Making Adversarially-Trained Language Models Forget with Model Retraining: A Case Study on Hate Speech Detection

Marwan Omar
University of Central Florida
Orlando, FL, USA
marwan@knights.ucf.edu

David Mohaisen
University of Central Florida
Orlando, FL, USA
mohaisen@ucf.edu

ABSTRACT

Adversarial training has become almost the de facto standard for robustifying Natural Language Processing models against adversarial attacks. Although adversarial training has proven to achieve accuracy gains and boost the performance of algorithms, research has not shown how adversarial training will stand "the test of times" when models are deployed and updated with new non-adversarial data samples. In this study, we aim to quantify the temporal impact of adversarial training on naturally-evolving language models using the hate speech task. We conduct extensive experiments on the Tweet Eval benchmark dataset using multiple hate speech classification models. In particular, our findings indicate that adversarial training is highly task-dependent as well as dataset dependent as models trained on the same dataset achieve high prediction accuracy but fare poorly when tested with new dataset even after retraining models with adversarial examples. We attribute this temporal and limited effect of adversarial training to distribution shift of the training data which implies that models' quality will degrade over-time as models are deployed in the real world and start serving new data.

CCS CONCEPTS

• **Adversarial attacks**; • **Adversarial training**; • **Concept drift**; • **NLP Robustness**;

KEYWORDS

Hate speech detection, robustness, concept drift, adversarial training

ACM Reference Format:
Marwan Omar and David Mohaisen. 2022. Making Adversarially-Trained Language Models Forget with Model Retraining: A Case Study on Hate Speech Detection. In *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3487553.3524667

1 INTRODUCTION

Adversarial attacks have been studied in the Natural Language Processing (NLP) domain to expose weaknesses of language models. Adversarial attacks are input perturbations that aim to fool NLP models to make erroneous predictions thereby lowering their prediction accuracy. The adversarial examples can then be integrated with model training to improve the robustness of these models to future attacks[12]. Adversarial examples have been applied on numerous NLP tasks, including sentiment analysis, questions answering, and hate speech detection. In the hate speech detection space, for example, adversarial examples have been utilized to break hate speech detection classifiers and identify vulnerable spots within them. Then the same hate speech detection classifiers are retrained with adversarial examples to enhance robustness against real-world attacks as a defense. For example, Dinan et al. [6] conducted a research study on the importance of using adversarial attacks to break dialogue safety NLP models and then made those models more robust by incorporating adversarial training into the model retraining. Additionally, Kurita et al. [13] used adversarial examples to attack NLP toxicity classifiers through character-level perturbations and made the classifier produce erroneous predictions. They also studied the effect of adversarial training to defend against adversarial attacks. Both research studies leveraged adversarial examples in the training as a strategy to enhance robustness of the classifiers.

Adversarial training is a technique used to augment training data with adversarially-generated examples in each training loop. The idea of adversarial training was first introduced by Goodfellow et al. [9] in the image domain, where learning algorithms are used to generate adversarial examples. The adversarial examples can then be used to attack ML models and fool them to make erroneous predictions. The benefit of adversarial training is that the adversarially-generated examples can be used for retraining the language models (e.g. BERT and RoBERTa) to make models more resistant to adversarial attacks and boost the prediction accuracy.

Numerous research works in the NLP domain, and particularly for the hate speech task, have studied the operational aspects of adversarial training. For instance, Groundah et al. [10] used adversarial training to generate stochastically transformed versions of the original training samples thereby doubling the size of training data. The study concluded that adversarial training can help improve the classification accuracy of hate speech models against adversarial examples, however, it stops short in offering any insights on how to extend adversarial training to other NLP tasks and different model architectures. In a similar study, Tran et al. [21] utilized adversarial training to retrain a fine-tuned BERT model for the hate speech classification task. In their study, they demonstrated that building a hate speech classifier with regularized adversarial training yields performance gains and improves model robustness against adversarial attacks. The study was conducted on the BERT model only using the hate speech task so it is unclear if the findings

WWW '22 Companion, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9130-6/22/04…$15.00
https://doi.org/10.1145/3487553.3524667

can be extended to other language models and different NLP tasks (sentiment analysis, question anaswering, etc.).

