The More, the Better? A Study on Collaborative Machine Learning for DGA Detection

Arthur Drichel
drichel@itsec.rwth-aachen.de
RWTH Aachen University

Benedikt Holmes
holmes@itsec.rwth-aachen.de
RWTH Aachen University

Justus von Brandt
justus.von.brandt@rwth-aachen.de
RWTH Aachen University

Ulrike Meyer
meyer@itsec.rwth-aachen.de
RWTH Aachen University

ABSTRACT

Domain generation algorithms (DGAs) prevent the connection between a botnet and its master from being blocked by generating a large number of domain names. Promising single-data-source approaches have been proposed for separating benign from DGA-generated domains. Collaborative machine learning (ML) can be used in order to enhance a classifier's detection rate, reduce its false positive rate (FPR), and to improve the classifier's generalization capability to different networks. In this paper, we complement the research area of DGA detection by conducting a comprehensive collaborative learning study, including a total of 13,440 evaluation runs. In two real-world scenarios we evaluate a total of eleven different variations of collaborative learning using three different state-of-the-art classifiers. We show that collaborative ML can lead to a reduction in FPR by up to 51.7%. However, while collaborative ML is beneficial for DGA detection, not all approaches and classifier types profit equally. We round up our comprehensive study with a thorough discussion of the privacy threats implicated by the different collaborative ML approaches.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; • **Computing methodologies → Distributed artificial intelligence**.

KEYWORDS

Collaborative machine learning; Domain Generation Algorithm (DGA) detection; privacy

ACM Reference Format:
Arthur Drichel, Benedikt Holmes, Justus von Brandt, and Ulrike Meyer. 2021. The More, the Better? A Study on Collaborative Machine Learning for DGA Detection. In *Proceedings of the 3rd Workshop on Cyber-Security Arms Race (CYSARM '21), November 19, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3474374.3486915

CYSARM '21, November 19, 2021, Virtual Event, Republic of Korea

ACM ISBN 978-1-4503-8661-6/21/11…$15.00
https://doi.org/10.1145/3474374.3486915

1 INTRODUCTION

Domain generation algorithms (DGAs) are used by botnets to establish a connection between infected hosts and their command and control (C2) server. To this end, DGAs generate and query a large number of domain names, most of which result in non-existing domain (NXD) responses. This is because the botnet master only registers a handful of the generated domains while the bots query each domain. As a result, DGAs create an asymmetrical situation in which defenders have to block any generated domain name while a single resolving domain is sufficient for the botnet to receive new commands.

As a countermeasure, several machine learning (ML) classifiers have been proposed that attempt to separate benign from DGA-generated domains. These classifiers are trained in a supervised manner using malicious and benign labeled samples. Malicious labeled data can be easily obtained from open source intelligence (OSINT) feeds such as DGArchive [39]. Domain names that are available in public lists such as Alexa[1] or Tranco [28] can be used as benign data. However, it has been shown that classifiers trained on public benign data are less robust against adversarial attacks [10]. Moreover, in Section 4.2, we show that training with publicly available data is insufficient for DGA detection on private NXD data. There are additional practical benefits of using NXDs extracted from non-resolving DNS traffic (NX-traffic) for classifier training. First, DGAs are typically recognized in NX-traffic long before they resolve a registered domain for their C2 server. Therefore, infected hosts can be disinfected before they are ordered to take part in malicious actions. Second, the amount of NX-traffic is an order of magnitude less than the amount of full DNS traffic, which makes it easier to monitor. And third, NXDs are less privacy sensitive compared to resolving domain names because it is not possible to recover the full browser history of users from NXDs.

Nevertheless, NXDs may also contain sensitive information about individuals within an organization or confidential information about an organization as a whole. For instance, typing errors can still hint websites visited by employees. Moreover, the knowledge of NXDs generated by misconfigured or outdated software could be leveraged by an adversary to attack the organization's network. Due to these privacy issues, it is obviously not possible to directly share benign NXDs in a collaborative ML setting for DGA detection. Thus, we consider the disclosure of benign samples to be the main privacy-critical aspect in our use case.

[1] https://www.alexa.com/topsites

The goal of collaborative ML is to solve a data-driven task by using data stored by several different parties, on the assumption that more can be learned by combining knowledge from the participants' local datasets. Thereby, collaborative learning can aid in generalization of a trained global model and is a vital tool whenever local datasets have significant pairwise statistical difference or comprise only few samples each. In the past, several approaches for collaborative learning have been proposed including: (1) Transfer Learning, (2) Ensemble constructs, (3) Teacher-Student (T/S) learning, and (4) Federated Learning (FL). In each approach, different types of intelligence are shared including trained models or parts thereof, predictions for unknown samples, or weight updates during training. Depending on the type of intelligence shared, the precision with which information on the sensitive training data may be inferred varies.

In the context of DGA detection many single-data-source approaches exist. This paper complements this research area with an analysis of collaborative learning on this task. To the best of our knowledge, such a study has not been done so far and is missing in related literature.

Our contribution primarily focuses on a comparative evaluation of different collaborative learning approaches using three different state-of-the-art classifiers. Thereby, we answer the following three research questions:

(1) Is collaborative training beneficial for organizations that mostly classify data from their own network?
(2) Do jointly trained classifiers improve in their detection performance for samples originating from different networks?
(3) Do jointly trained classifiers improve in their detection performance with increasing participants?

The first two research questions are derived from possible real-world application environments for trained classifiers. The answer to the last research question reveals suitable approaches for DGA detection. In fact, we show that out of the four investigated collaborative ML approaches only Feature Extractor Sharing (an approach related to Transfer Learning) and FL are beneficial for DGA detection. Four collaborative ML variants based on T/S learning and Ensemble classification lead even to worse results compared to the baseline. In addition, in our study we compare three different state-of-the-art classifiers and show that different types of classifiers are better suited for different sharing approaches.

Secondary, we thoroughly discuss the privacy aspects concerning the disclosure of benign training samples for the collaborative approaches Feature Extractor Sharing and FL as well as the performance implications caused by privacy-preserving measures. As data is never directly shared in our approaches, we link data privacy solely to the trained models' or training procedures' vulnerability to privacy-threatening inference attacks.

2 RELATED WORK

In the following we briefly present the studied use case, introduce collaborative ML approaches as well as provide an overview on privacy research in deep learning (DL).

2.1 DGA Detection

Various approaches have been proposed in the past to capture DGA activity within networks. These approaches can be roughly divided into context-less (e.g., [10, 43, 45, 56, 61]) and context-aware approaches (e.g., [3, 6, 19, 44, 47, 58]). The former group uses information extracted only from individual domain names and ignores any contextual data to separate benign from DGA-generated domains. On the other hand, context-aware approaches use additional knowledge to further improve the detection performance. Previous studies such as [10, 45, 56, 61] have shown that the context-less approaches achieve state-of-the-art performance, are less resource-demanding, and are less privacy-invasive than context-aware approaches.

The proposed context-less ML classifiers can be further divided into feature-based such as support vector machines or random forests (e.g., [45]), and feature-less (DL) classifiers such as recurrent (RNN), convolutional (CNN), or residual neural networks (ResNet) (e.g., [10, 43, 56, 61]). When comparing both types of classifiers, it was shown that the approaches based on DL achieve superior detection performance [10, 37, 51, 56].

All of these proposed approaches have in common that they are single-data-source approaches. In this work, we use context-less DL based classifiers trained on NX-traffic to complement this research area with an analysis of collaborative learning for this detection task.

