



Calhoun: The NPS Institutional Archive
DSpace Repository

NPS Scholarship

Theses

2024-12

TOR NETWORK VIDEO FINGERPRINTING OVER RESIDENTIAL WI-FI AND CONGESTED NETWORK INTERFACES

Falk, Theodore B.

Monterey, CA; Naval Postgraduate School

<https://hdl.handle.net/10945/73449>

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>



**NAVAL
POSTGRADUATE
SCHOOL**

MONTEREY, CALIFORNIA

THESIS

**TOR NETWORK VIDEO FINGERPRINTING OVER
RESIDENTIAL WI-FI AND CONGESTED
NETWORK INTERFACES**

by

Theodore B. Falk

December 2024

Thesis Advisor:
Co-Advisor:

Armon C. Barton
Timothy Walsh

Distribution Statement A. Approved for public release: Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			<i>Form Approved OMB No. 0704-0188</i>
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC, 20503.			
1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE December 2024	3. REPORT TYPE AND DATES COVERED Master's thesis	
4. TITLE AND SUBTITLE TOR NETWORK VIDEO FINGERPRINTING OVER RESIDENTIAL WI-FI AND CONGESTED NETWORK INTERFACES		5. FUNDING NUMBERS	
6. AUTHOR(S) Theodore B. Falk			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		10. SPONSORING / MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Statement A. Approved for public release: Distribution is unlimited.		12b. DISTRIBUTION CODE A	
13. ABSTRACT (maximum 200 words) Both ordinary, concerned citizens and malicious actors, such as extremist groups, use the Onion Router (Tor) to encrypt their web traffic and protect their online anonymity. This usage has led to an interest in attacks on Tor, such as video fingerprinting, which seeks to identify what video is being streamed solely by analyzing features in the encrypted traffic trace. This thesis seeks to continue a previous line of work in "Exploring the Capabilities and Limitations of Video Stream Fingerprinting" to use Convolutional Neural Networks to conduct video fingerprinting attacks on network traffic transmitted over Tor in an open-world environment. It will expand upon previous research using a realistic vantage point to collect data utilizing Wi-Fi and traffic over the network interface. The end goal of this line of research is to increase the Department of Defense's understanding of how much risk exists to the privacy of the users of the Tor network. While performance suffers compared to models trained on data sets collected under ideal conditions, we show that video fingerprinting is still viable in typical network conditions, including robustness to traffic congestion in all but the most extreme cases.			
14. SUBJECT TERMS TOR, onion routing, machine learning		15. NUMBER OF PAGES 61	
		16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18

THIS PAGE INTENTIONALLY LEFT BLANK

Distribution Statement A. Approved for public release: Distribution is unlimited.

**TOR NETWORK VIDEO FINGERPRINTING OVER RESIDENTIAL WI-FI
AND CONGESTED NETWORK INTERFACES**

Theodore B. Falk
Lieutenant Commander, United States Navy
BS, York College of Pennsylvania, 2013

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

**NAVAL POSTGRADUATE SCHOOL
December 2024**

Approved by: Armon C. Barton
Advisor

Timothy Walsh
Co-Advisor

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Both ordinary, concerned citizens and malicious actors, such as extremist groups, use the Onion Router (Tor) to encrypt their web traffic and protect their online anonymity. This usage has led to an interest in attacks on Tor, such as video fingerprinting, which seeks to identify what video is being streamed solely by analyzing features in the encrypted traffic trace. This thesis seeks to continue a previous line of work in “Exploring the Capabilities and Limitations of Video Stream Fingerprinting” to use Convolutional Neural Networks to conduct video fingerprinting attacks on network traffic transmitted over Tor in an open-world environment. It will expand upon previous research using a realistic vantage point to collect data utilizing Wi-Fi and traffic over the network interface. The end goal of this line of research is to increase the Department of Defense’s understanding of how much risk exists to the privacy of the users of the Tor network. While performance suffers compared to models trained on data sets collected under ideal conditions, we show that video fingerprinting is still viable in typical network conditions, including robustness to traffic congestion in all but the most extreme cases.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

1	Introduction	1
1.1	Introduction	1
1.2	Purpose and Scope.	2
1.3	Thesis Organization	2
2	Background and Related Work	5
2.1	The Need for Online Vigilance	5
2.2	The Onion Router	6
2.3	Website Fingerprinting	9
2.4	Video Fingerprinting	10
2.5	Machine Learning	11
2.6	Deep Learning Models	12
2.7	Realistic Conditions	13
3	Generalization across Local Network Conditions	15
3.1	Methodology	15
3.2	Test Design	17
3.3	Closed World Results.	17
3.4	Open-World Results	18
3.5	Enhanced Methods Results	20
3.6	Chapter Results	22
4	Effects of Local Network Congestion	25
4.1	Test Design	25
4.2	Data Set Collection	27
4.3	Results	28
4.4	20 Mbps/s Bandwidth Limit	31
4.5	Chapter Results	34

5 Conclusion	35
5.1 Limitations	35
5.2 Recommendation for Future Work	35
5.3 Final Conclusions	37
Appendix: Videos Used in the Congested Sets	39
List of References	41
Initial Distribution List	45

List of Figures

Figure 2.1	The messages exchanged between client and server in HTTPS.	6
Figure 2.2	The growth in Tor relays over the past 15 years.	7
Figure 2.3	Diagram of the Tor network.	9
Figure 3.1	Various Precision-Recall curve for the 32K open-world size.	19
Figure 3.2	PR Curve for the Baseline approach and the approach using Bayesian methods and Mixup.	22
Figure 4.1	Traffic over the interface at full saturation.	26
Figure 4.2	Diagram showing the setup of the experiment. The laptop hosting the web crawler is receiving traffic from both the video stream and the iPerf3 client.	27
Figure 4.3	Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the HTTPS data set.	28
Figure 4.4	Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the Tor data set.	30
Figure 4.5	20 Mbps/s Bandwidth Limit: Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the HTTPS data set.	32
Figure 4.6	20 Mbps/s Bandwidth Limit: Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the Tor data set.	33

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

Table 3.1	Tor accuracy results	18
Table 4.1	HTTP average accuracy results at 10 Mbps/s bandwidth limit . . .	29
Table 4.2	Tor average accuracy results at 10 Mbps/s bandwidth limit	30
Table 4.3	HTTPS average accuracy results at 20 Mbps/s bandwidth limit . .	32
Table 4.4	Tor average accuracy results at 20 Mbps/s bandwidth limit	34

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

CNN	Convolutional Neural Network
DASH	Dynamic Adaptive Streaming over HTTP
DOD	Department of Defense
FPR	False Positive Rate
HTB	Hierarchical Token Bucket
HTTP	Hypertext Transmission Protocol
HTTPS	Hypertext Transmission Protocol Secure
ISP	Internet Service Provider
NPS	Naval Postgraduate School
PCAP	Packet Capture
TCP	Transmission Control Protocol
TLS	Transport Layer Security
USN	U.S. Navy
WF	Website Fingerprinting

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1: Introduction

1.1 Introduction

As the internet has emerged as the leading and dominant method for information exchange, users have developed an appreciation for maintaining their privacy online despite confusion about the best actions to take to maintain that security, as confirmed by a recent Pew survey [1]. Internet users face various threats - some are accidental, like commercial entities that will mishandle private information, leading to costly data breaches, which reached an all-time high in 2023 [2]. Others are malicious, like authoritarian governments that censor internet content and flag the users attempting to view it or criminal organizations that try to steal sensitive information and sell it to the highest bidder. To address these fears, new markets have emerged to cater to those worried about maintaining their privacy, with roughly 50 percent of American consumers willing to pay eight dollars a month to maintain their privacy, according to a recent consumer study [3]. Listen to any podcast or watch a random YouTube video, and there is a good chance that there will be an ad for a sketchy VPN service offering to protect all web browsing but will typically over-promise or provide misleading information, according to a recent study [4]. So, what proven technologies do consumers have at their disposal?

One popular method to gain increased security for web browsing is to utilize the Onion Router network, commonly called the Tor network, which has an average of over a million users over the past year according to their metrics [5]. Using this network, users can bounce their traffic across several nodes located worldwide to prevent their activity from being directly traced back to them. This network encrypts traffic such that an eavesdropper could find the identity of the next and previous hop on the route but leave the identity of the origin and final destination hidden. It is often regarded as one of the best ways to evade censorship and protect online privacy.

Unfortunately, the Tor network has been shown to have its flaws. One vulnerability is analyzing collected traffic to determine what website is being visited despite lying behind

an encrypted connection. By focusing on the timing, direction, and packet size of network traffic, machine learning models can theoretically identify what sites the Tor user is browsing if the model has been trained on previous visits in a process called website fingerprinting. This process has expanded to identifying individual sub-pages and streaming videos.

The current state-of-the-art method utilizes a Convolutional Neural Network (CNN) to perform this fingerprinting classification task. We seek to provide insight into how well results from this model will generalize to realistic scenarios, with a large open world and data collected over residential wireless networks.

1.2 Purpose and Scope

Two experiments have been performed to determine how collecting data over residential Wi-Fi affects the accuracy of video fingerprinting. A different data set will be collected for each experiment, one under general Wi-Fi network conditions and another with the amount of network congestion varied over different collection runs.

We first collected 200 visits from a set of 55 Vimeo-hosted videos to train and test a deep-learning model. A second set of visits to 40,000 Vimeo-hosted videos will also be collected to form an open-world test set to determine the model's ability to differentiate between videos in the monitored and unmonitored sets. This set-up replicates the process used by Walsh et al. [6], and we will compare our accuracy results to theirs.

The second experiment will focus on collecting data to isolate the effects of a traffic-congested internet connection. We will collect a data set of visits to ten YouTube videos over several different congestion levels of internet traffic. The results from the machine learning model will be used to determine how performance degrades as the network interface becomes more congested, both in absolute terms and as a percentage of the bandwidth.

