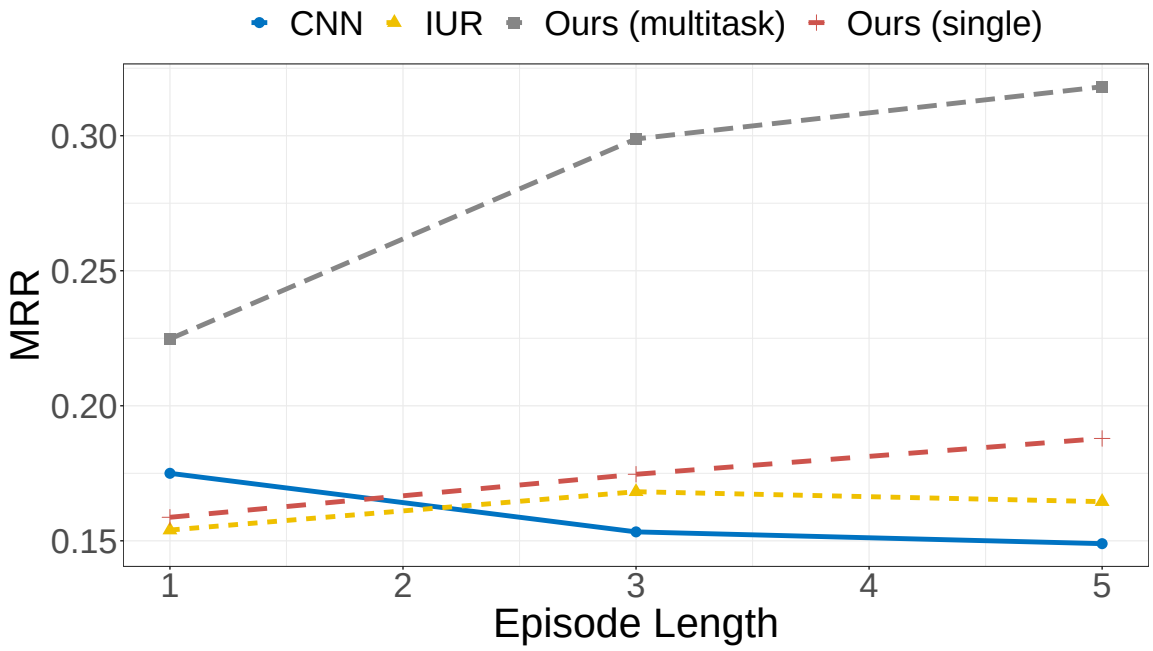


Figure 4.9: SYSML is more effective at utilizing multi post stylometric information

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 4.4: Additional results for 9 posts per episode

From Figure 4.10, 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 4.3 and 4.4 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.

