Subpopulation Data Poisoning Attacks on NLP Models

Kris Frasheri
kfrasher@uwaterloo.ca

Judy Lin
yz3lin@uwaterloo.ca

Prabhjot Singh
prabhjot.singh@uwaterloo.ca

Abstract

Data is power. This ability allows us to design machine learning and deep learning algorithms that aid us in nearly every aspect of life. However, research has demonstrated that we can exploit this power and manipulate these algorithms by introducing data anomalies. One such method is to employ data poisoning attacks, which corrupt the data being used to train a model and influence system's infrastructure. The most recent of these types of attacks is the subpopulation data poisoning attack, which allows the adversary to achieve maximum damage to the intended target with minimal collateral damage to the overall model. In this paper, we investigate the application of the subpopulation attack to NLP models. We performed empirical analysis on three prominent Transformer-based Pre-trained Language Models: BERT, XLNet and ELECTRA; demonstrating a high target damage for all models. We then employ the poisoning availability defense TRIM to determine subpopulation attack efficacy against defended models. Overall, we found that the subpopulation attack was less effective against XLNet than it was against BERT and ELECTRA.

1 Introduction

In this data age, scholars have designed advanced deep and machine learning systems to automate all the complex tasks. These tasks can be broadly classified to computer vision and natural language processing tasks. Computer vision tasks (use image as data source) include face recognition, object detection, video rendering and so on. CNN models have shown great success in supervised computer vision problems. Whereas, NLP tasks (use text and audio as data source) include emotion detection, voice recognition, language translation etc. Given the sequential nature of the data, RNN have proved to be more affective. With time, researchers have moved to generative models, which are more extensive and provide better results. Now the question arises, Are these systems security reliable? The answer to this is unfortunately no.

Over time, researchers have demonstrated that deep learning networks can be deceived by introducing modest modifications to the input data. These kinds of attacks are referred to as evasion attacks, and they have been extensively investigated in image classification Szegedy et al. [2013], Goodfellow et al. [2014], Carlini and Wagner [2017] and speech recognition Carlini and Wagner [2018], Schönherr et al. [2018]. As every firm in the world increasingly relies on data to power machine learning systems, adversaries can now manipulate these data points via data poisoning attacks. As poisoning assaults tamper with the machine learning infrastructure of these machine learning systems, their unreliability increases.

In existing poisoning attacks, adversaries can inject corrupted, poisoned data during training to produce a certain classification conclusion during inference. Existing poisoning attacks can be categorised as follows: availability attacks Biggio et al. [2012], Xiao et al. [2015], in which the

Preprint. Under review.

overall accuracy of the model is diminished; targeted attacks Koh and Liang [2017], Shafahi et al. [2018], in which specific test instances are targeted for misclassification; and backdoor attacks Gu et al. [2019], in which a backdoor pattern added to testing points causes misclassification. Poisoning attacks differ in the amount of knowledge the attacker has about the ML system, including knowledge about feature representation, model architecture, and training data, among others Suciu et al. [2018].

Literature-defined threat models for poisoning assaults rely on substantial assumptions regarding the adversary's capabilities. In both poisoning availability attacks Biggio et al. [2012], Xiao et al. [2015] and backdoor attacksGu et al. [2019], the adversary must access a significant portion of the training data (e.g., 10% or 20%) in order to impact the model at inference time. In addition, in backdoor assaults, it is expected that the adversary has the capacity to modify both the training and testing data to incorporate the backdoor pattern. In targeted attacks, it is assumed that the adversary knows the exact locations of the target points during training, which is not always the case. Moreover, the impact of a targeted attack is confined to a single site or a small group of points Koh and Liang [2017], Shafahi et al. [2018]. This bridge between this category of attacks is eliminated by subpopulation attack. This attack tries to maximize the damage for target test case, using minimum number of examples while preserving the overall accuracy of the model. They achieve this without tampering with the examples (use label flipping), in-turn making this type of attack, hard to detect and create countermeasures.

Since, subpopulation attack is relatively a new attack, the work in efficacy and feasibility of this attack has not been completed. Jagielski et al. [2020] design an effective threat model for this attack and explore the effectiveness of this attack in computer vision field. In the NLP field they just use BERT model Jagielski et al. [2020] on IMDB dataset to check the working of the attack, but no work is done further than that. This research gap is exploited by us, we check the effectiveness of subpopulation attack on different models, demonstrate how this attack is happening and check its feasibility against common defence.

2 Related Works

Poisoning attacks against machine learning models is classified into availability, backdoor and targeted attacks. Impactful research in above fields is discussed below.

Poisoning Availability Attacks: The concept of manipulating the training data of an automated classifier in order to introduce errors in the final model has been the subject of numerous studies over time. Attacks against polymorphic worm detectors Perdisci et al. [2006], network packet anomaly detectors Rubinstein et al. [2009], and behavioural malware clustering Biggio et al. [2014] are among the earliest studies in this field. Multiple models, including linear regression Xiao et al. [2015], Jagielski et al. [2018a] logistic regression Mei and Zhu [2015], and SVM Biggio et al. [2012], have been the subject of availability attacks based on gradient descent. These assaults aim to impair the accuracy of the model without discrimination. Regarding defences, SEVER Diakonikolas et al. [2019] use SVD to eliminate gradient-biased points, whilst TRIM Jagielski et al. [2018a] and ILTM Shen and Sanghavi [2019] eliminate points with significant loss. These defences function sequentially, finding and eliminating outlying spots at each stage until convergence is achieved. Demontis et al. [2019] Demontis et al. examine the transferability of poisoning availability attacks.

Backdoor Attacks: While red-herring attacks Newsome et al. [2006] might be viewed as antecedents to backdoor attacks, Gu et al. [2019] is recognised as the first backdoor attack against modern neural networks. It involved producing poisoned data with a backdoor pattern to influence the model to falsely classify additional backdoored testing points in order to identify a security risk with ML-as-a-service models. Clean-label backdoor attacks presuppose that the adversary does not have control over the labelling function Turner et al. [2019]. Other machine learning applications, such as Federated Learning models, are susceptible to backdoor attacks Bagdasaryan et al. [2020]. In order to guard against backdoor assaults, Tran et al. [2018] use SVD decomposition on the latent space learned by the network to generate an outlier score. The Liu et al. [2018] algorithm combines network pruning and fine-tuning. Wang et al. [2019] Identify poisoning by measuring the lowest perturbation necessary to change inputs into a target class.

Targeted Attacks: Shafahi et al. [2018] present an optimization-based, clean-label poisoning attack. Suciu et al. [2018] examine the transferability of targeted attacks. Schuster et al. [2020] demonstrate targeted poisoning attacks against NLP-related word embedding algorithms. Koh and Liang [2017]

describe an influence-based targeted attack and provide an illustration of a targeted strike that simultaneously affects many sites. Koh et al. [2019] analyse the efficacy of influence functions for forecasting how models evolve while removing numerous points from a dataset. Witches' Brew Geiping et al. [2021] use a gradient matching loss in conjunction with several optimization strategies to conduct targeted attacks that need little knowledge of the learner's setup.