1.3 Thesis Organization

The rest of the thesis will cover the following. Chapter 2 will provide background on the importance of internet privacy and the Tor Network and the current progress of using machine learning for website and video fingerprinting. Chapter 3 will describe the process of collecting a data set over a home Wi-Fi and how the current standard model generalizes

to the data collected over residential networks. Chapter 4 will further explain how this data set performs under a congested interface. Finally, Chapter 5 will go over conclusions and recommendations for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2: Background and Related Work

2.1 The Need for Online Vigilance

As the general population spends more and more of their life on the internet, either actively or passively, users have developed a greater appreciation for online privacy, with 42 percent of Americans expressing that they are “very worried” about protecting their data online [1]. These fears are not unfounded as the general population faces threats to their privacy from totalitarian governments, criminals, and commercial actors, necessitating that some level of vigilance is prudent to maintain their privacy and protect their sensitive personal data. However, malicious actors might also have less innocent reasons for maintaining their anonymity over the internet, instead seeking to cover their tracks. According to the Council of Economic Advisers [7], cybercrime is a significant issue, costing the US economy between 57 billion and 109 billion dollars in 2016. Another suspicious corner of the internet is the Dark Net, a series of web pages unavailable on the public internet. According to the Congressional Research Service [8], it has emerged as an active marketplace for illicit drugs, document forgeries, and stolen personal information. To both sides of the law, navigating the internet using safe and secure methods has become essential.

This rise in internet traffic sensitivity has not escaped Washington policymakers’ notice. The White House’s 2023 National Cybersecurity Strategy [9] states that “the privacy of our data and communications” is fundamental for securing America’s economic prosperity. The corresponding implementation plan [10] outlines ways to protect consumer privacy, such as updating the National Privacy Research Strategy and embracing public-private partnerships. While these solutions may be effective on the national level, everyday internet users still need to look for more personal ways to protect their privacy.

One approach to improve internet privacy and security is the widespread adoption of the Hypertext Transmission Protocol Secure (HTTPS) as an improvement to the traditional Hypertext Transmission Protocol (HTTP). HTTP would send packets in plain text, making it easy for interlopers to harvest passwords in transit, creating paranoia about conducting

sensitive work over public internet access points. HTTPS improves on this protocol by using encryption to protect data, which works like this. First, before any of the actual relevant information is exchanged, the client and server will exchange messages that allow them to build an encrypted channel. Also exchanged will be the server’s digital certificate, which the client can use to check that the server is valid with the appropriate certificate authority. Once the encrypted channel has been built, the server and browser can exchange sensitive information without risk of interception. This process is illustrated in Figure 2.1.

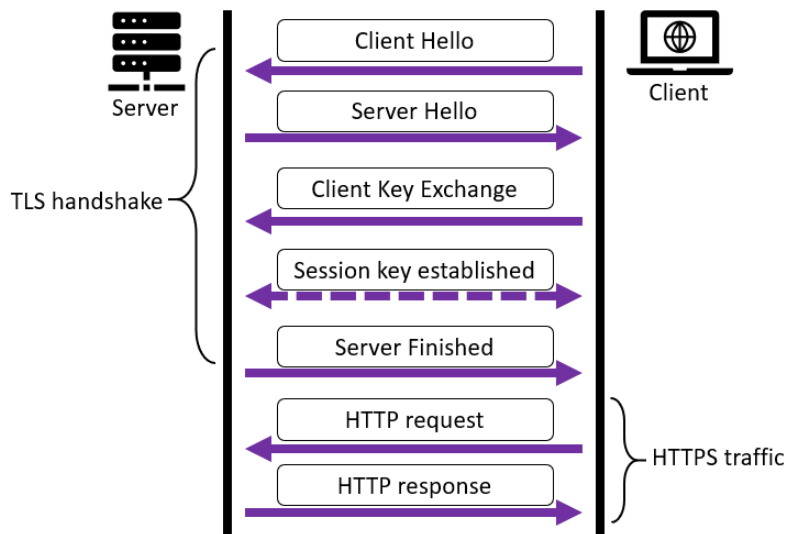


Figure 2.1. The messages exchanged between client and server in HTTPS.
Source: [11]

The rise of HTTPS has been successful in helping users become more comfortable with sensitive tasks like online banking and shopping. It has become the default protocol for 86.9 percent of all websites, according to a recent W3Techs report [12]. However, more advanced users can take advantage of more sophisticated encrypted networks to protect their online privacy.

2.2 The Onion Router

The Tor Network provides a robust solution for people seeking increased privacy in their internet browsing. The Tor Browser, maintained by the non-profit organization the Tor

Project [13], is a free and open-sourced software suite that establishes secure connections by bouncing web traffic over a network of trusted relays located all over the world. The article describing its design, “Tor: The Second-Generation Onion Router,” also online a focus on deployability, citing goals that Tor “must not place a heavy liability burden on operators” nor “be difficult or expensive to implement” [14].

Since its original release, the Tor Browser has provided several valuable use cases to those looking to add another layer of privacy to their internet browsing. These can include government personnel who connect to government websites from within hostile territories, journalists looking to protect their sources, or activists looking to avoid the watchful eyes of authoritarian governments. Another less savory group that utilizes Tor is cyber criminals, or people looking to disseminate extremist material and incite others to commit acts of violence.

The Tor routing process works by utilizing a method called onion routing, where web traffic is encrypted three times and then run through three relay nodes. These nodes come in three different flavors: the guard nodes, the middle nodes, and the exit nodes. The entry relay links the host running the Tor Browser into the Tor network. The middle node connects between the entry and exit nodes. The exit node finally routes the traffic between the Tor network and the destination. There are currently about 8,000 nodes being maintained, a number that has steadily increased over the past 15 years, as seen in Figure 2.2.

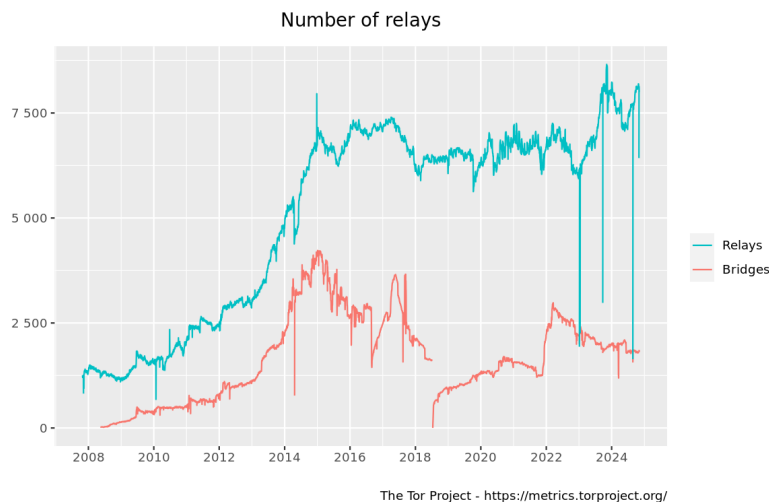


Figure 2.2. The growth in Tor relays over the past 15 years. Source: [15]

The encryption chain works like this: it starts with the Tor client encrypting the original data so that only the exit node can de-encrypt it. Then, the client adds another layer of encryption that the middle node can de-encrypt and, finally, a layer that the entry relay will strip off. Then, the nodes will remove the encryption as the packet travels through the circuit. This prevents eavesdropping near the connection's endpoint (such as by an Internet Service Provider (ISP)) because they only see encrypted traffic between the client and the entry node. At the far end, the website host will see the unencrypted traffic but can only trace it as far back as the exit relay node.

To build this circuit, the client needs access to a list of all the available relays, and the Tor Project facilitates this process by maintaining a list of relays hosted worldwide that are used to build the connections Tor uses. Around the world, ten Tor servers called directory authorities maintain the list of all known relays. Hourly, these directories will agree on a list of relays and publicly publish it as the consensus file [16].

Additionally, there are special nodes called bridges. Since the IP addresses of relays are publicly available, a nation-state that wanted to block its citizens from using Tor could simply block traffic from the entire list of relays. Bridges are special entry nodes that are not publicly available and are maintained by their own special directory authority. They can be used in place of traditional entry relays.

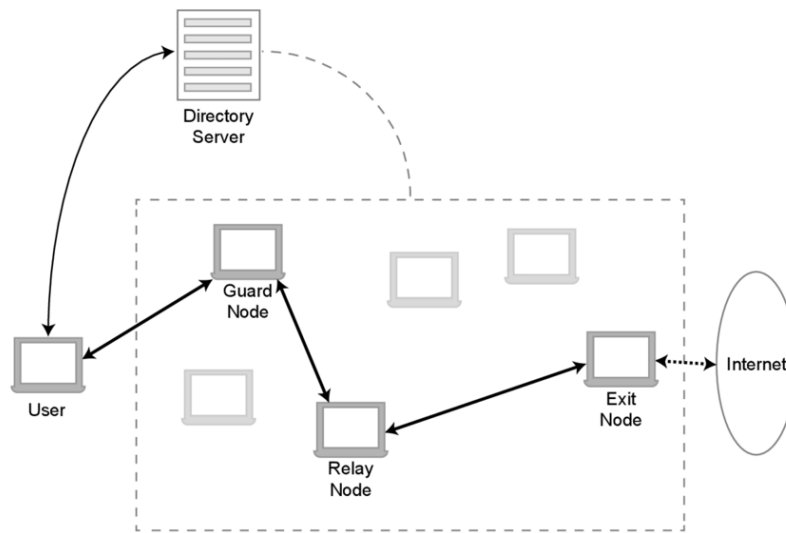


Figure 2.3. The typical path of a Tor Connection, showing all three relays and the directory. Source: [17]

The main vulnerability of the Tor Network is a theoretical attack where a well-resourced adversary can control a large percentage of the Tor relays, called a Sybil Attack. It is described by Winter et al. [18] as “an attacker controls many virtual identities to obtain disproportionately large influence in a network.” If the adversary controls all the nodes along a circuit, it can unmask the traffic of a user who believes that Tor is defending him directly. However, our research will look at a subtler attack, which looks at the characteristics of the traffic itself to degrade the privacy provided by Tor.