Dinan et al. [6] integrated adversarial training into a "build it, break it, fix it" strategy whereby researchers built a toxicity detection classifier, broke the model with adversarial attacks, and then fixed the model by retraining the same model with examples collected from the adversarial attack phase. The authors demonstrate that adversarial training improves models performance and robustness against adversarial attacks. This study is somewhat unique in that it used humans in the loop to attack the model rather than using algorithms to generate attacks. However, the study falls short in discussing the long-term effectiveness of adversarial training and whether or not models remain robust under different model architectures with new dataset introduced via retraining.

Although the research community has demonstrated that adversarial training can boost machine learning models' accuracy and robustness against adversarial attacks, the current efforts stop short of understanding the caveats of adversarial retraining under realistic deployment scenarios, including the introduction of new training samples in the model due to retraining. Even though models may perform well after adversarial training, this does not imply that the same models will consistently perform when deployed to the real world. Adversarial training is essentially a snapshot of a model's performance. However, ML models (including NLP models) are prone to shift and drift as part of their normal evolution, and this in turn will likely cause the performance and accuracy to degrade over time, which we explore in this paper.

Contributions. In this paper, we make the following two contributions. First, we demonstrate that the effectiveness of adversarial training on language models degrades significantly when models when new data is introduced to the model via model retraining. An explanation of this degradation is the distribution shift in the input data, which shifts the model's parameters (in the tuning phase) in a way that alters its awareness of the adversarial examples incorporated in an adversarial training phase. We argue that adversarial examples do not actually contribute to model robustness in the abstract, but to the current snapshot, and their sensitive to model update due to previously unseen data from the real-world. Second, We empirically validate this shortcoming of adversarial training due to concept drift on hate speech detection, a popular natural language processing task. In doing so, we utilize various learning techniques and datasets. Among other intriguing observations, we found that the efficacy of adversarial training degrades upon introducing new learning techniques as well as datasets, which highlights the multi-dimensional set of factors affecting the robustness of such models.

Organization. The related work is outlined in §2 The background and preliminaries are outlined in §3. Methodology is outlined in §4. The results and discussion are highlighted in §5, followed by concluding remarks and open research directions in §6.

2 RELATED WORK

In the NLP space, numerous research studies have been conducted to demonstrate the effectiveness of adversarial training in defending NLP models against adversarial attacks [3, 5, 6, 8, 10–13, 19, 20, 22, 25]. Most of the research works conducted in this direction focus primarily on generating adversarial examples for hate speech detection models and then incorporate those examples into retraining of NLP models to robustify them against future adversarial attacks. The fundamental shorting of this strategy is that it considers a static view of NLP models and does not take into account the dynamic and evolving nature of models by introducing new data after adversarial retraining. This shortcoming becomes even more pronounced and critical once such models are deployed in the real-world and begin to incorporate previously unseen data in their fine-tuning. In such a case, the adversarial training becomes less relevant to the new data because the new data might carry different distribution characteristics than those pertaining to the data the model originally trained with.

Recently, Mou *et al.* [19] conducted an extensive study on hate speech detection models using CNNs and LSTMs, among other neural networks. As such, the authors used character- and word-level attacks under two settings: black-box and white-box, and demonstrated the effectiveness of their speech detection framework by comparing it to several state-of-the-art baselines. The authors showed that their framework contributes to enhanced robustness of hate speech detection models against future adversarial attacks. However, the study only considers static data, and does not assess the effectiveness of their framework with new data upon retraining.

Also related to the role and contribution of adversarial examples to robustness of hate speech detection models, Tran *et al.* [21] conducted a study using adversarial training along with source-ensemble-head-fine-tuning architecture to improve the robustness of hate speech classifiers against adversarial attacks. Although they conducted experiments and demonstrated that their framework outperforms other state-of-the-art models, e.g., BERT, they only use one dataset. To ensure transferability of a learning framework, it is crucial to test it using different learning tasks with various benchmark dataset, an issue not addressed in their work.

Zhou *et al.* [26] took a different approach and conducted a study to defend NLP classification models against adversarial attacks without retraining models. In their pursuit, the authors developed a novel framework for detecting and *blocking* adversarial attacks. Experimental results on benchmark datasets for a classification task showed that their novel learning framework outperforms state-of-the-art learning models in defending against adversarial attacks without incorporating adversarial examples. The study stops short of discussing the generalizibility of their findings to other classification tasks, such as hate speech detection.