The subpopulation attack largely falls within the category of targeted attacks. A priori work selects random target examples, whereas subpopulation attack does not. It uses *FeatureMatch* or *ClusterMatch* algorithm Jagielski et al. [2020] to locate target subpopulations and generate poisoned data relating to those targets (which helps us determine target damage).

3 Subpopulation Attack

As the name literally suggest this is the attack on subpopulation of a dataset. Before considering the details of the attack a formal definition of subpopulation needs to be stated.

A subpopulation of a data distribution is a restriction of its input domain to a set of points which are close to each other based on some distance function.

3.1 Threat Model

Akin to most poisoning attacks, the adversary's objective is to introduce a small number of contaminants into the data used to train a machine learning classification model in order to add a desired characteristic into the learnt parameters. We examine a realistic adversary that does not have access to the victim model's internal settings and, like availability poisoning assaults, cannot edit any data point submitted to the victim model during testing. In addition, the opponent is unable to discern the precise training data points and can only modify the training set by adding new points. This depicts the scenario in which the attacker can only spread poisoned points, which are then compiled with a huge number of benign points by the victim model's developers to form the training set. However, we let the adversary to have the computational resources necessary to train a model equivalent to that of the victim, as well as access to a second auxiliary dataset D_{aux}, different from the training data D, sampled from the same distribution. We also permit the adversary to know the learner's loss function and architecture. The adversary is unaware of the victim's model's parameters and training data Jagielski et al. [2020].

3.2 Attack Method

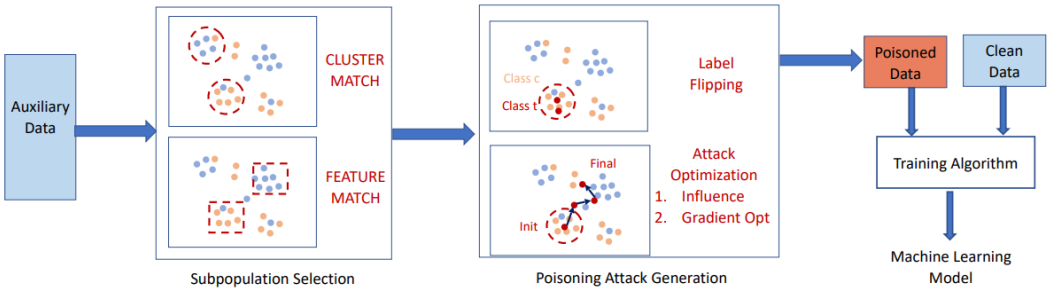

Figure 1: Overview of our subpopulation attack framework. The attacker has access to an auxiliary dataset, from which it can determine vulnerable subpopulation by using either *FeatureMatch* or *ClusterMatch*. Poisoning attack generation can be done by label flipping (where a point drawn from a subpopulation with majority class c is added with label $t \neq c$, or with attack optimization (starting from label flipping, use either influence or gradient optimization for the final attack point).

The training data is equally divided in D_{aux} and D dataset. The D_{aux} is used to study the dataset, examine target subpopulation and generate poisoned images. These poisoned images are then added to the D dataset and final model is trained on this concatenated dataset. The subpopulation selection is done using *ClusterMatch* algorithm and poisoning of images is done using label flipping method, which are discussed in detailed below.

3.3 *ClusterMatch* Algorithm

Algorithm 1 ClusterMatch Algorithm - automatically identify subpopulations

Input: $X \in D_{aux}$ - feature values; k_{cluster} - number of clusters; PreProcess - preprocessing function
centers = Cluster(PreProcess(X), k_{cluster})
target = PickCluster(centers)
return $\mathcal{F} = \lambda\ x : \mathbb{1}$(ClosestCenter(PreProcess(x), centers) == target)

ClusterMatch algorithm eliminates the need for annotation by identifying subpopulations of interest using clustering. By recognising natural clusters in the data, it is possible to compromise the model for a particular cluster but not for others. Presented in Algorithm 1, in *ClusterMatch* the attacker clusters and identifies the most vulnerable subpopulations using the auxiliary dataset D_{aux}. Before we can use *ClusterMatch*, there are some design considerations that must be addressed. We must describe a clustering algorithm and a preprocessing function for the supplementary data. We begin the preprocessing phase by utilising the representation layer of a neural network trained on D_{aux}, and then we apply a PCA projection. We utilise KMeans algorithm for clustering.

3.4 Label Flipping

We begin the production of poisoning attacks by adapting a common baseline procedure, label flipping, to our environment. In poisoning availability attacks Xiao et al. [2015], label flipping has been used to construct poisoning points with comparable feature values to legal data, but with a different label. If the subpopulation size is *m* and the adversary employs a poisoning rate α relative to the subpopulation, the adversary will add αm poisoned points, which should be negligible in comparison to the size of the overall dataset. In label flipping attacks, these points are generated by sampling *m* points that fulfil the D_{aux} filter function and adding them to the training set with a different label *t* than the original one *c*. We select a single label for the entire subpopulation in order to optimise the loss at the poison point. Label flipping assures significant target damage, while the filter function itself ensures minimum collateral damage - if the separation is strong enough, the learning algorithm will be able to learn the poisoned subpopulation independently, without affecting the rest of the distribution.

Overall, *ClusterMatch* is used on D_{aux} to find the worst subpoupation, then αm examples are poisoned using label flipping and finally model is trained on $D \cup D_p$.

4 NLP Models

There are many different types of models used for any natural language processing task, including the task of text classification that is considered in this paper. The types of deep learning models for text classification range from feed-forward networks to Transformers and everything in-between. Minaee et al. [2021] includes an overview of each category of models.

In recent years, there has been a boom of large-scale Transformer-base Pre-trained Language Models. Transformers [Vaswani et al., 2017] utilize self-attention mechanisms to enable parallel computation and model how each word influences other words in the same input sequence. The parallelization makes Transformers more efficient than previously used methods like RNNs and CNNs, which enables the existence of very large pre-trained models that are trained on large amounts of text corpora. The pre-training process is considered unsupervised or self-supervised learning and learns contextual text representation. Further supervised fine-tuning is required to utilize the pre-trained model for specific tasks, like text classification. Fined-tuned pre-trained models have become state-of-the-art models for many natural language processing tasks.