2.3 Website Fingerprinting

Because of these illicit activities on the Tor network, attacks on the network’s confidentiality have emerged. One is “website fingerprinting,” where analyzing the direction, size, and timing of the encrypted traffic trail can inadvertently reveal information about the content going through the encrypted connection. Hintz [19] pioneered this term in 2001 and explored if it is possible to identify websites based on encrypted traffic patterns. Hintz’s research showed that despite encryption, distinct traffic features can still be detected, and an observer can make educated guesses about the website visited based on the traffic signature. Since then, website fingerprinting has become a continuing field of study in network security, with

various attack techniques that try to refine and improve the accuracy of identifying websites and sub-pages based on traffic alone.

More research has been done to build upon this, including how packet sizes, arrival times, and packet directionality contribute to website identification, such as Cai et al. [20]. This has implications for users of Tor and anyone interested in securing their privacy against such analysis.

2.4 Video Fingerprinting

According to the 2024 Sandvine report [21], video streaming makes up 40 percent of all internet traffic, so it is no surprise that the concepts behind WF would also be applied to video. The broad idea is the same: using timing, size, and direction of internet traffic to identify which video a user is streaming, even over an encrypted connection. Video fingerprinting benefits over website fingerprinting because video files are relatively static on hosting platforms, compared to the individual components of a website that may change frequently and slightly, which can break pre-trained models.

Video fingerprinting presents a new set of challenges and opportunities for privacy. The critical thing to note is that the video hosting platform segments the video into smaller chunks (usually a couple of seconds each) before sending it to the user. This segmentation allows for better bandwidth usage and a smoother user experience but also creates patterns in the traffic. However, since each segment is requested separately, each with its own timing and size, a video fingerprinting model can use these features to distinguish one video from another.

Specifically, a method that adds fingerprintable features to a video stream is video compression. Compression is accomplished with video codecs like the popular H.264 codec [22], algorithms that shrink videos by recognizing patterns and discarding unnecessary information. These codecs can compress simpler, static videos more than more dynamic videos, meaning that the same video will show a similar amount of compression when streamed. Often paired with compression is variable bitrate encoding, which, as opposed to constant bitrate, allows the bitrate to fluctuate depending on the quality of the current video segment. So, it can scale up the bitrate when transferring detailed segments with less compression to

preserve quality and scale down for simple compressed segments, which is a feature that is repeatable when the same video is steamed multiple times.

Adaptive bitrate streaming protocols, the most widely used one being Dynamic Adaptive Streaming over HTTP (DASH) [23], also need to be considered in this context as a feature that makes video fingerprinting more difficult. DASH dynamically adjusts video quality based on the user's available bandwidth so a video can be played at a higher resolution, such as 1080p on fast network connections, but can shift down to a low resolution if latency occurs. This behavior allows for better bandwidth usage and a smoother user experience. It also makes the traffic trace of the playback of the same video different if it was streamed under different network conditions, making it harder to fingerprint.

2.5 Machine Learning

Website fingerprinting and video fingerprinting are examples of classification problems that machine learning can be used to solve. In machine learning, input is typically broken down into several features, such as pixels in an image or statistics about an object. Then, the model takes these feature inputs and presents a prediction, either a continuous number in the regression case or a class in the classification case. We can measure how well a model makes predictions based on an objective loss function, which measures the distance between the predictions and the known labels. Typically, mean squared error is used for regression tasks, and cross-entropy is used for classification. This loss function is how the model is trained in the first place. A learning algorithm, such as gradient descent, is used to slightly adjust model parameters to minimize the model's loss function. For website or video fingerprinting, we train classifier models to recognize the distinct features of a website or video on known traffic and then make predictions about traffic traces of unknown traffic instances.

Since the internet contains a growing and practically endless amount of videos, it is unrealistic to assume that the videos a user watches are from the set of videos an eavesdropper might be interested in. Nor would it be practical for an adversary to train a model on the near-infinite number of videos a user could watch. For this reason, we can distinguish between open-world and closed-world scenarios in machine learning. In a closed-world scenario, every example will be from the classes on which a machine-learning model has been trained.

In an open-world scenario, we will have examples not from the set of monitored classes, and it is the model’s task to separate the examples we care about and everything else. In this case, we will label these unmonitored instances as their own separate class and task our model with differentiating between the monitored and unmonitored instances.

2.6 Deep Learning Models

The technology that has emerged as the most important for implementing video fingerprinting is deep learning models, which have proven highly effective for analyzing and recognizing patterns within complex datasets. Deep learning employs neural networks, which consist of interconnected layers of nodes or “neurons” organized into distinct layers: an input layer that receives the features of the data, multiple hidden layers that process the data through weighted connections, and an output layer that provides the final classification or prediction.

One particularly powerful form of deep learning for sequence recognition and classification is the CNN. CNNs have gained prominence due to their ability to learn spatial hierarchies in data and have, according to Géron, “managed to achieve superhuman performance on complex visual tasks” [24]. These models differ from traditional neural networks by incorporating convolutional and pooling layers.

Convolutional layers consist of filters scanning the input data for feature detection and highlighting within localized regions. This process effectively encompasses the unique spatial dependencies in the data. The spatial dimensions are reduced through additional pooling layers that enable us to reduce the number of parameters, improve computational efficiency, and help prevent over-fitting by retaining only the most relevant features.

This was the approach followed by representative studies such as those by Schuster et al. [25], and Sirinam et al. [26], which demonstrated the potency of CNNs to achieve robust video and website fingerprinting, respectively. Their work introduced a technique referred to as “Deep Fingerprinting,” a method using deep CNN architectures to learn distinctive patterns in video traffic. Subsequently, Walsh et al. [6] adopted the network architecture followed by Sirinam et al., using a sequence of one-dimensional convolutional layers that are successively utilized for feature extraction. The other components of this architecture are batch normalization layers, which help stabilize training; pooling layers, which down-

sample the features; and dropout layers, which prevent over-fitting by randomly setting the weight of neurons to zero at training time.

This sequence is followed by fully connected layers integrating the learned features across all previous layers. Finally, the softmax layer produces the output vector of prediction probabilities over the potential classifications. This architecture enables CNNs to capture the intricate patterns in traffic trace data required for precise fingerprinting and identification.

2.7 Realistic Conditions

An important aspect of video fingerprinting research involves examining how local network conditions influence model performance. Much of the prior work in this field has primarily focused on idealized network conditions. For example, Walsh et al. [6] had their web crawlers operating in controlled virtual machine environments hosted in a data center. This excludes the latency, packet loss, and congestion most users experience when streaming a video from their personal devices. It even ignores some geographical considerations, as data centers are often located close to internet exchange points for faster access, so a typical user must take a longer path to reach the servers of the desired content delivery network, according to Holt [27]. Some research has looked at more realistic network conditions, such as when Zhang et al. [28] explored the impact of a “poor network environment” but only investigated performance under a fixed, predefined set of network limitations. Similarly, Carlson et al. [29] employed artificial techniques to simulate variable bandwidth conditions, offering a controlled but arguably less realistic depiction of real-world network behavior.

In contrast, our research aims to fill this gap by studying video streaming data captured from actual hardware operating over commercial internet connections. This approach allows us to account for the variability and unpredictability inherent in real-world settings, such as latency fluctuations and packet loss, which characterize the everyday network environments of thousands of households. By leveraging this methodology, we aim to provide a more robust understanding of how network conditions influence the reliability and accuracy of video fingerprinting models.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:

Generalization across Local Network Conditions

3.1 Methodology

This chapter will describe collecting a monitored and unmonitored data set over a residential wireless connection and compare the results to the baseline approach and improvements proposed by Walsh et al. [30]. This will give us a direct comparison of the same methods used with a data set collected under ideal conditions versus one collected under more typical conditions.

3.1.1 Data Collection

For this experiment, we will collect two data sets, one forming the monitored set and the other the more extensive unmonitored set. The monitored set will consist of a list of the same 60 Vimeo videos used by Walsh et al. [30], with at least a hundred visits to each. In practice, the monitored set has only 55 videos because five of the original videos have been removed from Vimeo.

The second data set will be our unmonitored set, comprising 40,000 visits to videos hosted on the Vimeo site. Each visit will be to a unique URL and will all be labeled as the same class when training our model. With these two sets, we can simulate an open-world scenario to see how well a model can differentiate between a narrow list of monitored videos and all the other possible videos a user could stream from the same platform.

3.1.2 Lab Environment

Each web crawler will be hosted on five consumer-grade laptops. Each laptop will have the Ubuntu 22.04 operating system installed and at least 2GB of RAM.

Then, each laptop will be connected to the same 5 GHz wireless access point. A Rogers Xfinity Gateway modem (XB7) [31] provides the wireless access point with an advertised bandwidth of 200 Mbit/s. This router also services other consumer devices to simulate

nominal network traffic. All model training was done on the Hamming supercomputer cluster hosted at the Naval Postgraduate School (NPS).

3.1.3 Web Crawler

Collecting a large data set of packet captures for each visit to a video streaming site is a tedious process and would likely represent the major bottleneck to a potential adversary. To automate this process, we will rely on the Tor-browser-crawler project [32] to perform the web scraping necessary to train a machine-learning model.

The Tor Browser Crawler is a program written in Python that uses Selenium WebDriver to autonomously visit a list of video URLs provided by the user. Using the WebDriver allows us to handle the usual behavior of the video hosting site, such as skipping ads or cookie pop-ups as a real user would. For each visit, the program uses tcpdump to produce a PCAP file that contains all the traffic sent from and to the host. We only retain the headers of each packet to cut down on the size of the raw data set. The crawler also takes a screenshot at the beginning and end of each capture, allowing us to check the data to verify a successful capture manually.