We observe from our literature review that the vast majority of research works examine the effectiveness of adversarial training on hate speech classification from a static point of view, meaning that they consider learning models to be static. Moreover, the state-of-the-art in the literature does not further evaluate the impact of adversarial training on model robustness beyond training data and without extending the application to other learning tasks or domains. Our research is different in that we examine the near-/long-term impact of adversarial training on robustifying language models by introducing new datasets (which might be out-of-distribution from training data) as well as various learning algorithms.

3 BACKGROUND AND PRELIMINARIES

In this section, we will review the background and present the preliminaries, which shall guide our presentation in the rest of this paper. To begin our discussion, we review the primary natural language task of interest, hate speech detection. We then present various notations and definitions of adversarial training and adversarial examples, followed by a formal definition of the objective that may cause the difference in the outcomes of the model performance upon retraining, and the various metrics used to measure performance of NLP models.

3.1 Hate Speech Detection

Acknowledging that adversarial examples can be applied to numerous linguistic tasks, we utilize hate speech detection as our main task, due to the general prevalence of this linguistic task.

Formal Definition: Hate Speech Detection. Let $x \in X$ be an instance of input text, where X is the space of the input text, and y be the target prediction of task (e.g., for hate speech classification; offensive vs. non-offensive, y could be either 0 or 1[1]). Our learning task, that is the hate speech detection classifier is formally denoted as a mapping function $y \in \mathbb{R}^n$, where n is the number of classes being predicted. The hate speech classification task is denoted by a mapping function $h_\theta : \mathcal{X} \rightarrow \mathbb{R}^n$ (alternatively, $h_\theta : x \rightarrow y$), where θ is the model parameters (more on that is below).

In this definition, it is worth noting that hate speech detection as a classifier can be implemented using numerous learning algorithms to obtain h_θ, including deep neural networks, such as BERT and RoBERTa. Given the binary classification task we have (hate speech detection with two classes only: hate speech or non-hate speech) and in order to minimize $\ell(h_\theta(x), y)$ we can define a binary loss function (also refereed to as cost function) to reduce the entropy; that is to minimize the average loss across the training samples.

$$\min_\theta \frac{1}{m} \sum_{i=1}^{m} \ell(h_\theta(x_i), y_i) \tag{1}$$

3.2 Adversarial Examples

An adversarial example is an input designed (or crafted) to fool an NLP model and potentially lead to incorrect predictions [9][2]. Let $x\prime$ be an adversarial example if it can fool the classifier $h_\theta(.)$ and causes it to produce erroneous prediction (classifies offensive as non-offensive and vise-versa). Thus, an an adversarial example x' can be obtained by solving the following optimization problem:

$$\max_{x'} \ell(h_\theta(x'), y) = \max_{\delta \in \Delta} \ell(h_\theta(x + \delta), y) \tag{2}$$

That is, by solving this optimization our goal is to learn x' defined as $x + \delta$ for some perturbation δ obtained from the affordable perturbations, Δ, in a way that maximizes the loss between the prediction function h_θ and the original output y. We observe that the distribution of x typically determines $\delta \in \Delta$[3].

In this study, we are only concerned with adversarial examples on the textual inputs for the hate speech detection models. Acknowledging that numerous studies have addressed the optimization problem in (2). In the following, we consider one such work [12] for highlighting the background and the key elements employed in our analysis, although this part can be replaced by any adversarial generation algorithm as it is a secondary aspect to the contribution. The algorithm works in two major steps. First, starting with a sentence of n words, $x = w_1, w_2, \ldots, w_n$, the algorithm ranks those words based on their importance. Noting that only the important words in x are going to contribute more significantly to the outcome of h_θ, the authors argue that changing those important words will significantly affect the output of $h_\theta(x)$. Moreover, given that the modification will only affect those important words, it will, by definition, be minimal. Once the important words are selected, the second step of *word transformer* is invoked, where words are replaced with other words that have similar semantics, fit well in the context, and force the target model to misclassify the input sentence x. Upon executing the word transformer step, various sanity checks are imposed, including part-of-speech enforcement and semantic similarity check.

For evaluating our work, we use an easy-to-interpret metric, which is the accuracy, defined in the following.

Accuracy. This metric measures the overall prediction accuracy of the hate speech classifier. The accuracy is calculated as the number of correctly classified predictions made normalized by the total number of predictions.

The accuracy is not the only metric used for measuring the performance of hate speech detection under adversarial attacks, and other metrics may include the *attack success rate* maybe used. The *attack success rate* is calculated as the number of adversarial examples incorrectly predicted by the classifier to the total number of samples. This metric is useful to determine the effectiveness of adversarial attacks.