2.2 Collaborative Machine Learning

Focusing on different attributes such as communication & computation costs, retraining effort, as well as data availability, locality or privacy, yields various sharing approaches. We list the following approaches by an increasing order regarding the involvement and intercommunication of the sharing parties. Related collaborative learning or sharing approaches include (1) Transfer Learning, (2) Ensemble constructs, as well as more involved approaches, such as (3) Teacher-Student learning and (4) Federated Learning.

In Transfer Learning (e.g., [18, 34, 55, 65] and literature bodies referenced therein) a consuming party may leverage an existing neural network model pre-trained by a different party on a related task using a larger, more general, or more complex dataset than the one owned by the consumer. The consuming party can exchange and fit the tail of the model (decision layers) to its own dataset, making use of the knowledge held in the weights of the pre-trained extraction layers which remain unchanged during retraining.

Ensemble classifiers (e.g., [9, 18, 42]) denote a collection of ML models (trained on the same task) with the advantage that the individual errors of each single model are rectifiable by the others. Ensembles can be trained as a collective or constructed from pre-trained models to provide a global inference interface by combination of their distinct outputs (e.g., by averaging soft-labels or voting). In this work we view the latter case.

In a Teacher-Student (T/S) scenario (originated from [22]) one or multiple pre-trained teacher models guide the training of a student model. Guided training can, for instance, serve the purpose of improved training time on related tasks or model compression while maintaining the teacher's behavior and utility on the certain task (a

concept known as knowledge distillation [22]). In our work, we investigate the student's capability to retain and combine intelligence from all teachers.

Federated Learning (FL), introduced by McMahan et al. [31], enhances the state of distributed collaborative learning, as participants are not required to directly share their local private data but instead share gradient updates in an iterative training process in which local updates are aggregated and applied to a global model.

Impact of collaborative learning and sharing approaches with regard to improved performance in DL can be witnessed by examples in the research fields computer vision and natural language processing (e.g., [22, 40, 59]).

2.3 Privacy in Machine Learning

Due to its high consummation of sensitive data, DL models or training procedures have become the main suspect of research investigating threats (termed inference attacks) and defenses concerning the natural information leakage of consumed training data that is inherent to learning (e.g., [1, 2, 4, 15, 32, 36, 48]).

The most prominently researched privacy threat against classification models is the Membership Inference attack (MemI) [48] which aims at disclosing the participation of a known data sample in the training process of the targeted model. Another relevant attack is Model Inversion [15, 16] which utilizes the gradient of a model to reconstruct its inputs.

The best possible defense against inference attacks is achieved by applying well-defined cryptographic tools, such as secure multiparty computation (SMC) [41] or homomorphic encryption (HE) [38], to either the data, the model, or the learning or inference process. Unfortunately, their application on deep neural networks are accompanied by a non-negligible performance overhead [38, 41].

Differential Privacy [12] is a scheme for privacy-preserving data release and a common basis for MemI defenses. By applying deliberate noise to the ML training process, participation is hidden, e.g., on a per-sample basis, and therefore the MemI attack's success is impeded. An extension of DP, commonly used in privacy-preserving learning, bounds the attacker's capability of *distinguishability* by ε with a relaxation that exceptions to the rule may occur with a likelihood of δ [13].

DP-SGD [1] is a variant of the standard stochastic gradient descent algorithm (on which many ML optimizer base) that applies (ε, δ)-DP noise to the gradient. PATE, by Papernot et al. [35], is another exemplary approach that accomplishes DP, in which a local ensemble of private teacher models, each trained on a distinct partition of a sensitive dataset, labels a public dataset. This single-party training procedure counters the MemI attack by hiding the contribution of the teachers' labeling votes with targeted DP noise. Subsequently, the securely labeled data can be used to train a public student model.

Especially in sharing approaches, that include non-trivial training procedures or exchange of information through data or model sharing, privacy deserves great attention. In this work, we focus primarily on utility benefit, yet we also comment on privacy in sharing approaches that we deem beneficial for our use case.

3 EVALUATION SETUP

In this section, we provide an overview of our evaluation setup, including the state-of-the-art classifiers selected, the data sources used, our dataset generation scheme, the collaborative ML approaches examined, and the evaluation methodology used.

3.1 State-of-the-Art Classifiers

In the following, we introduce three state-of-the-art classifiers for DGA detection that will be used as part of our evaluation. All three classifiers are based on neural networks, operate on single domain names, and output a probability that indicates whether the input domains are benign or DGA-generated. We choose classifiers based on the following three aspects: (1) they achieve state-of-the-art detection performance, (2) they operate context-less, and (3) they are based on different types of neural network architectures. The latter allows us to examine whether the type of architecture used has an impact on the success of collaborative ML and, secondly, to compare the potential increase in performance. In detail, our classifier selection includes a RNN-based, a CNN-based, and a ResNet-based classifier. In addition to the output generated, all three classifiers share the same input pre-processing steps. Before a domain name is input to any classifier it is encoded using integer encoding (i.e., each unique character is replaced by a unique integer). The encoded domain names are then consumed by an embedding layer in order to incorporate semantics into the encoding. From then on, each classifier processes the embedded domain names differently, which we briefly present below. Detailed information on the implementations of the individual classifiers can be found in [10, 56, 61].

Endgame. Woodbridge et al. [56] proposed a RNN classifier based on a long short-term memory (LSTM) layer consisting of 128 nodes with sigmoid activation. We refer to this classifier as Endgame in the following.

NYU. Yu et al. [61] proposed a CNN-based model that uses two stacked 1-dimensional convolutional layers with 128 filters for separating benign from DGA-generated domain names. We refer to this model as NYU in the following.

ResNet. Drichel et al. [10] proposed a ResNet-based model for DGA binary classification. The model is build of a single residual block which incorporates a skip connection between convolutional layers. The skip connection allows the gradient to bypass the convolutional layers unaltered during training in order to counteract the vanishing gradient problem. Similar to the NYU model, the ResNet model includes two 1-dimensional convolutional layers with 128 filters in the residual.

3.2 Data Sources

In the following, we introduce five different data sources from which we obtain NXDs for our collaborative ML experiments. We use one source to obtain malicious labeled samples and four distinct sources from different networks for benign labeled data. This rich data enables us to conduct collaborative ML experiments that are similar to a real-world setting. Moreover, the different benign data sources allow us to investigate whether the collaboratively trained classifiers generalize to different networks.

3.2.1 Malicious Data: DGArchive. We obtain malicious labeled domain names from the OSINT feed of DGArchive [39]. For our evaluation we include all available samples up to September 1st, 2020. In total, DGArchive provides us with approximately 126 million unique domain names generated by 95 different DGA families.

3.2.2 Benign Data. We obtain benign labeled samples from four different data sources: two university networks ($University_A$ and $University_B$), the networks of an *Association* of universities, and the networks of a large *Company*. $University_B$ is also a member of the Association and thus the samples generated in the network of $University_B$ are also observable in the networks of the Association. Since the time intervals of the recordings for these two data sources overlap, we remove the intersection of all samples from both records in order to prevent an artificial increase in classification performance for a jointly trained classifier if samples from both sources are used. Otherwise, it would be possible that a classifier is evaluated on a test set that contains samples that were also used to train the classifier. Additionally, we compare all domain names obtained from all benign data sources against DGArchive and remove all matches in order to clean our data as much as possible. In the following, we provide a brief overview of the benign data sources.

$University_A$. We obtained a one-month recording of September 2019 from the central DNS resolver of RWTH Aachen University which is located in Germany. This recording comprises approximately 26 million unique NXDs that originate from academic and administrative networks, student residences' networks, and networks of the university hospital of RWTH Aachen.