3.1.4 Data Representation

Once we have the collection of PCAP files that result from each crawler visit, we need to convert them to a numerical representation. We can then use this representation as the input vector for our prediction model.

Previous research has taken several approaches to turn the PCAP data into a numerical representation [25], [26], [33], [34]. Walsh et al. [6] have shown that the Schuster method with 1/8 second time steps is the most accurate for Tor traffic and still very effective for HTTPS traffic. The first 240 seconds of the video will be collected, resulting in an input vector that is 1920 features long. This is the representation that we will be using throughout this research.

Once we have decided on the representation, we will parse the Packet Capture (PCAP) file with the dpkt package. The distant end of the connection is then checked against a list of

Tor relay addresses taken from the consensus files [16] to ensure they are part of the Tor traffic stream.

3.2 Test Design

Generally, we will replicate the process Walsh et al. used in their paper “Exploring the Capabilities and Limitations of Video Stream Fingerprinting” [6]. To do so, we will collect training, test, and validation sets that contain a mix of videos from the monitored and unmonitored sets. The test set will include 15 percent of the 55 monitored video instances with an addition of 32,000 instances of unmonitored videos to simulate a large open-world environment.

After building out the test set, another 70 percent of the monitored captures and the leftover unmonitored captures will be combined to form the training set. The remaining 15 percent of the monitored videos will be chosen as the validation set. The model, again using the Sirinam architecture [26], will be trained for 120 epochs with early stopping based on the categorical cross-entropy loss of the validation set. For the model, we will use the hyperparameters (strides, activation functions, batch size, etc.) that Walsh et al. [6] found to be optimal for this data representation and protocol.

3.3 Closed World Results

Before looking at the open-world case, we will start by using only the monitored set. This will let us determine how well the model can differentiate between the 55 videos in the monitored set. Comparing our results to Walsh et al. [6], the model shows a slight degradation when used on our data set. Over ten trials the model on our data set has an average accuracy of 0.8167 with a standard deviation of 0.0111. So, as expected, the model performs slightly better on the data taken from a virtual machine environment but still performs well over the testing set.

To overcome this degradation, we can try using more training instances for our model. By doubling the number of captures to 200 for each of the classes available for the training and validation tests, we can improve our results over ten trials to an average of 0.8756 with a standard deviation of 0.0081. The results of all tests can be seen in Table 3.1.

Table 3.1. Tor accuracy results

	Tor
Walsh et al. [6]	0.9697
Our Work (100 Samples)	0.8167
Our Work (200 Samples)	0.8765

3.4 Open-World Results

Moving to the open-world case, we will consider how well a model can differentiate between the videos that are in the monitored set and all videos that are possible to stream represented by the unmonitored set. Since we are dealing with such unbalanced data sets between the monitored and unmonitored captures, accuracy is an inappropriate measurement for this experiment. Instead, we will be concerned with the precision and recall of our model. For a general sense of performance in this case, we can look at the Precision-Recall curves of our model, which were tested using the 32K world size, as seen in Figure 3.1.

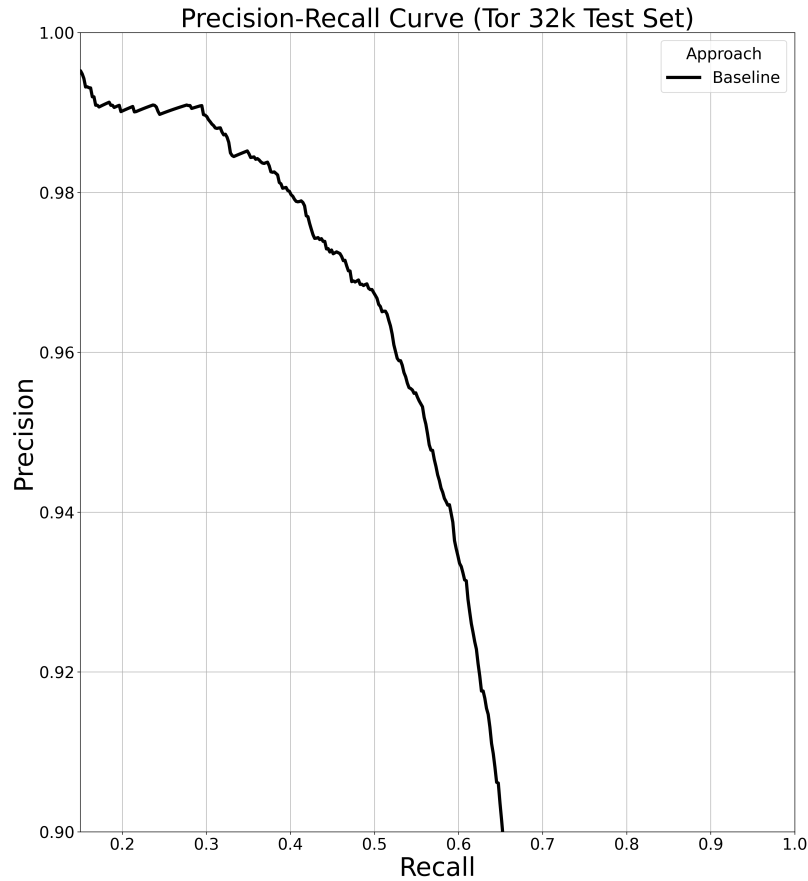


Figure 3.1. Various Precision-Recall curve for the 32K open-world size.

The Precision-Recall curve varies the threshold that we use to make the predictions. The threshold represents how high the prediction probability needs to be to be positively picked as that class. For example, if the highest prediction probability for a class was 0.75 but the threshold was 0.8, it would be labeled as an example of an unmonitored class. Following the PR Curve, we can see the trade-off between precision and recall as we go from high thresholds on the left to low thresholds on the right.

We can look at the area under the PR Curve (PR-AUC) to understand how well the model performs. Walsh [30] found that when performing a 32K open-world test with his model trained on ideal conditions, it achieved a PR-AUC of 0.8774. In our test, we achieved an average PR-AUC of 0.8211. So again, we find a performance delta compared to the model trained with ideal data. In fact, Walsh gets a score of 0.8408 for the 64K world size in his study, so for this case, using traffic collected over Wi-Fi is at least equivalent to doubling the world size in terms of lost performance.

Although the precision-recall numbers are difficult to compare directly between world sizes, one metric that can be used between world sizes is the False Positive Rate (FPR). Walsh et al. [6] showed that as the size of the open world increases, the FPR converges because each prediction on a member of the unmonitored class “depends only on the parameters learned through training and that instance’s relationship to the training data. The prediction is independent of all other unmonitored instances during testing, and the outcome is binary - true negative or false positive - so each encounter with an unmonitored instance is a Bernoulli trial.” Moreover, while this FPR will fluctuate at lower world sizes, it will eventually converge due to the Law of Large Numbers to a constant value. Using a threshold that produces a recall of 0.5 on the validation set, their research found that the FPR converges to approximately 0.000016. Looking at the performance of our model, when operating at the same threshold, we get a FPR of 0.000184. While this could result from the value not quite converging at a smaller world size, it also is another data point showing worse performance on our data set.

3.5 Enhanced Methods Results

The Sirinam model, while state-of-the-art at the time, presents opportunities for enhancement through modern techniques. Walsh et al. [30] demonstrated that the performance of this model can be improved using Bayesian methods and Mixup data augmentation on their data set. Building on this foundation, we aim to investigate the potential of these approaches with our data set.

Bayesian methods introduce a probabilistic framework by treating model parameters as random variables drawn from a distribution, allowing for uncertainty estimation and reg-

ularization. Specifically, we adopt the Spike-and-Slab method [35] and Concrete Dropout [36] to update the Sirinam model.

In addition to Bayesian methods, we incorporate Mixup, a data augmentation technique proposed by Zhang et al. [37]. Mixup generates synthetic training examples through linear interpolating pairs of input data and their corresponding labels, smoothing decision boundaries and improving model generalization. This augmentation effectively reduces overfitting and enhances robustness near the decision boundary.

Incorporating these methods into our approach yields mixed outcomes. While the augmented model outperforms the baseline at higher threshold values, it exhibits degraded performance at lower thresholds. Overall, the baseline approach performs better with an average PR-AUC of 0.8211, while the enhanced methods score 0.7705 on average. These results highlight the nuanced effects of integrating Bayesian techniques and Mixup into the Sirinam model, suggesting the need for further investigation into how these methods interact with varying data set characteristics and decision thresholds, including considering the use of new hyperparameters for these particular models. The Precision-Recall Curves for the baseline and enhanced approach can be seen in Figure 3.2.