To determine the efficiency of our attack model, we use as a metric the number of queries the attack framework (textattack) made to the target model and fetch the predicted probability scores.

Perturbed word percentage. Is a metric calculated as the ratio of the number of perturbed words to the text length.

3.3 Adversarial Training

Adversarial training is the process of generating adversarial examples by making perturbations on the input text. The generated examples will then augment the training data to robustify NLP hate speech classifier against adversarial attacks. The adversarial training process is considered a combinatorial problem and is solved using heuristic search algorithms [17]. It involves generating an adversarial example $x\prime$ from the original example x prior to training the hate speech classifier, $h_\theta(.)$, on both x and $x\prime$ [24].

Definition: Adversarial Training. *We define $J(h_\theta)$ as a loss function (loss under the true distribution of the samples) and write down*

[1] In this formal definition, we use a binary outcome for simplification, although nothing in the rest of this paper prevents from using non-binary class labels.

[2] We note that the outcome of the adversarial attack may or may not be to alter the classifier's outcome; e.g., it could only reducing the classifier's confidence. We note that the results in this work are independent of the eventual outcome of the adversarial attack, and only use misclassification since it is easy to interpret. Extending the results for other adversarial objectives, such as confidence reduction, is trivial.

[3] This observation is fundamental in the sense that adversarial examples are only model dependent w.r.t. to the currently used data in the model. Changing the underlying training data may break such dependency.

this loss function formally as:

$$J(h_\theta) = \mathbf{E}_{(x,y)\in\mathcal{D}}[\ell(h_\theta(x), y)] \tag{3}$$

where ℓ denotes the loss function, $h_\theta(x)$ is the expected label for the input example x, $\mathcal{D}$ is the empirical distribution over training data, $\mathbf{E}$ is the expectation function, and y is the target label. Since we often are not provided the actual (accurate, complete) distribution of data (i.e., real-world data), we approximate the distribution by considering a set of data points $D = \{(x_i, y_i) \sim \mathcal{D}\}; i = 1, \ldots, m$, from which we can calculate the the empirical loss as follows:

$$J'(h_\theta, D) = \frac{1}{|D|} \sum_{(x,y)\in D} \ell(h_\theta(x), y) \tag{4}$$

Our objective in adversarial training process is to minimize $J'(h_\theta, D_x)$, where D_x is a set of training examples. Using the same notation above, the loss function can be generalized into an adversarial loss, which is defined as follows:

$$J_{\text{adv}}(h_\theta) = \mathbf{E}_{(x,y)\in\mathcal{D}}[\max_{\delta\in\Delta} \ell(h_\theta(x+\delta), y)] \tag{5}$$

Similar to the rationale highlighted for (4), we define the empirical adversarial loss as:

$$J'_{\text{adv}}(h_\theta, D) = \frac{1}{|D|} \sum_{(x,y)\in D} \max_{\delta\in\Delta} \ell(h_\theta(x+\delta), y) \tag{6}$$

As mentioned above, the goal of the adversarial training is to minimize $J'_{\text{adv}}(h_\theta, D)$ in (6).

3.4 Model Retraining

Given the static nature of NLP models (i.e., hate speech detection classifier), distribution changes are likely to cause degradation of model performance and thus lead to incorrect predictions. To cope with this issue, we retrain each model with samples of new training data using a new dataset. Retraining with new samples and fine-tuning models will address the risk of model deterioration with the passing of time. Given the previous notation, our aim with retraining will make the model resistant to changes in the input data distribution.

4 METHODOLOGY

For our experimental setup, We adopted three state-of-the-art pre-trained hate speech detection models namely: BERT from [16], CNERG from [1], and RoBERTa from [4]. Additionally, we use the Tweet Eval benchmark dataset provided by [4] which contains three datasets: irony, hate, and offensive.

Initially, we started by measuring the classification accuracy of each of the three models listed above using the three datasets separately. This is a necessary step in order to gain an understanding of each model's performance. We then attacked each model using the textattack framework provided by [18] (incorporating the attack strategy highlighted earlier). The textattack framework is used to test models' performance under adversarial attacks which provides a sense of how the model will perform when tested with new data. We further retrained all three models with the irony dataset in order to cope with the effects of adversarial attacks and ultimately robsustify models against attacks.