$University_B$. We obtained a one-month recording from mid-May 2020 until mid-June 2020 from the networks of Masaryk University which is located in the Czech Republic. This recording contains approximately 8 million unique samples.

Association. We received additional benign samples from CESNET: an association of universities of the Czech Republic and the Czech Academy of Sciences consisting of 27 members in total. CESNET operates and develops the national e-infrastructure for science, research, and education. From this data source, we obtained a subset of occurred NXDs from the day recording of 2020-06-15. In total, we obtained approximately 362k unique samples.

Company. We obtained a one-month recording of July 2019 that comprises approximately 21 million unique NXDs from several DNS resolvers of Siemens AG which is a large company that operates in Asia, Europe, and in the USA.

3.3 Dataset Generation

In the following, we first describe our process of generating suitable datasets for our experiments using the above data sources. We provide an illustration of this process in Fig. 1 for convenience.

In order to create diverse datasets and to cope with the large amount of available training samples, we first subsample our malicious labeled data into a smaller set that includes at most 10k samples per DGA family. We include all samples for DGA families for which less than 10k samples are available. Thereby, we also include samples from underrepresented DGA families. We do this because in [11] it was shown that by including a few samples to the training of a classifier, its detection performance for underrepresented DGAs can be increased significantly without reducing its detection rates for well-represented DGAs.

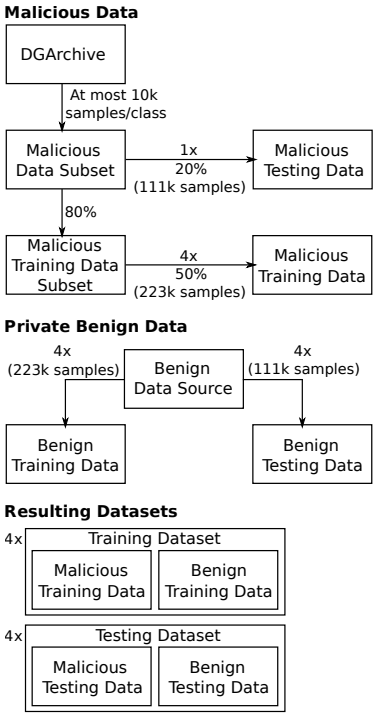

Figure 1: Datasets Generation Scheme

From the selected subset we then split 20% (approximately 111k samples) stratified across all included DGA families for the test sets. For each of our benign data sources we select individual malicious training data by subsampling 50% (approximately 223k samples) from the remaining malicious labeled samples. By subsampling from a larger common pool of malicious samples, we can create four training sets that contain both duplicate and unique malicious samples. We do this because, in a real-world scenario, it is very likely that the collaborating parties are using overlapping sets of malicious samples. Note, in contrast to the benign labeled samples, the malicious samples are available in public repositories and are not privacy-sensitive.

The benign samples are not shared between different training sets as they are considered to be the main privacy-critical aspect of the collaborative DGA detection use case. We carry out a similar selection process for the benign training and testing samples. From each of the four benign data sources, we randomly subsample the same amount of benign training and testing samples as we did for malicious training and testing samples, respectively.

We use these data selections to create four training and testing dataset pairs, one for each benign data source. To this end, we combine the respective malicious and benign data selections to balanced training and testing datasets. Note that during this dataset generation we ensured that the samples included in the training and testing datasets are completely disjoint. Each of the four training

and testing datasets include approximately 446k and 223k samples, respectively.

Additionally, we create two balanced training datasets that include publicly available benign data using the same generation process. These datasets are used to train initial global models for our FL experiments. The public benign data originates from the Tranco list [28], which contains a ranking of the most popular domains that has been hardened against manipulation. Using this data we create two datasets, one contains the top entries of the list, while the benign data of the other dataset consists of random samples.

3.4 Methodology & Sharing Approaches

Using these datasets, we are able to precisely measure the influence of collaborative ML on the classification performance of various classifiers. To obtain meaningful results, we repeat the whole dataset generation process five times and thereby create 20 individual training and testing dataset pairs which include malicious labeled samples from DGArchive and benign data from the four benign data sources. By this means, we also generate ten training datasets used in the FL experiments for training an initial global model using publicly available data. In the following, we repeat every experiment five times and present the averages of the individual results. Note, the datasets are generated similarly to a five-fold cross validation, i.e., the testing datasets are completely disjoint with both the training datasets and the testing datasets within the repetitions.

In this work, we train all classifiers using early stopping with a patience of three epochs to avoid overfitting and assess their performance during training on holdout sets that consist of random 5% splits of the used training data.

To measure the impact of collaborative ML on classification performance, we evaluate all possible combinations of participants for each examined approach. Overall, our comprehensive study consists of 13,440 evaluation passes. The total number of individual experiments differs between the various collaborative ML approaches.

In the following, we provide an overview on the sharing approaches we consider, all of which address the problem of collaborative ML. We first present the baseline and then the different sharing approaches including the number of experiments per approach.

3.4.1 Baseline. All sharing approaches are compared against the baseline to evaluate their performance. The baseline evaluations are similar to traditional training and testing of a classifier using training data from a single organization. Each organization trains their own model using their own private benign data and malicious training samples from DGArchive. No training data is shared among any organizations and also no global model is derived.

We train one classifier for each of the four organizations and evaluate them on every available test dataset. To this end, we perform five repetitions of training and testing for four organizations ($\text{University}_\text{A}$, $\text{University}_\text{B}$, Association, Company) and three classifier models (Endgame, NYU, ResNet), thus perform $5 \cdot 4 \cdot 4 \cdot 3 = 240$ baseline evaluations in total.

3.4.2 Ensemble Classification. In Ensemble classification a global classifier is build using the classifiers trained by each organization. Similar to the baseline scenario each organization first trains a classifier using their own private benign data. These classifiers are then shared with all participants. Each party now combines the individual classifiers to an Ensemble classifier. The combination of the classifiers can be done (1) by using a majority voting system on the binary labels (hard labels) or (2) by averaging the results of the individual classifiers to a single confidence score (soft labels). A tie is possible when using the majority voting system with an even number of participants. In such a case, we resort to the soft labels approach, where we average all predictions and predict a domain name as malicious if the average is greater than 0.5 and benign otherwise.

In total there are $\binom{4}{2} + \binom{4}{3} + \binom{4}{4} = 11$ possible combinations of organizations for building a combined model using four different parties. Here, we also evaluate classifiers on all four different testing datasets, regardless of the combination of local classifiers used. Thereby, we are able to measure the generalization capability of classifiers on benign data from unknown networks. In total, we perform five repetitions using eleven possible organization combinations, two ensemble approaches (hard and soft labels), four test datasets, and three classifier models ($5 \cdot 11 \cdot 2 \cdot 4 \cdot 3 = 1320$ evaluation passes).

3.4.3 Federated Learning. Federated learning (FL) [31] is a technique to train a classifier collaboratively without sharing local data. First a global model is initialized that is shared among all participants in the collaborative training. This global model can either be randomly initialized using standard initialization methods or can be pre-trained using non-sensitive public data. In this work, we evaluate FL using three different initial global models, one that is randomly initialized (Random Model), and two pre-trained models. For pre-training a global model, we make use of malicious samples from DGArchive and benign domain names from the Tranco list [28]. This list contains a ranking of the most popular domain names, which is also protected against manipulation. One pre-trained model uses the top entries of the Tranco list for benign domain names, while the other model uses random samples. In the following, we refer to these models as *Tranco Top* and *Tranco Random*, respectively. After the global model is shared among all participants an iterative training procedure is performed. The global model is trained locally by each organization using their own private training data for each federation step. The weight updates to the global model in each federation step are then shared with all other participants, such that everyone can now average the weight updates of the current step and add them to the global classifiers weights. Thereby, each party obtains the same global model that is then used within the next federation step. This iterative training process continues until the global model converges. The only data shared between the organizations are the model weight updates after each federation step and the initial global model. In this work, we investigate two federation approaches. In the first approach, we federate after each local model epoch (*Federation after Model Epoch*), while in the second approach we only federate once after all local models have converged (*Federation after Model Convergence*).