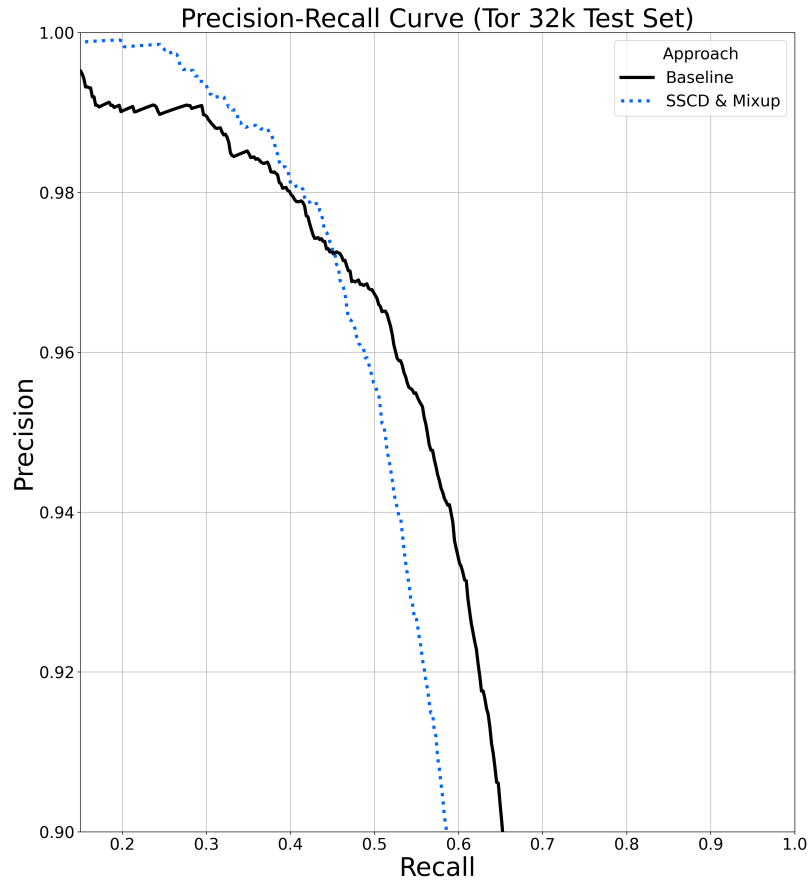


Figure 3.2. PR Curve for the Baseline approach and the approach using Bayesian methods and Mixup.

3.6 Chapter Results

In all three of the approaches we perform in this chapter, closed world, open world, and open world with enhancements, we are able to train models that can be used to identify the video behind a traffic trace but are unable to match the model performance that Walsh et al. were able to achieve — suggesting that some combination of the increased latency, network

congestion, packet loss, or geographical factors are enough to reduce the performance of models trained on typical residential traffic versus those that use traffic collected over ideal conditions such as a wired connection near the internet backbone.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4: Effects of Local Network Congestion

In order to help explain the discrepancy between the data set collected in a virtual environment and over a Wi-Fi network, in this chapter, we will try to isolate one of the possible reasons - congestion on the network. While in our previous example we could theoretically have congestion on the router interface, here we will move the congestion to the host network interface to explore the effects of congestion in general.

4.1 Test Design

4.1.1 Interface Bandwidth Limit

To replicate the conditions of a typical home internet connection, we utilize the Linux Traffic Control (tc) utility [38] to shape and manage network traffic on our web crawlers' computers. The tc utility provides a powerful framework for traffic shaping by enabling the implementation of the Hierarchical Token Bucket (HTB) queuing discipline on a host's network interface. This queuing mechanism is configured with specific classes that enforce defined bandwidth limits, effectively controlling the flow of outgoing network traffic. A filter is applied to route traffic through the designated class to ensure that all network traffic adheres to these restrictions.

In this study, we configure the HTB to impose a bandwidth limit of 10 megabits per second (Mbps/s). This value was chosen to provide a manageable bandwidth to fill with traffic and also, according to the Federal Communication Commission, provides the lower end of what a household would need in order to maintain "basic service" [39].

4.1.2 Induced Traffic

To control the congestion at the network interface, we will need to generate traffic to ensure the desired bandwidth usage. We will use the iPerf3 utility [40] to run a user-defined amount of traffic between two computers. First, we set aside one laptop to act as the dedicated iPerf3 client to indefinitely send a set level of traffic to the rest of the web crawler hosts. We will

send the appropriate amount of traffic to achieve congestion levels of 25, 50, and 100 percent of the bandwidth limit; for example, we would send 5 Mbps/s of traffic to a laptop with a 10 Mbps/s bandwidth to simulate 50 percent congestion. The bandwidth utilization at 100 percent can be seen in Figure 4.1.

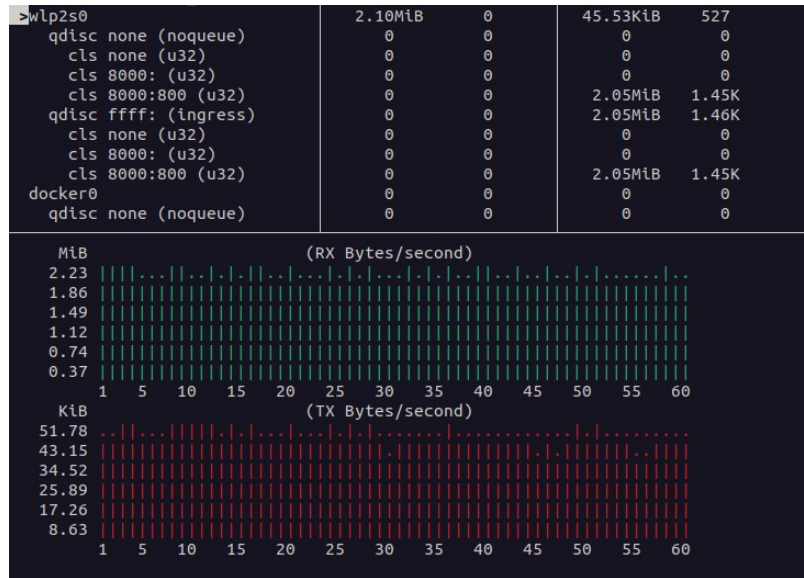


Figure 4.1. Traffic over the interface at full saturation.

4.1.3 Methodology

The experiment design will use the same lab environment as the previous chapter. Web crawlers will visit the ten monitored YouTube videos a hundred times each, using the previously discussed protocols (Tor and HTTPS). The full list of videos is listed in Appendix 1, and each video will be visited 100 times for each level of iPerf3 traffic. The laptops hosting the web crawlers will have limited bandwidth and will also transfer a constant amount of traffic using iPerf3. On average, looking at the PCAP of the crawlers, 52 percent of the total traffic during the capture was made up of iPerf3 traffic at 25 percent congestion. On average, this ramps up to 88 percent of total traffic from the iPerf3 server at full congestion. While these packets are stripped out when the PCAP files are transformed into the vector representation, the disruptions to the timing of the rest of the packets will remain and could potentially reduce the effectiveness of machine learning models that rely on those features.

Using the standard Sirinam CNN model, as in the previous chapter, we will train a 10-way classifier on the data set collected without any iPerf3 traffic. We will then record the model’s accuracy on test sets collected at increasing congestion levels on the network interface in closed-world conditions.

4.2 Data Set Collection

For this data set, we will use YouTube due to emergent issues with CAPTCHAs appearing in Vimeo videos, which is further explained in Chapter 5. Our data set will be acquired using the same process as the previous chapter and will consist of ten YouTube videos. We will then vary the congestion across the network interface by limiting the bandwidth and using iPerf3 to flood the interface with the appropriate amount of traffic, as shown in Figure 4.2. We will then collect a hundred instances of each video at each different monitored congestion level.

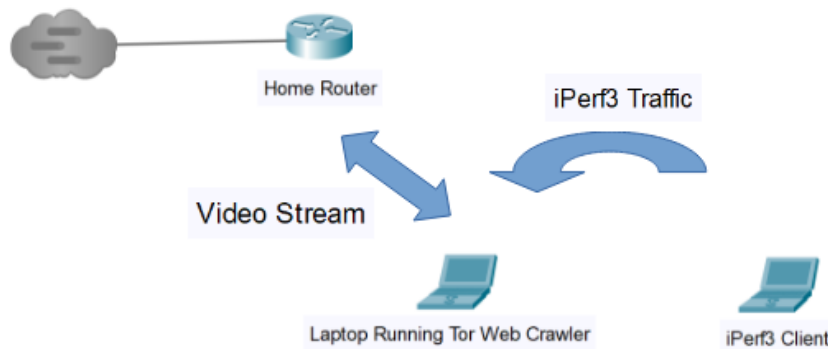


Figure 4.2. Diagram showing the setup of the experiment. The laptop hosting the web crawler is receiving traffic from both the video stream and the iPerf3 client.

Similar to the work done in Chapter 3, the data sets will be split 70/15/15 between training, test, and validation sets. Since the set will be balanced between the ten classes, we will use accuracy as the primary evaluation metric. All results are the average of ten trials with the model retrained for each iteration.

4.3 Results

4.3.1 HTTPS Data Set

We first trained the model on a training set that was captured using the HTTPS protocol and with the iPerf3 traffic turned off. Then, that model uses test sets at all the captured congestion levels to compare its performance. As expected, the model performs exceptionally well on the zero percent congestion test set, with a modest degradation in performance over the other test sets. Results can be seen in Figure 4.3 and the middle column of Table 4.1.

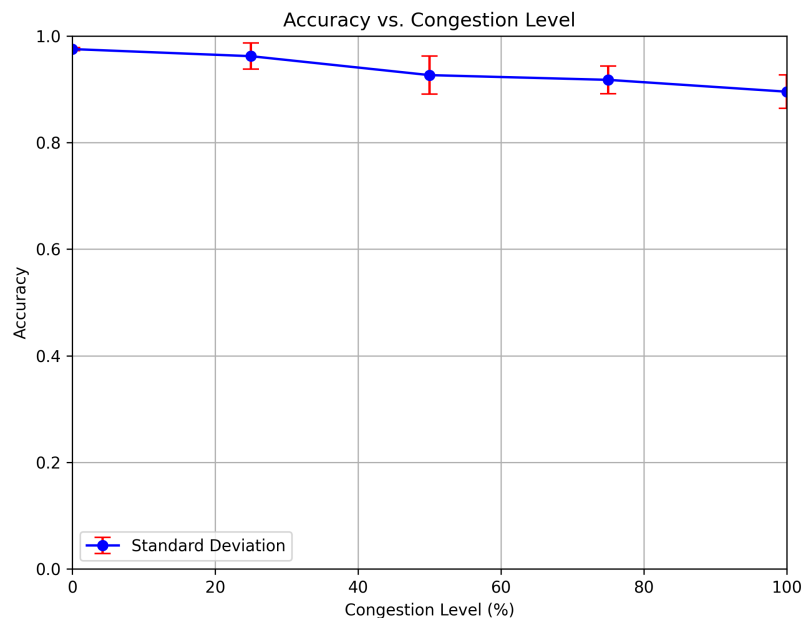


Figure 4.3. Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the HTTPS data set.