In the real world, NLP models will be serving data it has not seen before and therefore the mapping between the input and output will likely change causing the model to make incorrect predictions. This is due to data shift causing concept drift as discussed in the previous section. To simulate the impact distribution shift (i.e., data shift) on NLP models, we attacked each of the three models with two datasets: hate and offensive after they had been retrained with the irony dataset. Our aim was to see how each model architecture performs when tested with new dataset and to determine if hate speech detection is task-dependent. Its also interesting to see how the hate speech model transfers across multiple dataset.

4.1 Implementation Details

In conducting our experiments, we followed [27] and utilized the Projected Gradient Descent (PGD) algorithm on hate speech classifier with a softmax cross-entropy loss function. This classifier simply takes the input features X, multiplies them with a matrix of weights W and adds a vector of biases b afterwards. This will give us a score $x_i = W_i^t F + b_i$ for each i-th class in our classifier. We will then pass this score through a softmax activation function, $\texttt{softmax}(\boldsymbol{x}_i) = \frac{e^{\boldsymbol{x}_i}}{\sum_j e^{\boldsymbol{x}_j}}$. In order to evaluate the quality of our model's predictions on training data, we will use the softmax cross-entropy cost function $L = -\log(S_y)$.

We then performed the adversarial retraining, by feeding the models both the original data and the adversarial examples to cope with distribution shift and the subsequent drift in the model. We collected the adversarial examples curated from the Irony dataset which had been used in the attack phase to fool the three hate speech models. For generating adversarial attacks, we adopt the textattack framework introduced by [18] which divides the process of generating NLP adversarial examples into four parts: (1) a goal function which we aim to maximize in order to maximize the probability of a successful attack: $1 - P(y||x; \theta)$ where θ is the parameters and $P(y||x)$ is the model confidence of the output y given input x, (2) a search algorithm to search for the best input perturbations and find an adversarial example X_{adv} that can fool the classifier (we limit the search method to making 1,000 queries to the victim model for generation of adversarial examples), (3) a transformation module to perturb a text input from x to $x\prime$, (4) a set of constraints that filters out undesirable $x\prime$ to ensure that perturbed $x\prime$ preserves the semantics and fluency of the original x [24].

4.2 Data Shift and Associated Questions

In our implementation of the training process thus far, for both the baseline model training utilizing the optimization in (1) and the adversarial training depicted in (6), we assumed no data shift has happened and that our dataset is a static one, D. Even when training models with adversarial examples, perturbation δ will be of the same distribution of the dataset D. This, in turn, limits the efficacy of the adversarial training to the current distribution of data, and making the adversarial examples less relevant to training examples from other datasets. When NLP systems, such as hate speech detection models, are deployed to the real-world (production environment), however, the underlying model will be constantly serving new data. The newly incoming data may even be out of distribution (OOD) with respect to the original data used for training and evaluating the performance of the model (baseline; before deployment). Moreover, data shift will occur eventually, yielding

temporal changes to the underlying model consequently causing concept) drift [7].

Concept drift implies that the statistical properties of the target variable (i.e., model's prediction accuracy) can change over-time in unforeseeable ways [14]. In the context of NLP models, concept drift means that a model's performance is highly likely to deteriorate over time as it sees new data after deployment to production environments [14]. Going back to the notation used earlier in this section: the mapping between the input features and the target function $h_\theta : \mathcal{X} \rightarrow \mathbb{R}^k$ (alternatively, $h_\theta : x \rightarrow y$) will most likely change due to distribution shift causing concept drift. Let $P(x)$ be the probability of the input features x, and $P(y\|x)$ the conditional probability of the targets y given the input features x. Additionally, we will subscript the probabilities as P_t to refer to the training data, and as P_n to refer to the new data, that is the data the model has not yet seen (i.e., data the model will see after deployment to the real world). Formally, we can define model drift as $P_t(y\|x) \neq P_n(y\|x)$

Intuitively, when we train a model used to realize the goal in (1), we expect the model performance and prediction accuracy to be high since we would minimize model loss as part of this optimization). We then generate adversarial examples, as defined in (2), and expect the model loss $\ell(h_\theta(x+\delta), y)$ to rise significantly due to adversarial attacks. Addressing the high loss would require implementing adversarial training steps by solving the optimization in (6). We observe that adversarial examples are generated using D, an empirical distribution using a set of finite examples (x_i, y_i) for $i = 1 \ldots m$. An important question to address would be: how would the adversarial examples in $J'_{\text{adv}}(h_\theta, D)$ affect model loss if D is changed to D', updating h_θ without updating $J'_{\text{adv}}(h_\theta, D)$? In other words, by solving $\min_\theta J'_{\text{adv}}(h_\theta, D)$ for some D, then updating the dataset D to D', and solving for θ' in $\min_\theta J'(h_\theta, D')$, how big is the gap in the inner model loss? This is, how do we compute that gap as the following difference:

$$\frac{1}{|D'|} \sum_{(x,y)\in D'} \ell(h_{\theta'}(x), y) - \frac{1}{|D|} \sum_{(x,y)\in D} \ell(h_{\theta'}(x+\delta), y)$$