Here, we perform five repetitions using eleven organization combinations, three initial global models (Tranco Top, Tranco Random, Random Model), two possibilities for federation (after model epoch,

after model convergence), four test datasets, and three classifier models ($5 \cdot 11 \cdot 3 \cdot 2 \cdot 4 \cdot 3 = 3960$ evaluation passes).

3.4.4 Feature Extractor Sharing. This sharing approach is related to Transfer Learning. All classifiers under consideration (Section 3.1) are based on DL models that make use of a fully connected (dense) layer to output the final classification score. This layer can be viewed as a sort of classification layer that performs a logistic regression for binary classification. The output of this layer is a confidence score that indicates whether an input domain is benign (score < 0.5) or malicious (score ≥ 0.5). All layers before this classification layer can be treated as a feature extractor, which produces features used for classification by the final output layer. Instead of sharing the complete classifier as in the naive Ensemble approach, here, we hope to reduce the model's privacy leakage by sharing less layers. Thus in this approach, each organization trains a model based on their own private training data. Subsequently, the trained feature extractors are shared among all participants. Each organization now combines their own and received feature extractors to a new model. To this end, the feature extractors are applied in parallel and their outputs are concatenated and flattened. Additionally, a new dense classification layer is appended to the new model. This classification layer is not trained yet, thus the organizations freeze the weights of the feature extractors and use local training data to train the classification layer separately. In the end, each organization obtains a model which incorporates information about samples occurring in the other organizations through the shared feature extractors. The resulting models are not identical, since the training of the classification layer is performed on private training samples.

For this approach, each organization first trains a classifier using its own private benign data and derives an individual feature extractor. Then we combine the four feature extractors into eleven possible classifiers. In contrast to the two above mentioned approaches, here we require additional training to fit the final classification layer that combines the results of the shared feature extractors. Hence, in total we perform five repetitions using eleven possible organization combinations with four training datasets, four test datasets, and three classifier models ($5 \cdot 11 \cdot 4 \cdot 4 \cdot 3 = 2640$ evaluation passes).

3.4.5 Teacher-Student. The last examined sharing approach is based on a Teacher-Student (T/S) setup. Here, an organization queries trained classifiers of other organizations (teacher) in order to obtain labels for their own data. This labeled data is then used by the querying organization to train an own classifier (student). Using this approach the teacher classifiers are not exposed to the organization that is training the student classifier. Thereby, white-box attacks against the privacy of an organization that provides the labeling service are not applicable. Usually more than one teacher is involved in the labeling process of a training sample, thus the individual labels or scores need to be combined. Similar to Ensemble classification, we examine two approaches: (1) majority voting on binary labels (hard labels) and (2) soft labeling by averaging confidence scores. When using the majority voting approach, a tie is resolved in the same way as in Ensemble classification. For this approach no global shared model is trained, instead every party again derives an individual model similar to Feature Extractor Sharing.

We train classifiers that are similar to the baseline classifiers as teacher models for this approach. Similar to Feature Extractor Sharing, here we need a training dataset that is labeled by the teachers and used for training a student classifier. Thus, in total we perform five repetitions using eleven possible organization combinations with four training datasets, two teacher result combination approaches (hard and soft labeling), four test datasets, and three classifier models ($5 \cdot 11 \cdot 4 \cdot 2 \cdot 4 \cdot 3 = 5280$ evaluation passes).

3.4.6 Evaluation Metrics. As the benign training samples are identified as the main privacy-critical aspect of this use case we mainly use the false positive rate (FPR) as a proxy to determine the possible gain or loss in classification performance. Moreover, we argue that a low FPR is the most important attribute of a suitable classifier. Otherwise normal operation of a network would not be possible with too many false alarms. For the sake of completeness, we additionally provide the results for the accuracy (ACC) and the true positive rate (TPR) which is a proxy for determining the amount of detected DGA-generated domain names. Note that we use the same malicious samples in all four testing datasets within an experiment repetition. Thereby, we reduce the impact of the malicious samples on the ACC metric and thus ease the measurement of the classifiers' generalization ability for unseen benign samples that come from different networks.

3.4.7 Sharing Scenarios. The sharing approaches are evaluated in different scenarios which are derived from research questions on possible real-world application environments for trained classifiers.

Best Case. In this scenario, multiple network operators jointly train a classifier and are mostly interested in a good performance in their own networks. This is related to the following research question: is collaborative training beneficial for organizations that mostly classify data from their own distribution? In this best-case scenario, we provide averaged results for classifiers that are evaluated only on the test datasets containing samples coming from the organizations involved in the training.

Average Case. The average of all evaluations is used as a general performance indicator of the trained classifiers for each collaborative ML approach. We use this scenario for a comparative evaluation of the different sharing approaches.

Worst Case. The worst-case scenario is contrasting the best-case scenario. Here, classifiers are evaluated on all test datasets that contain samples from organizations that have not participated in the classifier training. Using this scenario, we examine the generalization capability of collaboratively trained classifiers (i.e., whether the classifiers improve in their detection performance for samples originating from different networks).

4 EVALUATION RESULTS

In this section, we present the results of our comprehensive study. First, we highlight differences between the four organizations' data and provide an overview of the performance in the three sharing scenarios. Subsequently, we present the results of our comparative evaluation. Finally, we analyze the effect of the number of participants in collaborative ML.

Table 1: Averaged Baseline Results for Endgame Classifier

Train Network	Test Network	ACC	TPR	FPR
University$_A$	**University$_A$**	0.99851	0.99968	0.00267
	University$_B$	0.99304	0.99968	0.01359
	Association	0.96892	0.99968	0.06184
	Company	0.99834	0.99968	0.00300
University$_B$	University$_A$	0.99717	0.99966	0.00532
	University$_B$	0.99808	0.99966	0.00350
	Association	0.97180	0.99966	0.05606
	Company	0.99853	0.99966	0.00259
Association	University$_A$	0.99674	0.99739	0.00390
	University$_B$	0.99444	0.99739	0.00852
	Association	0.99645	0.99739	0.00450
	Company	0.99771	0.99739	0.00196
Company	University$_A$	0.98923	0.99968	0.02122
	University$_B$	0.99037	0.99968	0.01894
	Association	0.96833	0.99968	0.06302
	Company	0.99888	0.99968	0.00192
Best Case		0.99798	0.99910	0.00315
Average Case		0.99103	0.99910	0.01703
Worst Case		0.98872	0.99910	0.02166

4.1 Network Differences & Sharing Scenarios

To better explain the actual evaluation steps and to detail the calculations done for the different sharing scenarios we present the average scores for five repetitions of the baseline experiment using the Endgame classifier in Table 1. We provide the results for the Endgame classifier as an example. The results for the NYU and ResNet models are similar.

Here we list the average scores for the five repetitions of Endgame classifiers per training dataset used and per test dataset separately.

The TPRs per training network are equal for all test networks as we use the same malicious samples within all four test datasets within an repetition (see Section 3.3 and Section 3.4.6).