We consider three Transformer-based Pre-trained Language Models. BERT [Devlin et al., 2019] is a bidirectional autoencoding language model that is trained using the masked language modelling task. It is very popular, having achieved state-of-the-art for a variety of text tasks, and is the basis for many related models. XLNet [Yang et al., 2019] aims to combine the advantages of both autoregressive and autoencoding models by using a permutation operation during pre-training. Empirical results

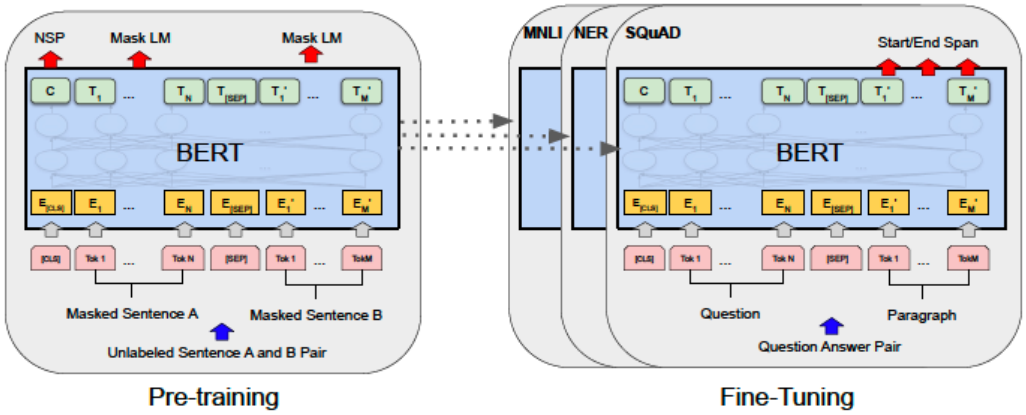

Figure 2: Overall pre-training and finetuning procedure for BERT [Devlin et al., 2019]. Apart from output layers, the same architecture is used for both pre-training and finetuning.

from Yang et al. [2019] shows that XLNet consistently achieves better results than BERT. Finally, ELECTRA [Clark et al., 2020] introduces a novel approach to training that produces a model with improved performance than other industry leading competitors while requiring a fraction of the compute power. The authors Clark et al. [2020] demonstrate ELECTRA's superior computational performance against XLNet and BERT.

4.1 BERT

Bidirectional **E**ncoder **R**epresentations from **T**ransformers (BERT) is a multi-layer bidirectional transformer encoder that uses self-attention. Most pre-trained language models that existed before BERT were unidirectional, which restricted their effectiveness. BERT overcame the unidirectionality problem by using a Masked Language Model (MLM) pre-training objective, also known as Cloze [Taylor, 1953]. To prevent the model from trivially predicting the target word during bidirectional conditioning, some percentage of the input tokens in the sequence would need to be masked. Now the model's task is to predict these masked tokens.

Pre-training for BERT consists of two unsupervised tasks. The first one is the masked language model pre-training objective mentioned above, the second is next sentence prediction. Next sentence prediction aims to capture the relationship between two sentences. When choosing sentences A and B for each training example, there is a 50% chance that B is next sentence following A. The other 50% chance is that B is some random sentence. After pre-training is complete, the BERT model can finetuned for downstream tasks. Figure 2 illustrates the pre-training and finetuning procedures.

One important note to consider is a problem referred to as pretrain-finetune discrepancy. This is due to the use of artificial symbols such as '[MASK]' during pre-training. However, these symbols do not exist in the real data used for finetuning, resulting in a discrepancy.

The popularity of BERT has been unmatched in recent years. It has become the basis for many new models such as RoBERTa [Liu et al., 2019], ALBERT [Lan et al., 2019], DistillBERT [Sanh et al., 2019], SpanBERT [Joshi et al., 2019], ELECTRA, ERNIE [Sun et al., 2019], and ALUM [Liu et al., 2020]. Despite the large number of newer, much more performant related models, BERT is still often used as a baseline model when comparing the efficacy of proposed MLM architectures.

4.2 XLNet

Transformer-based Pretrained Language Models can be split into two broad types, autoregressive language models and autoencoding language models. Autoregressive language models estimate the probability distribution of a text corpus by factorizing the likelihood into a forward or backward product. This results in it being only trained to encode a unidirectional context, which is a disadvantage as downstream finetuning tasks often require bidirectional context information. Autoencoding

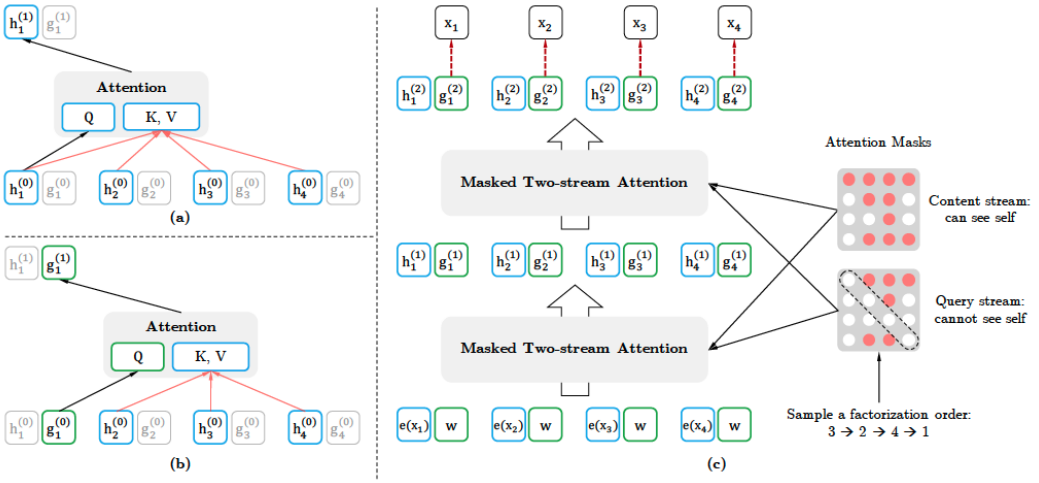

Figure 3: Overview of how XLNet [Yang et al., 2019] trains with the two-stream attention. (a): Content stream attention. (b): Query stream attention. (c): Training with two-stream attention.

language models work differently. The most famous autoencoding language model is BERT, which was introduced above.

Despite its popularity, BERT is not without disadvantages, it suffers from two main issues. The first is the pretrain-finetune discrepancy mentioned earlier. The second is the inability to model the joint probability as predicted tokens are masked in the input. Instead, BERT assumes predicted tokens are independent of each other given the unmasked tokens, which is an oversimplification and results in BERT not being able to capture high-order, long-range dependencies [Yang et al., 2019].

XLNet is a generalized autoregressive model that aims to combine the advantages of both autoregressive language models and autoencoding language models. To capture the bidirectional context that autoregressive models are missing, XLNet maximized the expected log likelihood of a sequence with regards to all possible permutations of a factorization order. Given a sequence of length T, there exists $T!$ different orders to perform autoregressive factorization. As model parameters are shared, information from the context is gathered from both sides, making the model bidirectional. As it is an autoregressive model and does not rely on data corruption, XLNet automatically avoids the disadvantages of autoencoding models.