One can trivially answer the above question in a binary form (i.e., whether there will be a gap or not, since it is intuitive that the model loss will cause the performance to deteriorate when tested against old adversarial examples generated by taking D into account). However, our focus in this work is rather the quantification of this gap, which is both interesting and important.

5 RESULTS AND DISCUSSION

In the following, we present the details of our experiments conducted on various datasets using different model architectures and validate our assumptions about the limited impact adversarial training has on hate speech classification models due to data shift in the underlying models caused by the arrival of new data.

We first report the accuracy of the three target models on the original test data before the attack and refer to it as the original prediction accuracy. Then we measure the prediction accuracy of the target models under the adversarial attacks. By comparing these two accuracy scores, we can determine the effectiveness of our adversarial attacks, the more significant the gap between the original accuracy and the accuracy under attack, the more effective and successful the adversarial attacks are. Additionally, we also report the perturbed word percentage as the ratio of the number of perturbed words to the text length. Additionally, to determine the efficiency of our attack model, we use as a metric the number of queries the attack framework (textattack) made to the target model and fetch the predicted probability scores.

5.1 Datasets, Algorithms, and Baseline

5.1.1 Datasets. As shown in Table 1, we conducted our experiments using three datasets: Irony, Hate, and Offensive, all of which are popular datasets and are part of the benchmark dataset tweeteval [4] which is widely used in the hate speech domain.

5.1.2 Algorithms. We utilize three deep learning algorithms that have been shown to provide state-of-the-art classification results for hate speech: BERT-base-uncased-hatexplain-rationale-two [16], twitter-roberta-base-offensive [4], and Hate-speech-CNERG [1]. For consistency and clarity, in our results we refer to BERT-base-uncased-hatexplain-rationale-two as BERT, twitter-roberta-base-offensive as RoBERTa, and Hate-speech-CNERG as CNERG model.

5.1.3 Evaluation Metrics. We use the classification accuracy as a metric. The accuracy is calculated as the number of correct predictions made normalized by the total number of predictions.

5.2 Results

5.2.1 Baseline. We implement the above three algorithms for establishing a baseline of performance (using the classification accuracy) of the learning algorithms without any attacks. As we can see from Table 2, BERT consistently achieves the highest classification accuracy (92.63%, 89.12%, and 90.57% on the three datasets, respectively). Conversely, CNERG consistently achieves the lowest classification accuracy (79.17%, 74.34%, and 81.98% on the three datasets, respectively). The fact that BERT outperforms the other two models under different datasets is expected as BERT is a state-of-the-art model and this is consistent with findings in the literature.

5.2.2 Classification Accuracy Under Adversarial Attacks. We study the impact of the adversarial examples on the performance of the different models using the different datasets by employing the adversarial example generation method mentioned in section. For each run, we use 200 adversarial examples to attack each model.

Table 3 shows that the classification accuracy upon introducing the adversarial examples is significantly low. In Table 3, we show that the classification accuracy of our BERT model drops significantly (accuracy as low as 13.36%) when tested with adversarial examples using a new dataset (Hate dataset). To further validate the temporal changes of models and prove the transfer ability and generalization of our findings, we extend our experiments to different model architectures and observe that BERT and CNERG models suffer even a lower classification accuracy. The CNERG model fared worst with a classification accuracy of as low as 11.86% when tested with adversarial examples using the Hate dataset.

5.2.3 Retraining with Adversarial Examples. To generate AEs we utilize the idea of Adversarial Text Generation by [22], although using a different model architecture and for different datasets.