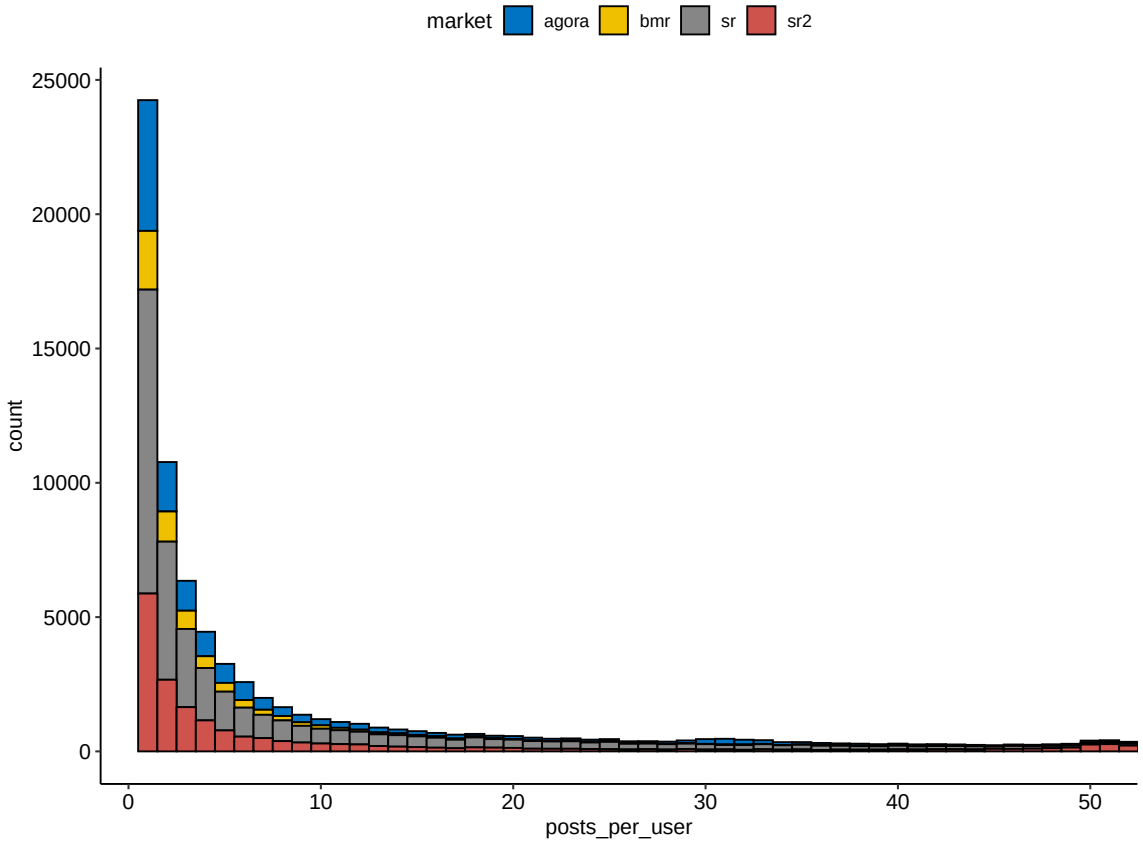


Figure 4.10: Frequency of number of posts per user

4.7 Case Study

4.7.1 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 4.5.

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[16]) who frequently provided information on security, where character n-grams corresponding to 'PGP', 'Key', 'security' are frequent (Table 4.6). 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 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 4.6 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 output with respect to the input features along a path from some reference baseline value (in our case, all `[PAD]` tokens) to the

[16] pseudonym

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

Table 4.5: Examples of highly identifiable posts.

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 4.6: Integrated Gradient based attribution of posts

input feature. In Table 4.6, 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).

4.7.2 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 4.11 shows a UMAP projection for T200. Users of each market are colored by sequential values of a single hue (i.e., reds - SR2, blues - 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.

TODO: Add additional results from Darkweb folder discussion.