To compute the representation necessary for the permutation language modeling object, a two-stream self-attention schema is needed. XLNet utilizes two sets of hidden representations. One is the context representation, which is analogous to the standard hidden states in Transformer. The context representation encodes both the context and the content. The other is the query representation, which only encodes the context and the position, but does not encode the content. Figure 3 provides an overview of how XLNet trains with the two-stream attention.

Given that XLNet combines the advantages of both autoregressive language models and autoencoding language models, we would expect better performance. Indeed, in Yang et al. [2019], it is shown that XLNet consistently exhibits better performance than BERT in a variety of problems, including our specific task of text classification.

4.3 ELECTRA

MLM pre-training methods, such as BERT, corrupt the input sequence by replacing a subset of tokens with a masked token and then train a model to reconstruct the original tokens. While in practice they present high quality results when transferred to downstream NLP tasks, a limitation of these models is they generally require large amounts of computation to be effective.

Proposed by Clark et al. [2020], ELECTRA is a novel pre-training model that presents a sample-efficient pre-training task called replaced token detection. Visualized in Figure 4, the author's model trains two transformer models representing a generator and discriminator. During training, instead

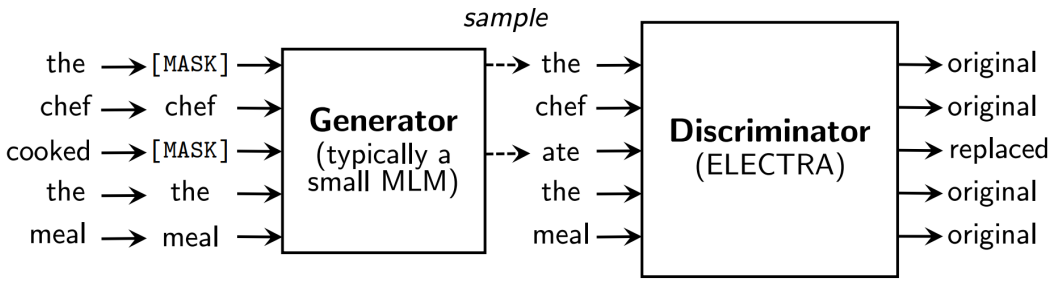

Figure 4: An overview of the replaced token detection architecture proposed by ELECTRA [Clark et al., 2020]. The generator can be any model that produces an output distribution over tokens, the author's utilized use a small masked language model that is trained jointly with the discriminator. After pre-training, only the discriminator is fine-tune on downstream tasks, becoming the ELECTRA model

of masking the input similar to BERT, ELECTRA corrupts the input sequence by replacing some tokens with plausible alternatives sampled from the generator network. Then, instead of training a model that predicts the original identities of the corrupted tokens, a discriminative model predicts whether each token in the corrupted input was replaced by a generator sample or not. The heuristic of omitting masked tokens from input sequences allows ELECTRA to alleviate the pretrain-finetune discrepancy problem.

The author's experimentation demonstrates their new pretraining task is more efficient than MLM since the task is defined over all input tokens rather than a small masked subset. Their analysis demonstrates the contextual representations learned by ELECTRA substantially outperform the ones learned by BERT given the same model size, data, and compute. The efficacy and performance superiority to BERT and XLNet are portrayed through their testing on popular NLP benchmarks. Within their experimentation on the GLUE dev set. Wang et al. [2018] the author's showcase ELECTRA achieving a score of 89.5 compared to XLNet's 89.1 and BERT's 79.8 while only using 25% compute to train. Similarly, on the SQuAD 2.0 [Rajpurkar et al., 2018] benchmark, ELECTRA presents a score of 91.4 compared to XLNet's 90.7 and BERT's 83.0 while consuming the same 25% compute resources. Overall, ELECTRA is a popular MLM model similar to BERT, however has been demonstrated to outperform both BERT and XLNet in practice while consuming significantly less computation resources. For our experimentation, ELECTRA was selected to observe the impact of subpopulation attacks present against other popular MLM models to determine if the results presented by Jagielski et al. [2020] on BERT extend into other common architectures.

5 Experiments

In this section we will begin by providing a brief overview of the dataset, various models, environment and hyperparameters used within our experiments.

We then explore the threat of the subpopulation attack on NLP models on our real world dataset. With the rise of popularity of pre-trained models in NLP (BERT [Devlin et al., 2019], ELECTRA [Clark et al., 2020], XLNet [Yang et al., 2019]) alongside the often high variance of data used to train them, they provide a target of interest for subpopulation attacks. We first explore the effectiveness of the label flipping attack in the transfer learning scenario, leveraging the *ClusterMatch* algorithm for subpopulation selection. This initial exploration is done over three poisoning modalities to demonstrate the generality and extensiveness of the attack. We then explore the scenario of bolstering our three transfer learnt models with a NLP defense in order to explore the potency of the subpopulation attack. We believe this depth of our experiments provides compelling evidence to support that subpopulation attacks are a versatile and effective threat model for poisoning attacks against NLP models.

Our attacks are evaluated using a similar generic approach proposed by Jagielski et al. [2020]. We partition our dataset into a training set D, auxiliary set D_{aux} and a test set D_t with each set dataset being disjoint. The adversary will be limited to having access to only D_{aux} to generate subpopulations, training a surrogate model for *ClusterMatch*. The adversary will then generate

poisoned data D_p where the model will then be trained on the union of $D \cup Dp$. Target damage will then be evaluated only on test points from D_t belonging to the target subpopulation.

5.1 Dataset

The dataset leveraged throughout our experiments is Stanford's IMDB Reviews dataset Maas et al. [2011]. IMDB Reviews contains 50000 reviews of popular movies left by users on the IMDB website alongside accompanying review scores. The dataset is predominantly used for binary sentiment classification, where a model is tasked with predicting whether or not a provided review is expressing positive or negative sentiment. The dataset provides a separation of 25000 reviews for training and 25000 for testing, where in our experimentation we split the training set into 12500 points for D and 12500 for D_{aux}.

5.2 Models

For our experiments we will be leveraging BERT, XLNet and ELECTRA. For all of these model's we will be fine-tuning all transformer blocks in addition to their classifiers. All models will be trained on vectors of 256 tokens for 4 epochs with a learning rate of 10^{-5} and a mini-batch size of 8. All models will utilize the AdamW optimizer [Loshchilov and Hutter, 2017], all using the same architecture and implementation from the Huggingface Transformers Library [Wolf et al., 2020]. BERT and ELECTRA are both composed of 12 transformer blocks whereas XLNet is composed of 24 transformer blocks. All model's contain an additional linear layer to be used in classification.