Table 1: Datasets: We employed the three real-world datasets: Irony, Hate, and Offensive for our hate speech task. Each dataset has binarized ratings and is set as positive and as negative. and split into a training, validation, and test sets

Dataset Name	Dataset Description	Atributes
Irony	tweet classification for irony task	set of 2862 for training and 784 for testing
Hate	tweet classification for hate task	set of 9000 for training, and 2970 for testing
Offensive	tweet classification for offensive task	set of 11916 for training and 860 for testing

Table 2: Baseline Experiment shows the classification accuracy of each of the three models prior to adversarial training. Note that BERT consistently achieves the highest classification accuracy. Conversely, CNERG consistently achieves the lowest classification accuracy

Dataset	Model	Accuracy
Irony	RoBERTa	87.39
	BERT	92.63
	CNERG	79.17
Hate	RoBERTa	86.98
	BERT	89.12
	CNERG	74.34
Offensive	RoBERTa	88.00
	BERT	90.57
	CNERG	81.98

Table 3: Classification accuracy under adversarial attacks. Note that the classification accuracy dropped significantly after attacking models with adversarial examples. We observe that BERT is consistently performing better than other models across multiple datasets

Dataset	Model	Accuracy
Irony	RoBERTa	14.94
	BERT	12.67
	CNERG	11.00
Hate	RoBERTa	11.98
	BERT	13.36
	CNERG	11.86
Offensive	RoBERTa	6.84
	BERT	7.98
	CNERG	4.58

Implementation Details. The adversarial examples generation model includes an encoder and a decoder for generating adversarial examples. The encoder and decoder are trained over a large text corpus to ensure that adversarial examples adhere to the linguistic constraints and preserve semantics [22]. To enforce semantic preservation, we follow the work of [17] and tighten the thresholds on the cosine similarity between embeddings of swapped words and between the sentence encoding of original and perturbed sentences. We ensure and enforce the grammaticality of the adversarial examples by validating perturbations with a grammar checker. Moreover, we apply the semantics as well as the grammatical constraints at each step of the search following [18] We conduct our experiments on real-world NLP datasets to demonstrate the effectiveness, applicability and generalizability of our approach. We show that our generated attacks are more diverse and more robust against model re-training and various model architectures. For the retraining, we adopted our three learning models (BERT, RoBERTa, and CNERG) to ensure the generalizability and transferability of our results to different network architecture and under multiple datasets.

Results. As shown in Table 4, retraining models with adversarial examples lowers the misclassification rate and improves the accuracy, which is consistent with the literature of adversarial training [13, 19, 23, 24]. In Table 5, we present the results of retraining each of the the three models (BERT, RoBERTa, and CNERG) with adversarial examples training. Specifically, we divide generated adversarial examples into two subsets, one is used for augmenting the training data, and the other is a hold-out set used for testing. With the augmented training data. We observe that adversarial training achieves accuracy gains consistently across all models and under different datasets.

Table 4: Adversarial example generation algorithm results. Note that the classification accuracy, after adversarial training, for each model consistently increased

Dataset	Model	Accuracy
Irony	RoBERTa	70.11
	BERT	78.35
	CNERG	62.98

Table 5: Classification accuracy under adversarial attacks after model retraining. Note that the classification accuracy dropped significantly after attacking models with adversarial examples upon model retraining. We observe that BERT is consistently performing better than other models across multiple datasets

Dataset	Model	Accuracy
Hate	RoBERTa	18.89
	BERT	21.67
	CNERG	17.45
Offensive	RoBERTa	12.92
	BERT	16.65
	CNERG	11.83

5.3 Comparative Analysis With SA Task

A key insight of this study is to highlight the impact of out-of-distribution data on the performance of adversarial training under different tasks. To this end, we have borrowed results from a previous work of ours demonstrating the impact of the ODD on the performance of AT using the sentiment analysis as the task of choice. Our objective was to determine if the performance of adversarially-trained language models would change under different tasks provided that we control the learning algorithm. To

answer that question, we have compared the performance of adversarial training under both the hate speech detection task as well as the sentiment analysis task. As we can observe from Table 6 that the performance of adversarially-trained model does actually change when the underlying language task changes (changing from the sentiment analysis to the hate speech task) upon exposing models to new data (i.e., OOD data). For instance, in the sentiment analysis task, the BERT-based model achieved only 15.75 prediction accuracy when tested with the MR dataset [15] compared to the same model in the hate speech detection task where the adversarial training lead to an accuracy of only 21.65 when tested with the Hate dataset. On the other hand, when the same model with tested with the Yelp dataset [2] under the sentiment analysis task, the prediction accuracy was only at 19.88 versus 16.65 for the hate speech detection task when tested with the offensive dataset. We observe from the above results that OOD data negatively impacts the performance of adversarial training on NLP models and that this impact may vary based, although slightly, on the learning task (e.g., sentiment analysis versus hate speech detection).