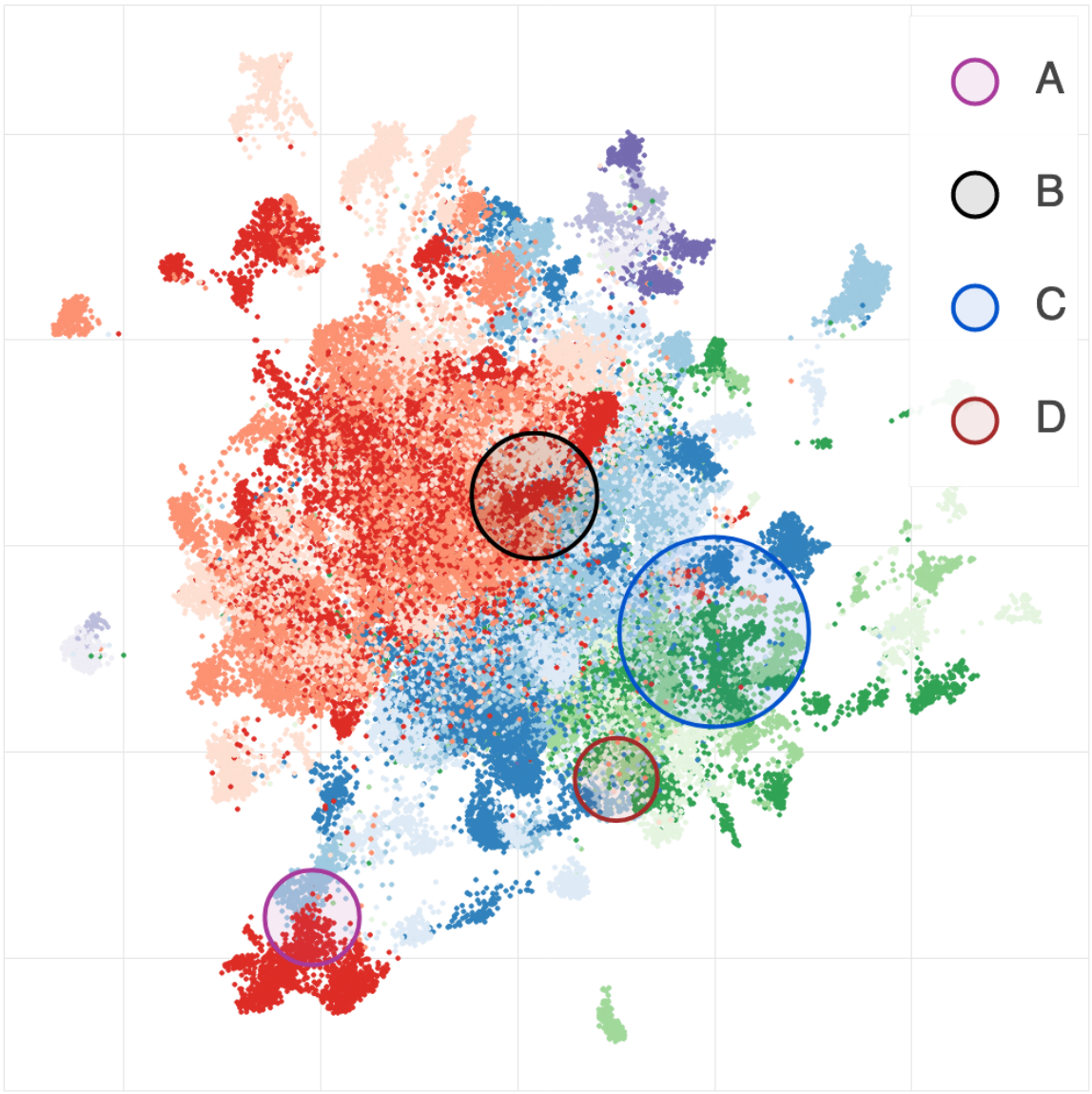


Figure 4.11: UMAP visualization of cross dataset embeddings for the top 200 authors, one hue per market. Circles denote the same user in two different markets.

4.8 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 (§4.3) 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 (§4.7.1) 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.

4.9 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 5: 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*, *online monitoring*, and *domain adaptation* to build more adaptive machine learning systems. We now discuss how we plan to extend these techniques in future work.

5.1 Large Scale Structure-aware Authorship Attribution

In this extension, our goal is to test the generalizability of our findings related to the improvements offered by utilizing forum graph structures on the darkweb (Chapter 4). 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 others (Rivera-Soto et al., 2021). Their findings imply that the diversity of topics expressed and number of unique users in one domain play a role in explaining why this domain may transfer better to others. However, this work does not utilize the additional structure and context present in these domains. We posit that these graphs can provide information orthogonal to that which is already present in the text. In scenarios with an abundance of text/users, these graph structures help improve authorship identification within a single domain, and also help better generalize across domains. 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 our 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 that help with *domain generalization*.

5.1.1 Graphs in Authorship Attribution

Plan

- Describe the structure of facebook, reddit, twitter, phpBB forums, twitch
- Show which meta structure parts are common
- Describe the approach (separate context and structure).
- Describe the upper bound (direct Graph model on reddit)

Prior to the prevalence of neural network approaches, seminal work in computational authorship attribution (Stamatatos, 2009) often used syntax-based features included, inter alia, part-of-speech tags, phrase structures, and syntactic error-based features. These features may improve neural

Figure 5.1: Graph Structures on Online Content Platforms

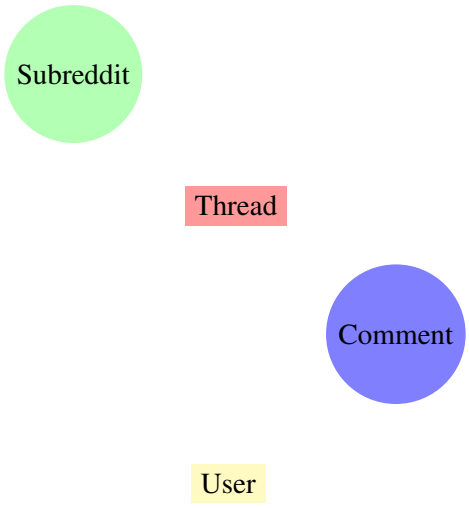


Figure 5.2: Metagraph of Reddit used for preliminary analysis.