5.3 Environment

Throughout all model's training, subpopulation attack generation, execution and defense experimentation the same runtime environment was utilized. All experiments were performed on Google Collab, leveraging a A100-SXM4-40GB GPU with 89.6 GB of RAM in the server. Due to the exceptionally expensive computations required to execute the subpopulation attack, explained further in our limitations section, our experiments restricted the number of clusters considered by *ClusterMatch*. For each experiment 100 clusters were generated, with 30 being considered for experimentation. This was done to align with the experimentation environment performed by [Jagielski et al., 2020] in their investigation of BERT using the *ClusterMatch* algorithm with label flipping.

5.4 Attack Performance

To show feasibility and efficacy of the subpopulation attack on more NLP models, we apply the label flipping attack on BERT, XLNet and ELECTRA. The attack is performed using *ClusterMatch* with projection dimension of 10 for PCA and using KMeans for clustering. 100 clusters were found, of which 30 were sampled to perform the attack on. Ideally, we would have liked to perform the attack on all 100 clusters but was constrained by running time. The 30 cluster were chosen by sampling 10 clusters each at the lowest, medium, and highest confidence levels. Matching the experiments done in Jagielski et al. [2020], we evaluated each model for poison rate to be 0.5, 1.0 and 2.0.

Table 1 shows the results of the worst 10 subpopulation clusters. It groups the target damage into averages of the worst 10, 5 and 1 clusters. It also lists the worst collateral damage and average cluster size. The clean accuracy is evaluated from a model trained on the clean training set D. Based on the clean accuracy, we can see that XLNet is the most accurate model of the three, it is closely followed by BERT and then ELECTRA. There is an overall trend of higher poison rate resulting in higher target damage, which is expected. The exception to this pattern is the results from ELECTRA at poison rate 1.0. This volatility of results is discussed later.

The attack on all models seems to be ineffective at lower poisoning rates, where the average performance over the worst 10 clusters for poison rate 0.5 is 2.9%. The best performing model over the worst 10 clusters is XLNet with target damage 3.7%. As the poisoning rate gets larger, the efficacy of the attack significantly improves with the average performance over the worst 10 clusters for poison rate 2.0 reaching 15.9%. The best performing model over the worst 10 clusters is BERT with a target damage of 20.6%, however ELECTRA's worst cluster achieves the highest overall target damage at 52.8%. At the highest poisoning rate of 2.0, all three models were able to achieve a high target

Table 1: Clean and target damages for fully trained models on the IMDB dataset that have been attacked with Label Flipping using *ClusterMatch*. The worst 10, 5, and 1 subpopulations are recorded over poisoning rates of 0.5, 1.0 and 2.0. The collateral damage is recorded from the worst performing cluster on each poison rate. Cluster size refers to average size of cluster from the worst 10 subpopulations for each poisoning rate.

Model	Poison Rate (α)	Clean Accuracy	Target Damage			Worst Collateral Damage	Avg Cluster Size
			10	5	1		
BERT	0.5	0.913	0.024	0.035	0.051	0.004	167.0
	1.0		0.080	0.129	0.204	0.007	152.3
	2.0		0.206	0.303	0.506	0.010	126.3
XLNet	0.5	0.934	0.037	0.051	0.071	0.003	127.4
	1.0		0.060	0.080	0.098	0.006	121.3
	2.0		0.184	0.244	0.291	0.015	123.7
ELECTRA	0.5	0.898	0.027	0.049	0.099	0.005	113.6
	1.0		0.016	0.025	0.050	0.000	121.0
	2.0		0.088	0.152	0.528	0.007	119.8

damage rate, the lowest being XLNet at 29.1%. This shows the feasibility and efficacy of the attack on these NLP models.

Overall, collateral damage is very low at all poisoning rates. The highest collateral damage is 1.5% for XLNet at poison rate 2.0, with majority of the worst collateral damages being under 1%. This indicates that the subpopulation attack fulfills the goal of being concealed and advantageous to the attack staying undetected.

Figure 5 and Figure 6 show visualizations comparing the clean XLNet and ELECTRA models with ones attacked at poison rate 2.0 on the worst cluster. The top row plots the ground truth labels while the bottom row plots the predictions determined by the models. In comparison with the clean models that mostly predicted the target cluster accurately, the attacked model shows clear indication of misclassification for the target cluster. The prominence of the misclassification is especially noticeable on the attacked ELECTRA model. Despite the large number of points being misclassified in the target class, the overall decision boundary is still clearly visible in the attacked models and is similarly situated in comparison to the clean model. This supports the claim of a low collateral damage rate and shows that the attack is extremely concealed.

The results for BERT and ELECTRA were relatively similar, while the target damage results for XLNet was significantly lower. This is potentially related to the fundamental structure of the models. BERT and ELECTRA are both autoencoding models using MLM pre-training methods. BERT suffers from pretrain-finetune discrepancy, which ELECTRA avoids as the corruption of input tokens replicates the fine-tuning of downstream tasks. Both models assume predicted tokens are independent of each other given unmasked tokens. In contrast, XLNet is an autoregressive model that considers bidirectional context. It does not suffer from the same deficiencies as BERT and ELECTRA, which potentially indicates that the autoencoding nature of BERT and ELECTRA makes them more susceptible to the subpopulation attack.

One important observation to note is the volatility of these results. It is very noticeable that the target damage in the worst cluster is drastically different from the average of the worst 5 or worst 10 clusters. This indicates a large difference between the target damage of the worst cluster and the remaining 4 or 9 clusters, the difference being a drastic decrease in the target damage. Another example of this volatility is the abnormally low results from attack ELECTRA with poison rate 1.0, those results did not fit the pattern that was observed throughout the rest of the results. This volatility possibly stems from sampling 30 clusters instead of evaluating all 100 clusters. When sampling is used, there is guarantee that the worst clusters were sampled nor any guarantees on the quality of the