If we train a model on data at a particular congestion level and then test it only on traffic from the same congestion level, we see even that modest tail-off disappear. The model consistently achieves an accuracy of approximately 97 percent as long as it is trained and tested on the same level. Overall, the amount of network traffic on the interface has very little

effect on the performance of the machine-learning model when using the HTTPS protocol. Results can be seen in the last column of Table 4.1.

Table 4.1. HTTP average accuracy results at 10 Mbps/s bandwidth limit

Congestion Lvl	Accuracy of the 0% Congestion Model	Accuracy at Own Congestion Lvl
0 Percent	0.9756	0.9756
25 Percent	0.9622	0.9911
50 Percent	0.9267	0.9756
75 Percent	0.9178	0.9600
100 Percent	0.8956	0.9467

4.3.2 Tor Data Set

We extended our analysis to the data set collected over the Tor network. Initially, we trained a model using data collected under no additional iPerf3 traffic conditions and evaluated its performance on data sets subjected to progressively increasing congestion levels. The results indicate a modest decline in accuracy as congestion increases, with a pronounced drop at 100 percent congestion, where the average accuracy plummets to 0.2289. These findings are illustrated in Figure 4.5 and the middle column of Table 4.2, which shows the model’s accuracy across varying network congestion levels.

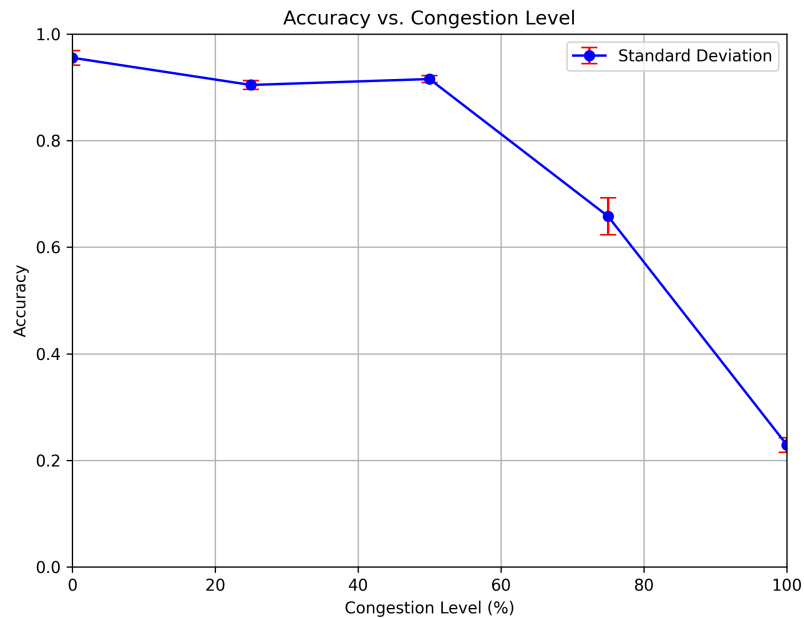


Figure 4.4. Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the Tor data set.

To further assess the model’s robustness, we trained and tested it on data sets with matching congestion levels. Under this condition, the model demonstrated consistent performance across the first three congestion levels. However, as with the cross-congestion analysis, a sharp decline in accuracy was observed at 100 percent congestion, with accuracy dropping to 0.4778. These results suggest that the model’s performance remains stable under moderate congestion but becomes significantly impacted under severe congestion. The last column of Table 4.2 provides a detailed comparison of these outcomes.

Table 4.2. Tor average accuracy results at 10 Mbps/s bandwidth limit

Congestion Lvl	Accuracy of the 0% Congestion Model	Accuracy at Own Congestion Lvl
0 Percent	0.9556	0.9556
25 Percent	0.9044	0.9444
50 Percent	0.9156	0.9267
75 Percent	0.6578	0.8648
100 Percent	0.2289	0.4778

4.4 20 Mbps/s Bandwidth Limit

The results of our experiment conducted with a 10 Mbps/s bandwidth limit leaves us with an important question: Does the observed degradation in model performance stem from the fact that iPerf3 traffic consumes 50 percent of the available bandwidth, or is it attributable to the absolute reduction in free bandwidth, leaving only 5 Mbps of overhead? This distinction is critical to understanding whether the model’s performance is primarily influenced by the relative percentage of unused bandwidth or the absolute amount of residual bandwidth. We plan to rerun the experiment under modified conditions to address this question. Specifically, we will increase the bandwidth limit to 20 Mbps/s while maintaining the same relative congestion levels, thereby enabling us to disentangle the effects of relative versus absolute bandwidth availability on model performance.

4.4.1 HTTPS Data Set

Unsurprisingly, the model trained on HTTPS traffic with zero iPerf3 traffic performs well across all the rest of the congestion levels again, even at the congested end of the bandwidth limit. Overall, the model maintains an accuracy between 0.9289 and 0.8511 for all the test levels. Results can be seen in Figure 4.6 and the middle column of Table 4.3.

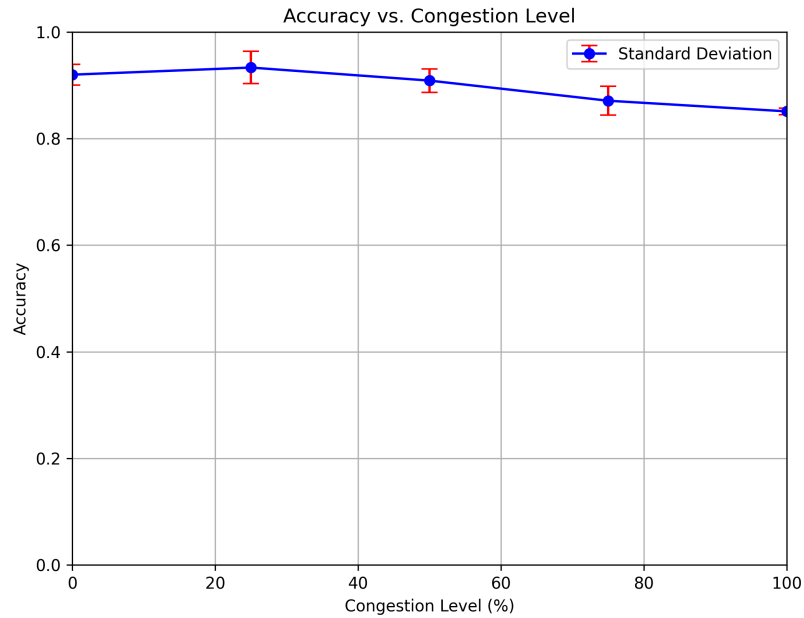


Figure 4.5. 20 Mbps/s Bandwidth Limit: Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the HTTPS data set.

If we alter our approach so that the model is only trained and tested on the same congestion level, we will find that the model performs well again. Overall, the model maintains an accuracy between 0.9644 and 0.9193 for all the test levels. Overall, there seems to be no combination of bandwidth traffic that would cause the HTTPS model to degrade in performance under these conditions, as seen in the last column in Table 4.3.

Table 4.3. HTTPS average accuracy results at 20 Mbps/s bandwidth limit

Congestion Lvl	Accuracy of the 0% Congestion Model	Accuracy at Own Congestion Lvl
0 Percent	0.9193	0.9193
25 Percent	0.9289	0.9400
50 Percent	0.9089	0.9644
75 Percent	0.8711	0.9267
100 Percent	0.8511	0.9571

4.4.2 Tor Data Set

The model trained on the Tor data set at the 20 Mbps/s bandwidth limit shows similar performance to those at the 10 Mbps/s limit, with good performance at the zero iPerf3 traffic level followed by a modest drop-off and then a dramatic dip at the 100 percent congestion level to 0.3533. Again, we find that Tor traffic suffers from accuracy degradation as it approaches the maximum congestion level.

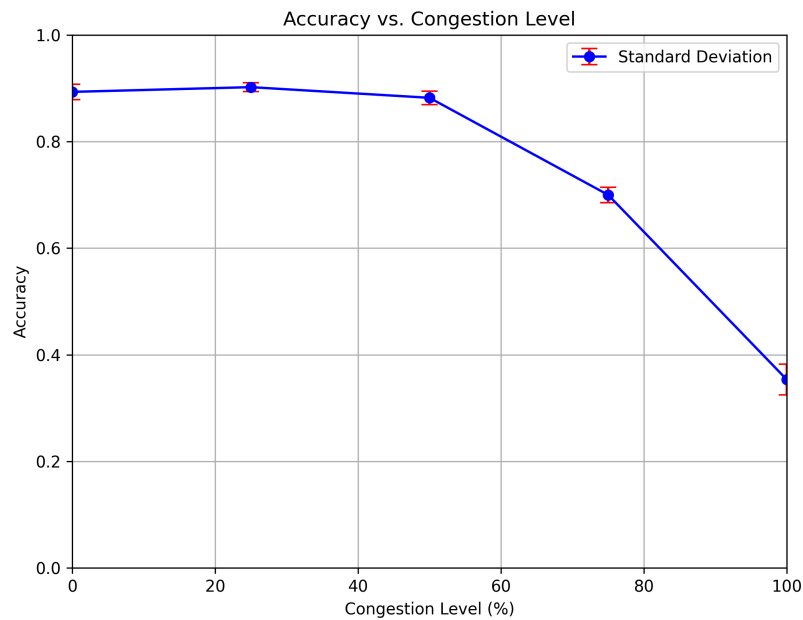


Figure 4.6. 20 Mbps/s Bandwidth Limit: Average accuracy of the model trained at the 0% congestion level and tested over different congestion levels on the Tor data set.