The best FPRs on the individual test networks are always achieved by the classifiers that were trained using benign samples which originate from the same network as the testing samples. This is expected since those classifiers are specifically trained to extract and classify characteristics of the benign domain names from the respective network. For example, benign samples from different networks may miss certain features or exhibit other characteristics. The average of the table entries where the train network is equal to the test network represents the best-case scenario and is presented at the bottom of the table.

The classifiers trained using benign samples from distinct networks achieve different results for the individual test datasets. Samples from the Association are most commonly classified wrong. In some cases the FPR for these evaluations is even greater than 6%. We reckon that this is due to the fact that the samples from this network are the most diverse as they originate from over 27 different organizations. Moreover, we filtered out the intersection of the samples from the University$_B$ network with the samples from the Association as both networks are interconnected. Thereby, we likely removed easily recognizable samples that are naturally occurring in both networks. This could be the reason for the larger FPRs for University$_B$ and the Association compared to those for the other two networks.

Table 2: Results of Pre-trained Models using Public Data

Classifier	Benign Data	ACC	TPR	FPR
Endgame	Tranco Top	0.68329	0.95492	0.58834
	Tranco Random	0.67730	0.95054	0.59594
NYU	Tranco Top	0.68612	0.94876	0.57652
	Tranco Random	0.71336	0.94310	0.51637
ResNet	Tranco Top	0.78667	0.94952	0.37617
	Tranco Random	0.79899	0.93619	0.33821

Similarly to the best case, we provide the results for the average and worst case in the lower part of Table 1. While the average case is calculated using the average of all table entries, the worst case only contains the results of the entries for which the train network differs from the test network. The results for the different sharing scenarios are not of interest for the baseline evaluation. As could be expect, the best case results are better than the average case results, which are better than the worst case results.

4.2 Sharing Approaches

In this section, we compare the different approaches for collaborative ML. First, for comparison and to show that training on publicly available data is not enough for DGA detection on private data, we display the averaged results achieved by the two different types of pre-trained models that we use within our FL experiments for all three classifier types over all test datasets in Table 2.

All six trained classifiers yield high FPRs between 33% and 59%, indicating that training on publicly available data alone is insufficient for classifying privacy-sensitive domain names.

In Table 3, we present the results for the average case, for all classifiers, and all collaborative ML approaches examined.

In order to assess whether the collaborative training is beneficial we additionally provide the baseline results at the top of the table. For convenience, we color table entries red if the scores are worse than the ones of the baseline and green otherwise. All approaches except for the FL setting *Random Model - Model Convergence* achieve better TPRs than the baseline. This is an expected outcome because the training datasets used by the individual organizations contain additional malicious labeled samples from which a collaboratively trained classifier can learn. Thus, due to collaboration, intelligence about additional malicious labeled training samples is combined in the jointly trained classifiers.

The only exception is the FL setting *Random Model - Model Convergence*. In this setting, we use a randomly initialized model and federate the updates of the local models after they converged. While the NYU and the ResNet model are still functional and only achieve slightly worse classification scores, the TPR of the Endgame classifier falls from over 99.9% (baseline) to 55%. We reckon the reason to be that the model updates from a random initialized model to a fully converged model vary quite large and the individual organizations optimize their models to different local optima. Averaging and applying all model updates may result in an non-optimal global model. The Endgame model is far more affected by this compared to

Table 3: Results of the Average Case Including All Classifier Types, and All Collaborative ML Approaches

Approach	Endgame			NYU			ResNet		
	ACC	TPR	FPR	ACC	TPR	FPR	ACC	TPR	FPR
Baseline	0.99103	0.99910	0.01703	0.99151	0.99903	0.01600	0.99102	0.99844	0.01640
Ensemble: Soft Labels	0.98812	0.99975	0.02352	0.98870	0.99976	0.02236	0.98834	0.99946	0.02279
Ensemble: Hard Labels	0.98842	0.99981	0.02296	0.98901	0.99976	0.02174	0.98926	0.99909	0.02057
FL: Random Model - Model Convergence	0.76755	0.55131	0.01621	0.98300	0.98764	0.02163	0.98671	0.99278	0.01937
FL: Random Model - Model Epoch	0.99396	0.99968	0.01177	0.99392	0.99983	0.01199	0.99511	0.99935	0.00913
FL: Tranco Top - Model Convergence	0.99336	0.99958	0.01286	0.99325	0.99975	0.01326	0.99291	0.99974	0.01393
FL: Tranco Top - Model Epoch	0.99553	0.99956	0.00850	0.99400	0.99981	0.01181	0.99491	0.99922	0.00940
FL: Tranco Random - Model Convergence	0.99365	0.99946	0.01216	0.99360	0.99978	0.01257	0.99268	0.99967	0.01431
FL: Tranco Random - Model Epoch	0.99565	0.99952	0.00823	0.99413	0.99978	0.01153	0.99498	0.99929	0.00934
Feature Extractor Sharing	0.99394	0.99921	0.01133	0.99411	0.99913	0.01090	0.99307	0.99880	0.01266
T/S: Soft Labels	0.98979	0.99963	0.02006	0.99029	0.99965	0.01908	0.99001	0.99949	0.01948
T/S: Hard Labels	0.98966	0.99968	0.02036	0.99017	0.99965	0.01930	0.99026	0.99950	0.01898

the CNN-based NYU and ResNet model. This is due to the fact that RNNs are processing inputs sequentially. Averaging the weight updates of fully converged models that are used to process sequential data can thus result in a non-functional global model. In the following, we exclude the FL setting *Random Model - Model Convergence* from our study and mainly focus on the FPR for our assessment.

All other FL scenarios lead to an improvement compared to the baseline results. Here, the Endgame model performs significantly better in scenarios where a pre-trained initial global model was used. We assume that this is due to the fact that when using a pre-trained model, there are significantly fewer gradients towards local optima for participants to optimize their models. Similar to the FL setting *Random Model - Model Convergence*, this is an important property, especially for RNN-based classifiers. In contrast, the ResNet model achieves the best results using the randomly initialized model. In all FL settings, federating after each model epoch achieves better results than federating after model convergence.

The only other approach that leads to better classification results is Feature Extractor Sharing. The achieved results are comparable to the ones achieved by FL.

Ensemble classification and the T/S approach lead to worse results than the baseline. Comparing both approaches, the T/S approach yields a lower FPR for all three classifier types. Furthermore, with either approach, it makes little difference whether soft or hard labels are used.

While the absolute improvement achieved by collaborative ML may seem rather small, the relative reduction in the FPR is significant and could be decisive for the real-world application of classifiers. Compared to the baseline classifiers, the best approaches achieve on average a FPR reduction of 51.7%, 31.9%, and 44.3% for Endgame, NYU, and ResNet, respectively.

In summary, additional malicious samples in collaborative ML improve the TPRs for all sharing approaches. In the average-case scenario, only the collaborative ML approaches Feature Extractor Sharing and FL are advantageous for the use case of DGA detection. Using federation after model epoch leads to better results than federation after model convergence in FL.

4.3 Collaboration Analysis

In this section, our goal is to determine whether an increasing number of participants has a positive or negative effect on the classification performance of jointly trained classifiers. To this end, we investigate two scenarios.

In the first scenario, we make use of the best-case scenario defined in Section 3.4.7. Here, organizations want to use jointly trained classifiers to classify samples from their own networks most of the time. Thus, our goal is to determine whether the classification performance on those samples increases or decreases with an increasing number of participants. Thereby, organizations can decide whether or not it makes sense to use a collaboratively trained classifier for their own network.