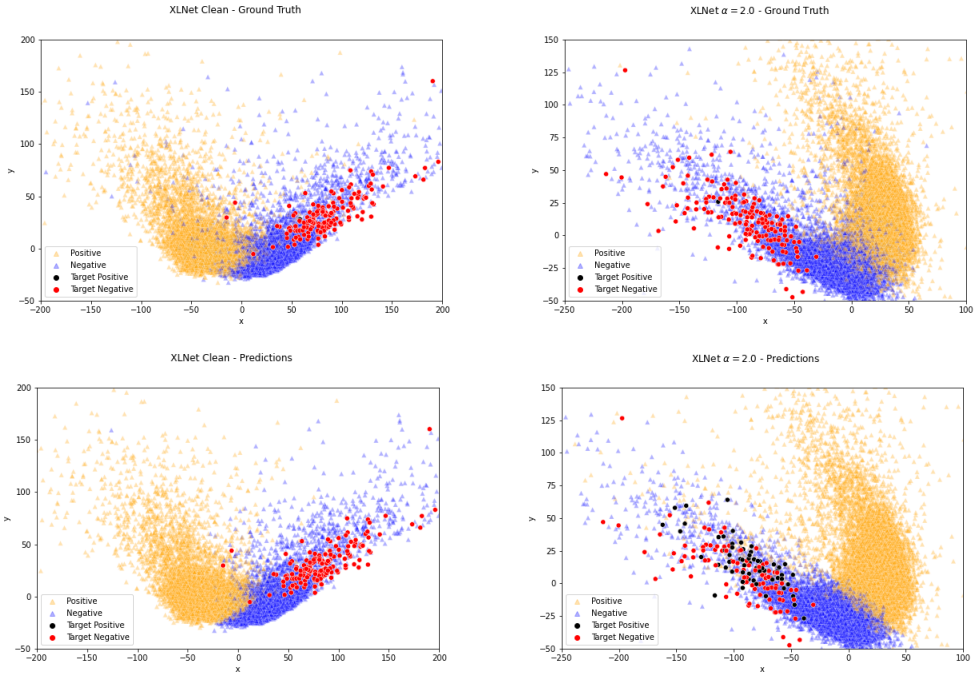

Figure 5: Clean XLNet model compared with one attacked at poison rate 2.0 on the worst subpopulation cluster. The top row plots the ground truth labels while the bottom row plots the predictions determined by the models.

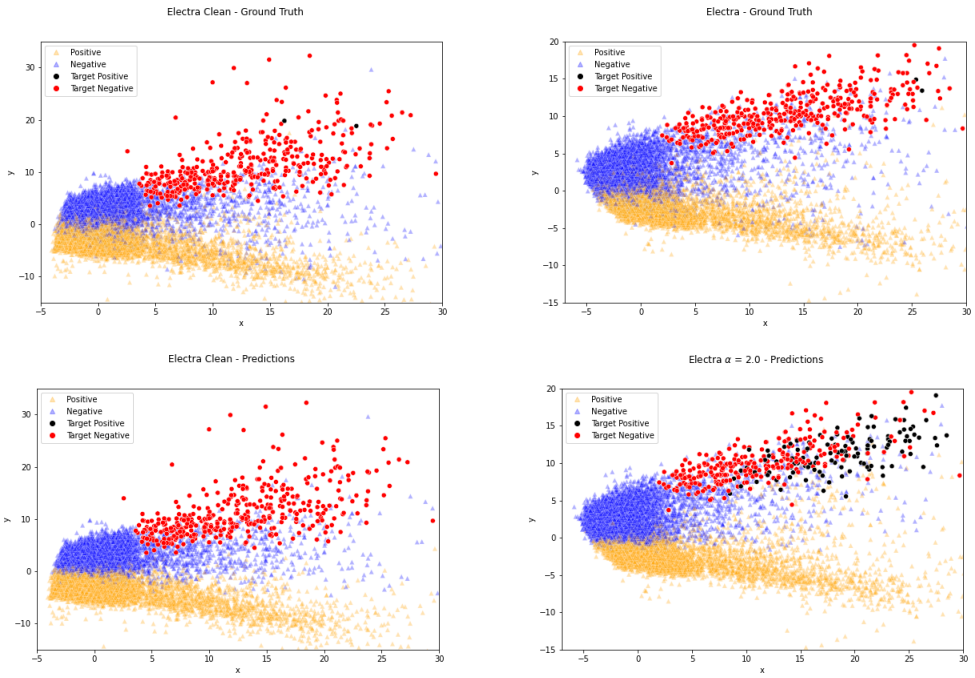

Figure 6: Clean ELECTRA model compared with one attacked at poison rate 2.0 on the worst subpopulation cluster. The top row plots the ground truth labels while the bottom row plots the predictions determined by the models.

Table 2: Target damages for fully trained models on the IMDB dataset that have been attacked with and without using the TRIM defence. The attacks were performed on the worst performing cluster with a poison rate of 2.0.

Model	Target Damage without TRIM	Target Damage with TRIM	Difference
BERT	0.506	0.419	-0.087
XLNet	0.291	0.085	-0.206
ELECTRA	0.528	0.530	$+0.002$

clusters sampled. The ideal would be to evaluate all 100 clusters, but the computational power and cost required limits our ability.

5.5 Effect of TRIM Defense

Algorithm 2 TRIM defense against availability attacks. Iteratively identifies poisoning by high loss values, and trains without those points.

Input: Training data D of n examples, loss function ℓ, attack count m, training algorithm A, maximum iteration count T
$\text{IND} = [n]$; $\text{IND\_PREV} = []$; $\textsc{Iterations} = 0$
while $(\text{IND} \neq \text{IND\_PREV} \wedge \textsc{Iterations} < T)$ **do**
  $\text{IND\_PREV} = \text{IND}$
  $\textsc{Iterations} = \textsc{Iterations} + 1$
  $f = A(D[\text{IND}])$ ▷ Train model on good indices
  $\textsc{Losses} = [\ell(x_i, y_i) | (x_i, y_i) \in D]$ ▷ Compute all losses
  $\text{IND} = \textsc{ArgSort}(\textsc{Losses})[: n - m]$ ▷ Lowest $n - m$ indices by loss
return f

To explore the versatility and effectiveness of the subpopulation attack, we now investigate to what extent existing defenses for poisoning attacks can be used to protect a model from subpopulation attacks. Proposed by Jagielski et al. [2018b], the TRIM defense was designed to defend a model from poisoning attacks by removing points with high loss from the training dataset. As seen in Algorithm 2, the defense works iteratively by identifying and removing outlying points at each step until convergence is reached. By detecting outlying points and removing them from the training dataset, the authors Jagielski et al. [2018b] demonstrated TRIM's effectiveness in restricting or eliminating poisoning availability attacks on transfer-learned models they defended.

For this section we consider the scenario of applying the subpopulation attack against the TRIM defense applied to all three of our models. Defenses for availability attacks ensure that poisoning does not compromise a model's accuracy significantly. The subpopulation attacks have been shown to have a modest impact on model accuracy in the previous section, focused on a selected target subpopulation. In theory, leveraging an availability attack defense for protecting against subpopulation attacks does show promise against restricting the efficacy of the attack. Our TRIM defense is set up in the same fashion used by Jagielski et al. [2020] in their investigation on BERT using a maximum iterative count of T=5. For all defense evaluations we employ the worst 1 subpopulation from XLNet and ELECTRA with poisoning rate = 2.0 to determine how TRIM may reduce target damage from the attacked subpopulation.