If we again alter our approach so that the model is only trained and tested on the same congestion level, we will find that the model performs well. Overall, the model maintains an accuracy between 0.9356 and 0.7311 for all the test levels. This shows that the model's performance improves if it is only trained on traffic from the same relative congestion level on which it will make predictions. Results can be seen in the last column of Table 4.4.

Table 4.4. Tor average accuracy results at 20 Mbps/s bandwidth limit

Congestion Lvl	Accuracy of the 0% Congestion Model	Accuracy at Own Congestion Lvl
0 Percent	0.8933	0.8933
25 Percent	0.9022	0.9156
50 Percent	0.8822	0.9356
75 Percent	0.7000	0.8422
100 Percent	0.3533	0.7311

4.5 Chapter Results

We can identify trends after testing HTTPS and Tor data sets over our congestion levels. HTTPS traffic does not seem very sensitive to congestion, as accuracy stays consistent through varying congestion levels. Tor traffic, on the other hand, does show some sensitivity, particularly at the upper limit. The accuracy results stay consistent throughout the 0/25/50 percent congestion levels on Tor traffic, with a noticeable dip at 75 percent and a sharp drop at 100 percent across both bandwidths. However, since practical situations where the majority of the bandwidth will be constantly used are very rare, likely congestion does not form a large part of our lesser model performance in Chapter 3 or a real-life scenario, especially since the models showed improved performance when they were trained on the same congestion level as they will be making predictions for, even at the higher bandwidth utilization.

As we speculate on the potential reasons behind the observed results, it is helpful to consider the effective bandwidth of traffic using the Tor network. Tor routes its traffic through a series of three relays, and the relay with the lowest available bandwidth in the path ultimately becomes the bottleneck, limiting the overall throughput of the network. According to the Tor network’s specifications, the minimum required bandwidth for each relay is 10 Mbps/s [41], which aligns with the bandwidth levels used in our experiment. Given this, it is plausible to hypothesize that additional network noise can easily overwhelm the relatively lower effective bandwidth available for Tor traffic. This congestion could disrupt the timing characteristics critical to the operation of prediction models, thereby affecting their accuracy. This disruption could lead to the degraded performance that we observed of the model’s performance on Tor traffic at higher congestion levels when compared to the HTTPS traffic.

CHAPTER 5: Conclusion

5.1 Limitations

Many of the limitations of this work come from the difficulty of collecting large video capture data sets. During the roughly seven months that data was collected for this project, changes had to be made to the setup. The first problem encountered was with Vimeo when the site renamed the play button element, which only required a minor rewrite to the web crawler. Later, further issues were encountered with Vimeo, prompting the web crawler to complete CAPTCHAs before beginning the video playback, an issue that led to changing the video hosting platform to YouTube for the congestion investigation. Even using YouTube, the crawler was prompted for a log-in of about 60 percent of visits, leading to Tor crawls taking more than twice as long to acquire the required number of PCAP files. Although an adversary with more resources and deep pockets could collect data in shorter time frames, those worried about their privacy can take solace in knowing that unmasking their traffic would require an actively maintained system to capture needed training data.

Due to time limitations, we only had Tor-collected data for our open-world experiment; seeing if HTTPS-collected data also generalized over Wi-Fi would be another interesting research question. Also, we were limited to enough monitored captures to form an open world set of 32,000, not the 64,000 world size of the most recent research in the area [6]. The same is true of the monitored set; after seeing improved performance from doubling the input to 200 examples, with more time, it would be interesting to see if these performance gains continue with more training examples and to what accuracy it converges to. Overall, with more time to collect data and more reliable methods, we could have better replicated and improved on previous work.

5.2 Recommendation for Future Work

In this work, we investigated the generalizability of video fingerprinting methods using data collected over a residential Wi-Fi network. Our results showed that video fingerprinting

is feasible under these conditions; however, the performance of models trained on traffic collected in a home environment needed to match the accuracy achieved by models trained on traffic gathered under idealized conditions, such as in virtualized or controlled network environments. To investigate this observed discrepancy, we found one possible cause related to the extraneous network traffic on the residential network interface, which would not occur in a VM setup, is additional traffic that introduces noise into the collected data. While this explanation seems plausible, further investigation is needed to evaluate other contributing factors systematically. The isolation and testing of other sources of variation, such as the effects of different network protocols or interference patterns in dynamic home environments, would help narrow down the cause of the reduced performance.

Another avenue for improvement is to address the geographic disparities between the site where data collection was performed and the data center's location hosting the virtual machines. All our network captures were done in Monterey, California. This geographic separation introduced variability in routing paths, which altered the timing characteristics of packet delivery. Since timing information is a critical feature for video fingerprinting models, discrepancies in packet timing likely affected overall model performance. This variability could be reduced by relocating the geographic vantage point closer to the data center hosting the VMs or by mimicking the routing paths more closely, allowing the model to perform better. This, therefore, emphasizes the need for careful consideration of geographic proximity and differences in routing paths in designing experiments with network traffic analysis.

Future work should also include the impact of capturing network traffic at different points in the connection path. While all the traffic in this study was collected at the endpoint host, an adversarial actor might intercept traffic upstream, such as at router hops or intermediary network nodes. A relevant research question is whether a model trained on traffic collected at the host can accurately predict video fingerprints when trained or tested on data collected at these upstream locations. This could provide critical insights into the robustness of fingerprinting methods across different vantage points in the network.

Additionally, our experiments relied on laptops with a single crawler operating on each device to simulate user behavior. This setup was used for controlled data collection; it is not a realistic approximation of the complex scenario when several network activities happen

simultaneously on a single device. Several concurrent crawlers running on a single device can give a more realistic approximation of typical user behavior and network usage. The study of Deng et al. [42] has discussed multi-tab browsing, which has already introduced some challenges to website fingerprinting. Given this work, further research might examine the interactions between multi-crawler scenarios and interface congestion to better contextualize the performance of video fingerprinting under various complex network conditions.

5.3 Final Conclusions

Overall, when trying to recreate Walsh et al.'s results, while we could not exactly match the performance using Wi-Fi-collected traffic, we could still get results that showed we could consistently make accurate predictions on video traffic streamed over Tor. This performance is improved by using more training data, which is feasible for a nation-state adversary to implement. So, while using a network with less-than-ideal network conditions offers some modest protection from video fingerprinting, users must still worry about being unmasked in situations where they would be at risk under ideal conditions using the current best approaches.

To narrow down the cause of the reduced performance, we looked at varying the additional traffic across the interface. Overall, the fingerprinting process is fairly robust to increasing congestion of the network interface, with only extreme performance reduction occurring when nearly all available bandwidth is utilized. This situation is unlikely to occur for any extended period during everyday use, especially when using a purpose-driven platform like Tor.

Still, there are several reasons to be cautious about the ability of models to make accurate predictions in realistic, real-world environments. As discussed previously, crawlers hosted at data centers have an inherent advantage due to their proximity to the internet backbone, allowing faster data access. This advantage can be extended to highlight how the geographic location of a model can impact the timing of the traffic trace, making it more challenging for a model trained in one location to generalize effectively to traffic from other regions. This potential geographic variation further complicates the potential for cross-location prediction accuracy.

However, the most significant concern with video fingerprinting lies in the vast scale of modern video hosting platforms. As we established earlier, the effectiveness of a model decreases as the size of the open world increases. YouTube, for instance, is estimated to host approximately 14 billion videos as of 2023 [43], which far exceeds the relatively modest unmonitored set of 32,000 videos used in our study. This disparity in scale suggests that, while users may be more vulnerable to fingerprinting on niche, boutique video hosting platforms, they are less likely to face similar risks on larger, more well-established platforms such as Vimeo and YouTube. Consequently, while fingerprinting may pose a threat in narrow cases, users can generally be more confident in their privacy when engaging with larger video hosting sites.

APPENDIX: Videos Used in the Congested Sets

Appendix: Video References

1. **'geometry unit 2 lesson 1'**
Duration: 15:14
Quality: 720p HD
https://www.youtube.com/watch?v=B_lcFkD3A7Q
2. **'calculus 3 5'**
Duration: 26:05
Quality: 720p HD
<https://www.youtube.com/watch?v=dfSJ66u6A-Q>
3. **'calculus 3 1 part 1'**
Duration: 26:04
Quality: 720p HD
<https://www.youtube.com/watch?v=tEvWu9cE5SQ>
4. **'calculus 4 4'**
Duration: 16:56
Quality: 720p HD
<https://www.youtube.com/watch?v=5qFCIJXJoX8>
5. **'geometry unit 2 lesson 2'**
Duration: 15:13
Quality: 720p HD
<https://www.youtube.com/watch?v=k-JYkw7CiXY>
6. **'calculus 2 4 part 1'**
Duration: 19:19
Quality: 720p HD
<https://www.youtube.com/watch?v=J881dPTAGb4>
7. **'calculus 3 7 part 1'**
Duration: 15:11
Quality: 720p HD
<https://www.youtube.com/watch?v=Bd4IBDWI21c>