In the second scenario, the worst case, we want to determine whether an increasing collaboration improves the generalization capabilities of the classifiers and thus the classification performance on samples from external networks.

We again use the FPR as a proxy to determine the performance of the classifiers. In Table 4, we present the achieved FPR scores for the different collaborative ML approaches and classifier types separated by the number of participants.

For visibility, we omit the ACC and the TPR metric, however, most of the time a better or worse FPR correlates with a better or worse ACC. In this evaluation we are not primarily interested in comparing the achieved scores of the different approaches with the performance of the baseline that is presented at the top of the table. Rather, we are interested in whether an increasing cooperation improves or worsens the performance achieved. Thus, we color code the entries different to Table 3. Here, we mark all table entries green if they are always improving with an increasing number of participants. When the approaches produce consecutive worse results we color them red. We do not color any entries for approaches for which the increases or decreases in classification performance are not consecutive. In the following, we present the evaluation results for the best and worst case in detail.

4.3.1 Best Case. Most of the collaborative approaches achieve (1) worse results compared to the baseline and (2) are decreasing in classification performance with increasing participants. This behavior can be explained by the fact that the baseline's best-case

Table 4: FPRs of the Best and Worst Case, Separated by Number of Participants

Approach	Parties	Best Case			Worst Case		
		Endgame	NYU	ResNet	Endgame	NYU	ResNet
Baseline	-	0.00315	0.00328	0.00326	0.02166	0.02024	0.02078
Ensemble: Soft Labels	2	0.02149	0.02023	0.02081	0.02420	0.02306	0.02376
	3	0.02393	0.02266	0.02316	0.02493	0.02409	0.02387
	4	0.02489	0.02402	0.02366	-	-	-
Ensemble: Hard Labels	2	0.02068	0.01978	0.01887	0.02399	0.02242	0.02127
	3	0.02332	0.02207	0.02077	0.02462	0.02322	0.02186
	4	0.02404	0.02316	0.02167	-	-	-
FL: Random Model - Model Epoch	2	0.00464	0.00620	0.00465	0.02032	0.01886	0.01668
	3	0.00738	0.00922	0.00516	0.02122	0.01786	0.01473
	4	0.01120	0.01120	0.00624	-	-	-
FL: Tranco Top - Model Convergence	2	0.00803	0.00783	0.00856	0.01769	0.01841	0.01924
	3	0.01147	0.01170	0.01230	0.01723	0.01834	0.01877
	4	0.01267	0.01365	0.01415	-	-	-
FL: Tranco Top - Model Epoch	2	0.00361	0.00535	0.00410	0.01571	0.01905	0.01655
	3	0.00491	0.00879	0.00597	0.01492	0.01830	0.01573
	4	0.00591	0.01205	0.00780	-	-	-
FL: Tranco Random - Model Convergence	2	0.00719	0.00745	0.00974	0.01763	0.01799	0.01889
	3	0.01022	0.01062	0.01297	0.01680	0.01773	0.01831
	4	0.01185	0.01235	0.01428	-	-	-
FL: Tranco Random - Model Epoch	2	0.00351	0.00520	0.00394	0.01551	0.01857	0.01593
	3	0.00458	0.00877	0.00599	0.01397	0.01775	0.01557
	4	0.00575	0.01146	0.00958	-	-	-
Feature Extractor Sharing	2	0.00306	0.00303	0.00314	0.01871	0.01756	0.01804
	3	0.00295	0.00302	0.00307	0.01796	0.01722	0.01784
	4	0.00293	0.00298	0.00302	-	-	-
T/S: Soft Labels	2	0.00328	0.00339	0.00339	0.02264	0.02143	0.02198
	3	0.01735	0.01703	0.01646	0.02252	0.02171	0.02247
	4	0.01822	0.01776	0.01744	-	-	-
T/S: Hard Labels	2	0.00329	0.00331	0.00333	0.02386	0.02132	0.02103
	3	0.01725	0.01795	0.01650	0.02208	0.02246	0.02218
	4	0.01766	0.01761	0.01694	-	-	-

scenario is the ideal training and classification setting. There, classifiers are assessed on data that comes from the same distribution as the samples used for training. No information of samples from other organizations are incorporated in those classifiers. Thereby, the baseline classifiers are specialized in classifying samples that originate from the same network as the training samples used. Thus, it is not surprising that the baseline classifiers achieve almost the best results compared to the other approaches. The collaborative ML approaches, on the other hand, incorporate also information of samples from other networks. Thereby, they are less specialized in classifying samples from a single network but rather are more generalized and thus achieve worse results compared to the baseline. The fact that these approaches achieve worse results with an increasing number of participants can be explained similarly. The more participants, the less the classifiers are specialized on samples of a single network. For example, Ensemble classification uses classifiers that are similar to the baseline classifiers. However, the more classifiers there are in an ensemble, the less influence a single classifier has on the final classification result. Therefore, in the best-case scenario, the classification performance decreases.

The only approach that improves with an increasing number of participants is Feature Extractor Sharing. Moreover, for all three classifier types, Feature Extractor Sharing achieves better FPRs than the baseline. This is because this approach creates models that are similar to the baseline but also incorporates additional information about samples from other networks via feature extractors that are applied in parallel. Since the shared feature extractors are not retrained, information about samples from individual networks is very well preserved with this approach. In addition, the intelligence incorporated in the shared feature extractors is harnessed by this approach and leads to improvement even beyond baseline. Note, although the differences in FPRs are rather small, we argue that our results are significant because of the large amount of evaluations done and the fact that this behavior is observable for all three types of classifiers.

From these results it can be seen that only Feature Extractor Sharing is beneficial in the best-case scenario, where organizations want to use collaborative ML classifiers to classify samples from their own network most of the time.

4.3.2 Worst Case. In this scenario, we evaluate whether an increasing number of participants improves the detection performance of jointly trained classifiers for samples that originate from external networks. Since we only have four different sources of benign data, the maximum number of participants in this scenario is three.

The results obtained for the Ensemble classification deteriorate as the number of participants increases for all three classifiers. The FPRs for the T/S approach improve only for the Endgame classifier. However, the achieved rates are worse than those of the baseline.

For FL, the FPRs improve in all settings and for all classifiers except for Endgame when a randomly initialized model is used. We assume that this is due to the same reasons given in Section 4.2. The ResNet classifier, however, achieves the best results using a randomly initialized model. The achieved FPRs for Endgame and ResNet using federation after model epoch are significantly lower than for the approaches that make use of federation after model convergence. For the NYU classifier, no significant difference can be measured for the various models.

For Feature Extractor Sharing, the FPRs achieved improve with an increased number of participants for all three classifier types. However, the rates are significantly worse than the ones achieved by Endgame and ResNet using FL with federation after model epoch. For NYU, the rates are comparable to those seen in the FL settings.

In summary, in the worst-case scenario, only the Feature Extractor Sharing and FL approaches improve in performance with an increasing number of participants and achieve better scores than the baseline. FL with federation after model epoch achieves best results for Endgame and ResNet and thus generalizes best to different networks. When comparing the different types of classifiers, Endgame and ResNet are better suited for FL than NYU. For RNN-based classifiers pre-trained initial models should be used.

5 PRIVACY DISCUSSION

We complement our most beneficial approaches, Feature Extractor Sharing and FL, with a thorough discussion on relevant privacy-threatening inference attacks.

5.1 Feature Extractor Sharing

Partial models are shared in this approach, giving an adversary white-box access to the gradient computation and the weights, that are directly influenced by the data in the learning process.