We present the performance of TRIM's defense at protecting against subpopulation attacks in Table 2. We see against BERT and XLNet that TRIM is able to reduce the target damage experienced by the subpopulation attack. On BERT we observe a 8.7% decrease in target damage with TRIM, reflecting the same results observed in Jagielski et al. [2020]. An interesting observation is XLNet experiencing the largest decrease in target damage with a 20.6% decrease, above double the impact as seen on BERT. In Figure 7, the target subpopulation between the undefended XLnet and TRIM defended XLNet is visually distinct. The major efficacy of the defense may be attributed to the target

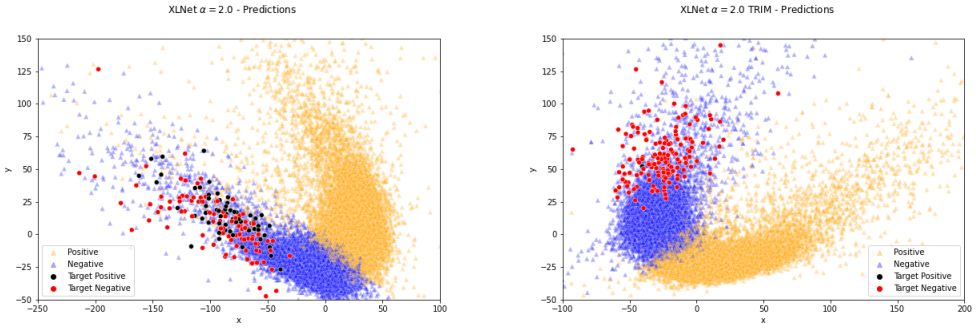

Figure 7: The predictions of the XLNet model (left) attacked with poison rate 2.0 on it's worst subpopulation cluster compared with the predictions of same attack applied to a TRIM defended XLNet (right).

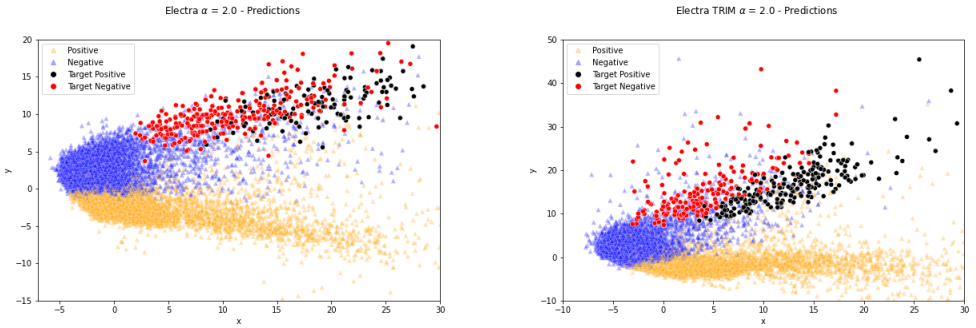

Figure 8: The predictions of the ELECTRA model (left) attacked with poison rate 2.0 on it's worst subpopulation cluster compared with the predictions of same attack applied to a TRIM defended ELECTRA (right).

subpopulation (red points) being located farther from the decision boundary between positive and negative data points. Though BERT and XLNet demonstrated the effectiveness of TRIM in reducing target damage, ELECTRA demonstrates an opposite effect. On ELECTRA's worst subpopulation for poison rate = 2.0 we observed a 0.2% increase in target damage. Though a negligible difference, this result illustrated a failure of defense observation theorized by the author's of Jagielski et al. [2020] where not all subpopulations are feasible to defend against. Within their discussions on defense impossibility the authors noted that if the subpopulation attack can cause a bias towards the attack class, it will be further exacerbated by the defense. Figure 8 illustrates this theory, where we observe the increased target damage that had occurred when ELECTRA with TRIM was poisoned. The shorter x and y range presented in the data points of ELECTRA compared to XLNet and BERT, alongside the density of the poisoned points presented a bias towards the attack class. The tight bias may have prevented TRIM from eliminating many outlier points, as the density of the subpopulation diluted the loss experienced from the true positive labels within the same region. This evaluation of TRIM demonstrates the efficacy of availability attack defenses against subpopulation attacks, while further showcasing the limitations these defenses may inherit due to subpopulation selection.

6 Conclusions

Although the subpopulation attack was previously shown to be effective against BERT, no other NLP models have been evaluated, making it hard to definitively claim that the subpopulation attack is effective against NLP models in general. We performed empirical analysis of the subpopulation attack on three prominent Transformer-based Pretrained Language Models: BERT, XLNet and ELECTRA. Of these, BERT is an autoencoding model and the most widely-used. XLNet is a generalized autoregressive model that captures bidirectional context. ELECTRA is an autoencoding model that alleviates BERT's pretrain-finetune discrepancy limitation.

Table 3: Amount of time necessary to perform the subpopulation attack. Training time per epoch is based on fully trained fine-tuning for 12500 examples and performed on an A100 GPU. Total attack time is based on performing and evaluating the attack on 30 clusters with each attack using 4 epochs.

Model	Training Time / Epoch	Total Subpopulation Attack Time
BERT	$\sim$ 2.5 minutes	$\sim$ 30 hours
XLNet	$\sim$ 3.5 minutes	$\sim$ 42 hours
ELECTRA	$\sim$ 2.0 minutes	$\sim$ 24 hours

Our experiments found that XLNet is much less susceptible to the subpopulation attack in comparison to the other two models that have comparable clean accuracy. Furthermore, XLNet was easily defendable with TRIM, mitigating the attack target damage significantly. TRIM proved to be ineffective or even counterproductive for BERT and ELECTRA. Although we were able to achieve a high target damage to all three models, XLNet's resilience to the subpopulation attack, and the ease of defense, lends itself to invoking more investigation into the application of the subpopulation attack on NLP models.

Furthermore, both BERT and ELECTRA, for whom the subpopulation attack was highly effectively against, utilizes masked language modeling methods. The shared weakness against the subpopulation attack potentially indicates that mask language modeling methods are naturally less resilient to the subpopulation attack. We also found that ELECTRA was much more volatile during our experiments. The results of poison rate α 1.0 for ELECTRA did not match the overall pattern we were observing from the other results. ELECTRA was also the only model where the TRIM defense resulted in an increase in target damage instead of a decrease.

To summarize, we made four contributions in this project. First, we performed empirical analysis on three prominent Transformer-based Pretrained Language Models, further exploring the application of the subpopulation attack to NLP models. Second, we obtained results showing the feasibility and efficacy of the subpopulation attack on XLNet and ELECTRA, obtaining high target damage for both models. Third, we showed that the TRIM defense was counterproductive for ELECTRA, but highly effective against the subpopulation attack for XLNet. Lastly, we found that the subpopulation attack was less effective against XLNet, an autoregressive model, than it was against the autoencoding models BERT and ELECTRA.