8. **'calculus 3 2 part 1'**

Duration: 8:32

Quality: 720p HD

<https://www.youtube.com/watch?v=R8tUS4pU5FA>

9. **'calculus 4 3 part 1'**

Duration: 23:33

Quality: 720p HD

<https://www.youtube.com/watch?v=f4qLx82HmPk>

10. **'calculus 3 1 part 2'**

Duration: 17:13

Quality: 720p HD

<https://www.youtube.com/watch?v=iTfQvKubHa0>

List of References

- [1] C. McClain, M. Faverio, M. Anderson, and E. Park, “How Americans View Data Privacy,” Pew Research Center, Rep., October 2023. Available: <https://www.pewresearch.org/>
- [2] Identity Theft Resource Center, “2023 Annual Data Breach Report,” 2024. Accessed: 2024-11-30. Available: <https://www.idtheftcenter.org/publication/2023-data-breach-report/>
- [3] A. Ghuman, “Research: A market where consumers can pay for privacy is emerging,” April 2021. Accessed: 2024-11-30. Available: <https://venturebeat.com/2021/04/30/research-a-market-where-consumers-can-pay-for-privacy-is-emerging/>
- [4] O. Akgul, R. Roberts, M. Namara, D. Levin, and M. Mazurek, “Investigating Influencer VPN Ads on YouTube,” in *2022 IEEE Symposium on Security and Privacy (SP)*, 2022. Available: <https://doi.org/10.1109/SP46214.2022.9833633>
- [5] The Tor Project, “Tor metrics: User statistics,” 2024. Accessed: 2024-11-28. Available: <https://metrics.torproject.org/userstats-relay-country.html?start=2024-01-01&end=2024-12-02&country=all&events=off>
- [6] T. Walsh, T. Thomas, and A. Barton, “Exploring the capabilities and limitations of video stream fingerprinting,” in *IEEE Security and Privacy Workshops*, 2024, pp. 28–39. Available: <https://doi.org/10.1109/SPW63631.2024.00008>
- [7] The Council of Economic Advisers, “The Cost of Malicious Cyber Activity to the U.S. Economy,” The Council of Economic Advisers, Rep., Feb. 2018. Available: <https://www.nitrd.gov/pubs/NationalPrivacyResearchStrategy.pdf>
- [8] Congressional Research Service, “Dark web,” Library of Congress, Rep., 2022. Accessed: 2024-10-21. Available: <https://crsreports.congress.gov/product/pdf/IF/IF12172>
- [9] The White House, “2023 National Cybersecurity Strategy,” 2023. Accessed: 2024-10-21. Available: <https://www.whitehouse.gov/briefing-room/statements-releases/2023/03/02/fact-sheet-biden-harris-administration-announces-national-cybersecurity-strategy/>
- [10] The White House, “2023 National Cybersecurity Strategy Implementation Plan,” 2023. Accessed: 2024-10-21. Available: <https://www.whitehouse.gov/briefing-room/statements-releases/2023/07/13/fact-sheet-biden-harris-administration-releases-implementation-plan-for-national-cybersecurity-strategy/>

- [11] G. King and H. Wang, “HTTPA: HTTPS Attestable Protocol,” 10 2021. Available: <https://doi.org/10.48550/arXiv.2110.07954>
- [12] W3Techs, “Usage statistics and market share of default protocol https for websites, october 2024,” 2024. Accessed: 2024-10-12. Available: <https://w3techs.com/technologies/details/ce-httpsdefault>
- [13] The Tor Project, “About tor.” Available: <https://support.torproject.org/about/>
- [14] R. Dingledine, N. Mathewson, and P. Syverson, “Tor: The second-generation onion router,” in *USENIX Security Symp.*, 2004. Available: <https://www.usenix.org/conference/13th-usenix-security-symposium/tor-second-generation-onion-router>
- [15] The Tor Project, “Network size metrics,” 2024. Accessed: 2024-10-13. Available: <https://metrics.torproject.org/networksize.html?start=2007-08-10&end=2024-11-08>
- [16] The Tor Project, “Collector: Relay descriptors archive - consensuses,” 2024. Accessed: 2024-10-13. Available: <https://metrics.torproject.org/collector/archive/relay-descriptors/consensuses/>
- [17] R. Jagerman, W. Sabee, L. Versluis, M. de Vos, and J. Pouwelse, “The fifteen year struggle of decentralizing privacy-enhancing technology,” *arXiv preprint arXiv:1404.4818*, 2014. Available: <https://arxiv.org/abs/1404.4818>
- [18] P. Winter, R. Ensafi, K. Loesing, and N. Feamster, “Identifying and characterizing sybils in the tor network,” in *25th USENIX Security Symposium (USENIX Security 16)*, 2016, pp. 1169–1185. Available: <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/winter>
- [19] A. Hintz, “Fingerprinting websites using traffic analysis,” in *Workshop on Privacy Enhancing Technologies*, 2002, pp. 171–178. Available: https://doi.org/10.1007/3-540-36467-6_13
- [20] X. Cai, X. Zhang, B. Joshi, and R. Johnson, “Touching from a distance: Website fingerprinting attacks and defenses,” in *ACM SIGSAC Conf. on Computer and Communications Security*, 2012, pp. 605–616. Available: <https://doi.org/10.1145/2382196.2382260>
- [21] Sandvine, “The Global Internet Phenomena Report October 2024,” https://www.sandvine.com/hubfs/Sandvine_Redesign_2019/Downloads/2024/GIPR/GIPR%202024.pdf, Mar 2024. [Online].
- [22] “ISO/IEC 14496-10:2022 Information Technology — Coding of Audio-Visual Objects — Part 10: Advanced Video Coding,” Geneva, Switzerland, Rep., 2022. Standard specification for advanced video coding.

- [23] International Organization for Standardization, “Information technology — Dynamic adaptive streaming over HTTP (DASH),” Rep. ISO/IEC 23009-1, 2022. Available: <https://www.iso.org/standard/83314.html>
- [24] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed. Sebastopol, CA: O’Reilly Media, 2023.
- [25] R. Schuster, V. Shmatikov, and E. Tromer, “Beauty and the burst: Remote identification of encrypted video streams,” in *USENIX Security Symp.*, 2017, pp. 1357–1374. Available: <https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/schuster>
- [26] P. Sirinam, M. Imani, M. Juarez, and M. Wright, “Deep fingerprinting: Undermining website fingerprinting defenses with deep learning,” in *ACM SIGSAC Conf. on Computer and Communications Security*, 2018, pp. 1928–1943. Available: <https://doi.org/10.1145/3243734.3243768>
- [27] J. Holt and P. Vonderau, *Where the internet lives: Data centers as cloud infrastructure*, 01 2015, pp. 71–93.
- [28] X. Zhang *et al.*, “Traffic spills the beans: A robust video identification attack against YouTube,” *Computers and Security*, vol. 137, no. C, p. 103623, 2024. Available: <https://doi.org/10.1016/j.cose.2023.103623>
- [29] A. Carlson, D. Hasselquist, E. Witwer, N. Johansson, and N. Carlsson, “Understanding and improving video fingerprinting attack accuracy under challenging conditions,” in *ACM Workshop on Privacy in the Electronic Society*, 2024. Available: <https://doi.org/10.1109/SP.2017.38>
- [30] T. Walsh, “Open-world video stream fingerprinting,” Ph.D. dissertation, Dept. of Comp. Sci., Naval Postgraduate School, Monterey, CA, USA. Unpublished doctoral dissertation.
- [31] Shaw Communications Inc., “Equipment info: Fibre+ gateway 2.0 (xb7).” Accessed: 2024-10-12. Available: <https://support.shaw.ca/t5/internet-articles/equipment-info-fibre-gateway-2-0-xb7/ta-p/29898>
- [32] T. Walsh, “tor-browser-crawler-video - capture streaming video traffic with and without the Tor Browser.” GitHub, May 16, 2024. Available: <https://github.com/timwalsh300/tor-browser-crawler-video>
- [33] M. S. Rahman, N. Matthews, and M. Wright, “Poster: Video fingerprinting in Tor,” in *ACM SIGSAC Conf. on Computer and Communications Security*, 2019, pp. 2629—2631. Available: <https://doi.org/10.1145/3319535.3363273>

- [34] B. Hayden, T. Walsh, and A. Barton, “Defending against deep learning based traffic fingerprinting attacks with adversarial examples,” *ACM Transactions on Privacy and Security*, vol. 28, no. 1, pp. 1:1–1:23, November 2024. Available: <https://doi.org/10.1145/3698591>
- [35] P. McClure *et al.*, “Knowing what you know in brain segmentation using bayesian deep neural networks.”
- [36] Y. Gal, J. Hron, and A. Kendall, “Concrete dropout,” 2017. Available: <https://arxiv.org/abs/1705.07832>
- [37] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2018. Available: <https://doi.org/10.48550/arXiv.1710.09412>
- [38] M. Kerrisk, *Linux tc command manual page*, 2024. Accessed: 2024-11-26. Available: <https://man7.org/linux/man-pages/man8/tc.8.html>
- [39] Federal Communications Commission, “FCC Household Broadband Guide,” 2022. Date Last Updated/Reviewed: Monday, July 18, 2022. Available: https://www.fcc.gov/sites/default/files/household_broadband_guide.pdf
- [40] The iPerf Project, “iperf - the ultimate speed test tool for tcp, udp, and sctp,” 2024. Accessed: 2024-11-28. Available: <https://iperf.fr/>
- [41] The Tor Project, “Relay requirements,” 2024. Accessed: 2024-12-01. Available: <https://community.torproject.org/relay/relays-requirements/>
- [42] X. Deng *et al.*, “Robust multi-tab website fingerprinting attacks in the wild,” in *IEEE Symp. on Security and Privacy*, 2023, pp. 1005–1022. Available: <https://doi.org/10.1109/SP46215.2023.10179464>
- [43] R. McGrady, K. Zheng, R. Curran, J. Baumgartner, and E. Zuckerman, “Dialing for videos: A random sample of youtube,” *Journal of Quantitative Description: Digital Media*, vol. 3, pp. 1–85, 2023. Licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International Public License. Available: <https://doi.org/10.51685/jqd.2023.022>

Initial Distribution List

1. Defense Technical Information Center
Fort Belvoir, Virginia
2. Dudley Knox Library
Naval Postgraduate School
Monterey, California



DUDLEY KNOX LIBRARY

NAVAL POSTGRADUATE SCHOOL

WWW.NPS.EDU

WHERE SCIENCE MEETS THE ART OF WARFARE