5.1.1 Model Inversion. In a white-box setting, gradient-based Model Inversion attacks [15, 16] may be deployed to iteratively reconstruct an input to the shared model utilizing its gradient and a loss comparing the desired output and the model's output for the reconstructed input. The reason we consider this a minor threat is two-fold: (1) Model Inversion attacks do not perform well in practice, especially if the targeted model is highly complex (in number of layers or weights) [23]. (2) Reconstructing true inputs, requires to know a set of the targeted party's feature vectors, which are however *not* shared in this approach.

5.1.2 Membership Inference. The classic MemI attack is defined for complete classifier models, i.e., with a final softmax output. Theoretically, the MemI attack setup (as in [48]) could be redefined for the case of a partial model. To the best of our knowledge such research has, however, not been conducted yet.

5.2 Federated Learning

The FL paradigm was proposed to enhance the data privacy for participants by reducing the exposure of their data. This alone is not sufficient, as attacks still threaten data privacy in FL [29, 46].

5.2.1 Gradient Leakage. To retain valuable contribution from parties in FL, it is crucial to ensure the non-disclosure of a participant's local data. However, retaining data locality in FL is not sufficient, as [33, 63, 64] demonstrate the necessity to shield gradient updates from inspection by the aggregating instance(s) that may reconstruct or infer sensitive data. Inference attacks during execution of the FL protocol can be rendered infeasible via a secure aggregation protocol [8], which computes the global average gradient in a SMC protocol that completely obstructs the inspection of local updates, thereby providing the best possible privacy. Multiple improvements have been proposed (e.g., [21, 50]): With the currently best performance overhead of $O(N\log N)$ per FL round [50], deploying secure aggregation is easily applicable in our use case with $N \leq 4$ parties.

5.2.2 Membership Inference. In FL, a MemI attack against the globally trained model threatens disclosure of participation on both sample level (i.e., the classic MemI attack [48]) as well as on client level [54]. A multitude of DP-driven research in FL exists (e.g., [17, 23, 25–27, 62]), that propose or investigate DP-based defenses and on occasion examine the inherent privacy/utility trade-off. Improvements of or alternatives to the classic DP-SGD for the FL setting are proposed in [24, 60], any of which can be applied to the local training of each party. DP yields sound privacy guarantees w.r.t. the bounds (ε, δ), the quality of which are however influenced by the individual use case and its available data and hence, application of DP would require a further assessment of the resulting privacy/utility trade-off.

5.2.3 Byzantine Attackers. Our work only considers the presence of trusted parties, yet for completeness we also give a quick view on Byzantine parties in FL [7], that are defined by arbitrary or faulty behavior, including intentional misbehavior such as privacy-threatening Free-riding [14] or sabotage (as in Poisoning [53] or Backdooring [5, 52]).

Distributed learning in the Byzantine setting has been studied in [7, 30] (and larger literature bodies referenced in [20, 49]). Blanchard et al. [7] argue that a single Byzantine user can influence any linear aggregation mechanism, and therefore also model conversion, to an arbitrarily large extent and present their first Byzantine-tolerant defense termed Krum. Other effective defenses have been proposed, which base on Krum or utilize similar insights that updates from malicious FL parties are separable from benign ones and can be filtered out [7, 49, 53, 57]. Malecki et al. [30] propose a Byzantine-robust and Sybil-resistant defense.

So et al. [49] additionally present an integration of their defense mechanism into the secure aggregation protocol by Bonawitz et al. [8] utilizing secure distance computation via HE. Unfortunately, Guerraoui et al. [20] provide first insights into the incompatibility of DP-based and Byzantine defenses.

6 CONCLUSION

In this paper, we performed a comprehensive study of collaborative ML for the real-world use case of DGA detection and discussed the privacy implications caused by beneficial sharing approaches. Thereby, we identified advantageous and disadvantageous approaches for different types of classifiers and showed that collaborative ML can lead to a reduction in FPR by up to 51.7%. Additionally, we showed that the usage of publicly available data is insufficient for DGA detection on private data. This shows the need for privacy-preserving collaborative ML approaches to make use of the advantages provided by classifying NXDs for DGA detection. In two real-world cases, we have shown that greater participation in collaborative ML does not always lead to better classification results. In fact, we only assess Feature Extractor Sharing and FL of the four examined collaborative ML approaches as beneficial for DGA detection. Feature Extractor Sharing should be used if a party wants to classify samples that come from their own network most of the time. FL on the other hand generalizes best to unknown networks. The four examined collaborative ML variants based on T/S learning and Ensemble classification lead to worse results than the baseline.

In terms of privacy, we have additionally discussed the applicability of inference attacks in our two beneficial approaches. Related state-of-the-art defense mechanisms enable enhanced privacy for sharing in our use case with negligible overhead. For the MemI attack it remains to assess the privacy/utility trade-off, that is inherent to DP-based defenses and the data in our use case. Also, MemI on partial models (Feature Extractor Sharing) is not regarded in prior work so far.

ACKNOWLEDGMENTS

The authors would like to thank Daniel Plohmann for granting us access to DGArchive as well as Jens Hektor from the IT Center of RWTH Aachen University, Masaryk University, CESNET, and Siemens AG for providing NXD data. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 833418. Simulations were performed with computing resources granted by RWTH Aachen University under project rwth0438.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *Computer and Communications Security*. ACM.

[2] Mohammad Al-Rubaie and J. Morris Chang. 2019. Privacy-Preserving Machine Learning: Threats and Solutions. In *Security & Privacy*. IEEE.

[3] Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou, Saeed Abu-Nimeh, Wenke Lee, and David Dagon. 2012. From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware. In *USENIX Security Symposium*.

[4] Giuseppe Ateniese, Luigi V Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. 2015. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. In *International Journal of Security and Networks*. Inderscience Publishers.

[5] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2020. How to Backdoor Federated Learning. In *Artificial Intelligence and Statistics*. PMLR.

[6] Leyla Bilge, Sevil Sen, Davide Balzarotti, Engin Kirda, and Christopher Kruegel. 2014. Exposure: A Passive DNS Analysis Service to Detect and Report Malicious Domains. In *Transactions on Information and System Security*. ACM.

[7] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. 2017. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Neural Information Processing Systems*. Curran Associates Inc.

[8] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. 2017. Practical Secure Aggregation for Privacy-Preserving Machine Learning. ACM.

[9] Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*. Springer.

[10] Arthur Drichel, Ulrike Meyer, Samuel Schüppen, and Dominik Teubert. 2020. Analyzing the Real-World Applicability of DGA Classifiers. In *Conference on Availability, Reliability and Security*. ACM.

[11] Arthur Drichel, Ulrike Meyer, Samuel Schüppen, and Dominik Teubert. 2020. Making Use of NXt to Nothing: Effect of Class Imbalances on DGA Detection Classifiers. In *Conference on Availability, Reliability and Security*. ACM.

[12] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming*. Springer.

[13] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *Advances in Cryptology*. Springer.

[14] Yann Fraboni, Richard Vidal, and Marco Lorenzi. 2021. Free-rider Attacks on Model Aggregation in Federated Learning. In *Artificial Intelligence and Statistics*. PMLR.

[15] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. 2015. Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures. In *Computer and Communications Security*. ACM.

[16] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. 2014. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In *USENIX Security Symposium*.

[17] Robin C Geyer, Tassilo Klein, and Moin Nabi. 2017. Differentially Private Federated Learning: A Client Level Perspective. *arXiv:1712.07557*.

[18] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep Learning*. MIT press.

[19] Martin Grill, Ivan Nikolaev, Veronica Valeros, and Martin Rehak. 2015. Detecting DGA Malware Using NetFlow. In *IFIP/IEEE International Symposium on Integrated Network Management*.