7 Limitations and Future Work

Throughout our investigation the limitation of compute time and resources was often a challenging issue. Presented in Table 3, every model sampled 100 clusters, computing 30 clusters to train on over 3 poison rates. With an average of $\sim$ 32 hours to train all models individually, ELECTRA's computationally efficient architecture performed the best while still requiring $\sim$24 hours to complete the complete subpopulation attack. This overhead cost of computation was minimizable during smaller dataset testing early on in our experimentation. When applied to the much larger IMDB dataset it was computationally expensive to our time and finances to run and re-iterate upon. Similar computational restrictions on the BERT model in Jagielski et al. [2020] further supports this observation. Future testing of the subpopulation attacks on NLP models would benefit from having much larger and dedicated resources.

An additional limitation present in our results is extending our attack and attack defense to different styles of sentiment classification datasets. Where IMDB was designed to address the binary classification of sentiment, our results may have benefitted from running our experiments on multi-class datasets, such as leveraging the Tweet Sentiment Extraction dataset. With larger compute resources available it may be feasible to perform our experiments on a plethora of sentiment analysis datasets.

Furthermore, future research into subpopulation attacks against NLP model's may benefit from evaluation of a wider depth of MLM and autoregressive based transformers. This may enable a deeper discussion into the vulnerability MLM models and autoencoding models may have against

the subpopulation attack. Additionally, examining a wider breadth of NLP models, beyond the transformer based models observed within our research, would extend insight into the impact subpopulation attacks have on other NLP architectures. Overall, we believe the results presented in our research presents a basis into observing further impacts subpopulation attacks have against NLP models.

References

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2013. URL https://arxiv.org/abs/1312.6199.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2014. URL https://arxiv.org/abs/1412.6572.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017. doi: 10.1109/SP.2017.49.

Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7, 2018. doi: 10.1109/SPW.2018.00009.

Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding, 2018. URL https://arxiv.org/abs/1808.05665.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines, 2012. URL https://arxiv.org/abs/1206.6389.

Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1689–1698, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/xiao15.html.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions, 2017. URL https://arxiv.org/abs/1703.04730.

Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks, 2018. URL https://arxiv.org/abs/1804.00792.

Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019. doi: 10.1109/ACCESS.2019.2909068.

Octavian Suciu, Radu Mărginean, Yiğitcan Kaya, Hal Daumé, and Tudor Dumitraş. When does machine learning fail? generalized transferability for evasion and poisoning attacks. In *Proceedings of the 27th USENIX Conference on Security Symposium*, SEC'18, page 1299–1316, USA, 2018. USENIX Association. ISBN 9781931971461.

Matthew Jagielski, Giorgio Severi, Niklas Pousette Harger, and Alina Oprea. Subpopulation data poisoning attacks. *CoRR*, abs/2006.14026, 2020. URL https://arxiv.org/abs/2006.14026.

R. Perdisci, D. Dagon, Wenke Lee, P. Fogla, and M. Sharif. Misleading worm signature generators using deliberate noise injection. In *2006 IEEE Symposium on Security and Privacy (SP'06)*, pages 15 pp.–31, 2006. doi: 10.1109/SP.2006.26.

Benjamin I.P. Rubinstein, Blaine Nelson, Ling Huang, Anthony D. Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J. D. Tygar. Antidote: Understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, IMC '09, page 1–14, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605587714. doi: 10.1145/1644893.1644895. URL https://doi.org/10.1145/1644893.1644895.

Battista Biggio, Konrad Rieck, Davide Ariu, Christian Wressnegger, Igino Corona, Giorgio Giacinto, and Fabio Roli. Poisoning behavioral malware clustering. In *Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop*, AISec '14, page 27–36, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450331531. doi: 10.1145/2666652.2666666. URL https://doi.org/10.1145/2666652.2666666.

Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35, 2018a. doi: 10.1109/SP.2018.00057.

Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2871–2877. AAAI Press, 2015. ISBN 0262511290.

Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1596–1606. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/diakonikolas19a.html.

Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5739–5748. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/shen19e.html.

Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC'19, page 321–338, USA, 2019. USENIX Association. ISBN 9781939133069.

James Newsome, Brad Karp, and Dawn Song. Paragraph: Thwarting signature learning by training maliciously. In Diego Zamboni and Christopher Kruegel, editors, *Recent Advances in Intrusion Detection*, pages 81–105, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-39725-0.

Alexander Turner, Dimitris Tsipras, and Aleksander Madry. Clean-label backdoor attacks, 2019. URL https://openreview.net/forum?id=HJg6e2CcK7.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2938–2948. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/bagdasaryan20a.html.

Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks, 2018. URL https://arxiv.org/abs/1811.00636.

Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks, 2018. URL https://arxiv.org/abs/1805.12185.

Bolun Wang, Yuanshun Yao, Shawn Shan, Huiying Li, Bimal Viswanath, Haitao Zheng, and Ben Y. Zhao. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 707–723, 2019. doi: 10.1109/SP.2019.00031.

Roei Schuster, Tal Schuster, Yoav Meri, and Vitaly Shmatikov. Humpty dumpty: Controlling word meanings via corpus poisoning, 2020. URL https://arxiv.org/abs/2001.04935.

Pang Wei W Koh, Kai-Siang Ang, Hubert Teo, and Percy S Liang. On the accuracy of influence functions for measuring group effects. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/a78482ce76496fcf49085f2190e675b4-Paper.pdf.

Jonas Geiping, Liam H. Fowl, W. Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches' brew: Industrial scale data poisoning via gradient matching. In *International Conference on Learning Representations*, April 2021. URL https://openreview.net/forum?id=01olnfLIbD.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv.*, 54(3), apr 2021. ISSN 0360-0300. doi: 10.1145/3439726. URL https://doi.org/10.1145/3439726.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555, 2020. URL https://arxiv.org/abs/2003.10555.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019. URL http://arxiv.org/abs/1909.11942.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *CoRR*, abs/1907.10529, 2019. URL http://arxiv.org/abs/1907.10529.

Yu Sun, Shuohuan Wang, Yu-Kun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. ERNIE: enhanced representation through knowledge integration. *CoRR*, abs/1904.09223, 2019. URL http://arxiv.org/abs/1904.09223.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. Adversarial training for large neural language models. *CoRR*, abs/2004.08994, 2020. URL https://arxiv.org/abs/2004.08994.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461, 2018. URL http://arxiv.org/abs/1804.07461.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822, 2018. URL http://arxiv.org/abs/1806.03822.

Andrew Maas, Raymond Daly, Peter Pham, Dan Huang, Andrew Ng, and Christopher Potts. Learning word vectors for sentiment analysis. pages 142–150, 01 2011.

Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. URL http://arxiv.org/abs/1711.05101.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. *CoRR*, abs/1804.00308, 2018b. URL http://arxiv.org/abs/1804.00308.