authorship attribution models but are not the graph structures that are focal in our analysis. We focus 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 platforms that face issues in content moderation, where authorship identification has the potential to play an important role. Figure 5.1 demonstrates that while individual platforms may differ, there exists a shared, underlying, meta-structure, which can be used to identify patterns that aid in stylometric analyses. In the following section, we focus on a specific platform - Reddit - and utilize its structure to provide preliminary evidence its potential.

5.1.2 Preliminary Analysis: Reddit Graph-aware Authorship Attribution

We collect data from a snapshot of Reddit comments over one month (Aug 2016). Figure 5.2 describes the meta graph that we used for analysis.

Table 5.1

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

Table 5.1: Dataset used for preliminary analysis of graphs for authorship attribution on Reddit.

5.1.3 Future Directions

5.2 Interpretable Stylometric Modeling

5.3 Monitoring Metrics in Non-iid Settings

5.4 Project Schedule and Timeline

Chapter 6: Conclusion

Bibliography

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. 27, 31, 40

Aws Albarghouthi and Samuel Vinitsky. 2019. Fairness-aware programming. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 211–219. 4, 25, 26, 27, 31, 33, 40

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. vi, 4, 5, 50, 51, 53, 54, 55, 59, 60, 68

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*. 10

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. 4, 5

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. viii, 4, 25, 26, 27, 28, 29, 31, 34, 36, 40, 47

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. 25, 26

Tim Berners-Lee, Roy T. Fielding, and Larry M Masinter. 2005. Uniform Resource Identifier (URI): Generic Syntax. RFC 3986. 7, 9

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. 51

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. 8, 10, 12

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. 13

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*. 2, 10

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. 68

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. 65

Michael Robert Brennan and Rachel Greenstadt. 2009. Practical attacks against authorship recognition techniques. In *IAAI*. 65

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*. 68

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. 9

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75. 3, 52

Alessandro Castelnovo, Riccardo Crupi, Greta Greco, and Daniele Regoli. 2021. The zoo of fairness metrics in machine learning. *arXiv preprint arXiv:2106.00467*. 28

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. 10

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124. 1, 9

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. 7

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*. 3

US Congress. 2019. Hr 2231–algorithmic accountability act of 2019. 24

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. 10

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. 57

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. 1

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. 3, 8, 9, 10, 11, 12, 13, 14, 17, 31, 55

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. 52

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. 51, 54, 55

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. 10

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. 51

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. 51

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. 51

Georgia Frantzeskou, Efstathios Stamatatos, Stefanos Gritzalis, Carole E Chaski, and Blake Stephen Howald. 2007. Identifying authorship by byte-level n-grams: The source code author profile (scap) method. *International Journal of Digital Evidence*, 6(1):1–18. 5

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. 51

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. 25

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. 10, 15

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. 7

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. 25, 26

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*. 25, 26

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. 2, 25, 26

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press. 1

Google. Making the world's information safely accessible. 8

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. 2

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*. 10

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*. 1

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. 51

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*. 1

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. 29

HelpNetSecurity. 2019. Phishing attacks at highest level in three years. 8

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). 1

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)*. 24

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780. 10

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. 51

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. 52

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. 30

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. 15

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. 7, 11

HuggingFace. Transformers - state-of-the-art natural language processing for pytorch and tensorflow 2.0. 17

Chip Huyen. 2022. *Designing Machine Learning Systems*. " O'Reilly Media, Inc.". 2, 5

Federal Bureau of Investigation. 2019. 2019 internet crime report. 8

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. 1

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. 10

Patrick Juola. 2006. Authorship attribution. *Found. Trends Inf. Retr.*, 1(3):233–334. 4, 50

B Justice Srikrishna. 2018. A free and fair digital economy: Protecting privacy, empowering indians. 24

Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. 2020. Learning and evaluating contextual embedding of source code. In *International Conference on Machine Learning*. 9

Moniba Keymanesh, Tanya Berger-Wolf, Micha Elsner, and Srinivasan Parthasarathy. 2021. Fairness-aware summarization for justified decision-making. *arXiv preprint arXiv:2107.06243*. 31