[20] Rachid Guerraoui, Nirupam Gupta, Rafaël Pinot, Sébastien Rouault, and John Stephan. 2021. Differential Privacy and Byzantine Resilience in SGD: Do They Add Up? *arXiv:2102.08166*.

[21] Jiale Guo, Ziyao Liu, Kwok-Yan Lam, Jun Zhao, Yiqiang Chen, and Chaoping Xing. 2020. Secure Weighted Aggregation in Federated Learning. *arXiv:2010.08730*.

[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *arXiv:1503.02531*.

[23] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. 2017. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. In *Computer and Communications Security*. ACM.

[24] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. 2021. Practical and Private (Deep) Learning without Sampling or Shuffling. *arXiv:2103.00039*.

[25] Peter Kairouz and H. Brendan McMahan. 2021. Advances and Open Problems in Federated Learning. *Foundations and Trends in Machine Learning*.

[26] Raouf Kerkouche, Gergely Ács, Claude Castelluccia, and Pierre Genevès. 2021. Constrained Differentially Private Federated Learning for Low-bandwidth Devices. *arXiv:2103.00342*.

[27] Muah Kim, Onur Gunlu, and Rafael F. Schaefer. 2021. Federated Learning with Local Differential Privacy: Trade-offs between Privacy, Utility, and Communication. Cryptology ePrint Archive, Report 2021/142.

[28] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczynski, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Network and Distributed System Security Symposium*. Internet Society.

[29] Lingjuan Lyu, Han Yu, Xingjun Ma, Lichao Sun, Jun Zhao, Qiang Yang, and Philip S. Yu. 2020. Privacy and Robustness in Federated Learning: Attacks and Defenses. *arXiv:2012.06337*.

[30] Nicholas Malecki, Hye-young Paik, Aleksandar Ignjatovic, Alan Blair, and Elisa Bertino. 2021. Simeon – Secure Federated Machine Learning Through Iterative Filtering. *arXiv:2103.07704*.

[31] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Artificial Intelligence and Statistics*. PMLR.

[32] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In *Security and Privacy*. IEEE.

[33] Tribhuvanesh Orekondy, Seong Joon Oh, Yang Zhang, Bernt Schiele, and Mario Fritz. 2020. Gradient-Leaks: Understanding and Controlling Deanonymization in Federated Learning. *arXiv:1805.05838*.

[34] Sinno Jialin Pan and Qiang Yang. 2010. A Survey on Transfer Learning. In *Transactions on Knowledge and Data Engineering*. IEEE.

[35] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2017. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. *arXiv:2012.02670*.

[36] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael P. Wellman. 2018. SoK: Security and Privacy in Machine Learning. In *European Symposium on Security and Privacy*. IEEE.

[37] Jonathan Peck, Claire Nie, Raaghavi Sivaguru, Charles Grumer, Femi Olumofin, Bin Yu, Anderson Nascimento, and Martine De Cock. 2019. CharBot: A Simple and Effective Method for Evading DGA Classifiers. *IEEE Access* 7 (2019).

[38] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. 2018. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. In *Transactions on Information Forensics and Security*. IEEE.

[39] Daniel Plohmann, Khaled Yakdan, Michael Klatt, Johannes Bader, and Elmar Gerhards-Padilla. 2016. A Comprehensive Measurement Study of Domain Generating Malware. In *USENIX Security Symposium*.

[40] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. In *Computer Vision and Pattern Recognition Workshops*. IEEE.

[41] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason V. Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. 2018. A generic framework for privacy preserving deep learning. *arXiv:1811.04017*.

[42] Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8, 4 (2018).

[43] Joshua Saxe and Konstantin Berlin. 2017. eXpose: A Character-Level Convolutional Neural Network with Embeddings For Detecting Malicious URLs, File Paths and Registry Keys. *arXiv:1702.08568*.

[44] Stefano Schiavoni, Federico Maggi, Lorenzo Cavallaro, and Stefano Zanero. 2014. Phoenix: DGA-Based Botnet Tracking and Intelligence. In *Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer.

[45] Samuel Schüppen, Dominik Teubert, Patrick Herrmann, and Ulrike Meyer. 2018. FANCI: Feature-Based Automated NXDomain Classification and Intelligence. In *USENIX Security Symposium*.

[46] Sheng Shen, Tianqing Zhu, Di Wu, Wei Wang, and Wanlei Zhou. 2020. From distributed machine learning to federated learning: In the view of data privacy and security. *Concurrency and Computation: Practice and Experience*.

[47] Yong Shi, Gong Chen, and Juntao Li. 2018. Malicious Domain Name Detection Based on Extreme Machine Learning. *Neural Processing Letters* 48, 3.

[48] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *Security and Privacy*. IEEE.

[49] Jinhyun So, Başak Güler, and A. Salman Avestimehr. 2020. Byzantine-Resilient Secure Federated Learning. *IEEE Journal on Selected Areas in Communications*.

[50] Jinhyun So, Başak Güler, and A. Salman Avestimehr. 2021. Turbo-Aggregate: Breaking the Quadratic Aggregation Barrier in Secure Federated Learning. *IEEE Journal on Selected Areas in Information Theory*.

[51] Jan Spooren, Davy Preuveneers, Lieven Desmet, Peter Janssen, and Wouter Joosen. 2019. Detection of Algorithmically Generated Domain Names Used by Botnets: A Dual Arms Race. In *Symposium On Applied Computing*. ACM.

[52] Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. 2019. Can You Really Backdoor Federated Learning? *arXiv:1911.07963*.

[53] Vale Tolpegin, Stacey Truex, Mehmet Emre Gursoy, and Ling Liu. 2020. Data Poisoning Attacks Against Federated Learning Systems. In *Computer Security – ESORICS 2020*. Springer.

[54] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning. In *Computer Communications*. IEEE.

[55] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big Data* 3, 1.

[56] Jonathan Woodbridge, Hyrum S. Anderson, Anjum Ahuja, and Daniel Grant. 2016. Predicting Domain Generation Algorithms with Long Short-Term Memory Networks. *arXiv:1611.00791*.

[57] Krishna Yadav and B. B. Gupta. 2021. Clustering Algorithm to Detect Adversaries in Federated Learning. *arXiv:2102.10799*.

[58] Sandeep Yadav and A. L. Narasimha Reddy. 2011. Winning with DNS Failures: Strategies for Faster Botnet Detection. In *Security and Privacy in Communication Systems*. Springer.

[59] Timothy Yang, Galen Andrew, Hubert Eichner, Haicheng Sun, Wei Li, Nicholas Kong, Daniel Ramage, and Françoise Beaufays. 2018. Applied federated learning: Improving google keyboard query suggestions. *arXiv:1812.02903*.

[60] Chencheng Ye and Ying Cui. 2021. Sample-based Federated Learning via Mini-batch SSCA. *arXiv:2103.09506*.

[61] Bin Yu, Jie Pan, Jiaming Hu, Anderson Nascimento, and Martine De Cock. 2018. Character Level Based Detection of DGA Domain Names. In *International Joint Conference on Neural Networks*. IEEE.

[62] Qinqing Zheng, Shuxiao Chen, Qi Long, and Weijie Su. 2021. Federated f-Differential Privacy. In *Artificial Intelligence and Statistics*. PMLR.

[63] Junyi Zhu and Matthew Blaschko. 2021. R-GAP: Recursive Gradient Attack on Privacy. *arXiv:2010.07733*.

[64] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep Leakage from Gradients. In *Advances in Neural Information Processing Systems*.

[65] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2021. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* 109, 1.