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. ix, 51, 54

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. 14, 17

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. 15

Ron Kohavi. 1996. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *KDD*, volume 96, pages 202–207. 32

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. 51, 68

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*. 16

Hung Le, Quang Pham, Doyen Sahoo, and Steven C H Hoi. 2018. URLNet: Learning a URL Representation with Deep Learning for Malicious URL Detection. Technical report. 8, 10, 13, 17

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444. 1

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. 3, 16

Tal Linzen and Marco Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7:195–212. 1

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*. 3, 9, 10, 11, 12, 13, 14

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. 5

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. 7

Haspelmath Martin. 2017. The indeterminacy of word segmentation and the nature of morphology and syntax. *Folia linguistica*, 51(s1000):31–80. 2

James Martin. 2014. *Drugs on the dark net: How cryptomarkets are transforming the global trade in illicit drugs*. Springer. 50

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. 7

Meta. Pytorch - from research to production. 17

Microsoft. Microsoft defender smartscreen. 8, 11

Tomas Mikolov, Kai Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space. In *ICLR*. 55

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. 10

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. 24

E.F. Moore. 1956. Gedanken-experiments on sequential machines. *Automata Studies, Princeton University Press*, 129-153. 25

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. 10

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. 52, 68

Office of the Director of National Intelligence. 2022. Human interpretable attribution of text using underlying structure. 5

Manoj Niverthi, Gaurav Verma, and Srijan Kumar. 2022. Characterizing, detecting, and predicting online ban evasion. In *Proceedings of the ACM Web Conference 2022*, pages 2614–2623. 5

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. 53, 68

Central Digital Office and Data. 2021. Algorithmic transparency standard. 24

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*. 17

European Parliament. 2018. 2018 reform of eu data protection rules. *European Commission*. 24

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). 7, 11

Geoffrey K Pullum and Barbara C Scholz. 2002. Empirical assessment of stimulus poverty arguments. *The linguistic review*, 19(1-2):9–50. 1

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI. 9, 13

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. 50

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. 52

Alec Ratdford, Karthik Narasimhan, Tim SSalimans, and Ilya Sutskeyver. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI. 9

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. 7, 11

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. 2, 7, 10

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. 4, 5, 69

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. 8, 10

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *ArXiv preprint*, abs/1706.05098. 52

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. 51

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. 1

Doyen Sahoo, Chenghao Liu, and Steven C. H. Hoi. 2017. Malicious URL Detection using Machine Learning: A Survey. Technical Report 1. 7, 8, 10

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. 31, 40

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. 13

Michael Lee Scott. 2000. *Programming language pragmatics*. Morgan Kaufmann. 1

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. 13, 51

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. vi, 4, 50, 51, 59, 60

Ravid Shwartz-Ziv and Amitai Armon. 2022. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90. 1

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. 17

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. 2

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. 24

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556. 69

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. 65

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. 52

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. 7, 8, 10, 13, 17

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433. 9, 14

Sebastian Thrun. 1998. Lifelong learning algorithms. *Learning to Learn*. 2

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. 25

Jacob Tyo, Bhuwan Dhingra, and Zachary C Lipton. 2022. On the state of the art in authorship attribution and authorship verification. *arXiv preprint arXiv:2209.06869*. 5

Richard Vanderford. 2022. New york's landmark ai bias law prompts uncertainty. *WSJ*. 24

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. 8, 10, 12, 55

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*. 1

Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *2018 ieee/acm international workshop on software fairness (fairware)*, pages 1–7. IEEE. 25, 26, 43, 44

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. 29

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. 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. 9

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. 57

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*. 2

Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R. Scott. 2019. 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. 57

Max Welling and Thomas N Kipf. 2016. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*. 1

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. 9

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*. 1

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. 13

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. 7, 10, 15

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. 9

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. 26

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. 25

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28. 10

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. 51

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. 13

Shengjia Zhao, Enze Zhou, Ashish Sabharwal, and Stefano Ermon. 2016. Adaptive concentration inequalities for sequential decision problems. *Advances in Neural Information Processing Systems*, 29. 5, 27, 30, 36, 37

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. 3

Zvelo. 2020. The rise of single-use phishing urls and the need for zero-second detection. 8