Table 6: Comparative Results. A-AT stands for accuracy upon adversarial training (percentage).

Dataset	Model	A-AT	Dataset	Model	A-AT
MR	BERT	15.75	Hate	BERT	21.65
Yelp	BERT	19.88	Offensive	BERT	16.65

6 CONCLUSIONS AND OPEN DIRECTIONS

In this paper, we demonstrate that adversarial training effectiveness is limited due to out-of-distribution data. We show that the performance of adversarial training deteriorates significantly with the passing of time as models serve new data. To validate our claims, we utilize the hate speech task on NLP models and run experiments on three real-world datasets (Irony, Hate, Speech) from the Tweet Eval benchmark dataset using three model architectures (BERT, RoBERTa, and CNERG). We observe that the prediction accuracy of models retrained with adversarial examples is almost as low as the ones without adversarial restraining. For instance, CNERG achieved a prediction accuracy of 9% while BERT's classification accuracy was as low as 14% when tested with the Hate dataset. As future work, we plan to extend this research and dive into the concept drift phenomena so that we may quantify and measure concept drift. It is intriguing to see how we can use statistical methods to measure the model's error-rate which might indicate how much a model has drifted after deployment to the real world.

The findings in this study call for further research into the impact of data shift and concept drift on the performance of NLP models after deployment into the real-world. While we addressed the impact of concept drift on the hate speech classification task, it is interesting to see how much of this insights can be applied to other linguistic task such as Natural Language Inference and Question Answering, or even other domains such as speech recognition and the vision domain. Designing solutions to the problem is also another direction. Such solutions might be as simple as reordering the training, increasing the model capacity, or sparing model variables to attend to the adversarial examples, which are all questions we will explore in our future work.

REFERENCES

[1] Sai Saket Aluru, Binny Mathew, Punyajoy Saha, and Animesh Mukherjee. 2020. Deep Learning Models for Multilingual Hate Speech Detection. *arXiv preprint arXiv:2004.06465* (2020).

[2] Nabiha Asghar. 2016. Yelp dataset challenge: Review rating prediction. *arXiv preprint arXiv:1605.05362* (2016).

[3] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th international conference on World Wide Web companion*. 759–760.

[4] Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. 2020. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*.

[5] Francesco Croce and Matthias Hein. 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*. PMLR, 2206–2216.

[6] Emily Dinan, Samuel Humeau, Bharath Chintagunta, and Jason Weston. 2019. Build it break it fix it for dialogue safety: Robustness from adversarial human attack. *arXiv preprint arXiv:1908.06083* (2019).

[7] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46, 4 (2014), 1–37.

[8] Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Workshop on abusive language online*. 85–90.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[10] Tommi Gröndahl, Luca Pajola, Mika Juuti, Mauro Conti, and N Asokan. 2018. All you need is" love" evading hate speech detection. In *Proceedings of the 11th ACM workshop on artificial intelligence and security*. 2–12.

[11] Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. 2019. Fermi at semeval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in twitter. In *International Workshop on Semantic Evaluation*. 70–74.

[12] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *AAAI conference on artificial intelligence*. 8018–8025.

[13] Keita Kurita, Anna Belova, and Antonios Anastasopoulos. 2019. Towards robust toxic content classification. *arXiv preprint arXiv:1912.06872* (2019).

[14] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. 2018. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering* 31, 12 (2018), 2346–2363.

[15] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Annual Meeting of the Association for Computational Linguistics*. 142–150.

[16] Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection. *arXiv preprint arXiv:2012.10289* (2020).

[17] John X Morris, Eli Lifland, Jack Lanchantin, Yangfeng Ji, and Yanjun Qi. 2020. Reevaluating adversarial examples in natural language. *arXiv preprint arXiv:2004.14174* (2020).

[18] John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *arXiv preprint arXiv:2005.05909* (2020).

[19] Guanyi Mou, Pengyi Ye, and Kyumin Lee. 2020. Swe2: Subword enriched and significant word emphasized framework for hate speech detection. In *ACM International Conference on Information & Knowledge Management*. 1145–1154.

[20] Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on twitter. *arXiv preprint arXiv:1706.01206* (2017).

[21] Thanh Tran, Yifan Hu, Changwei Hu, Kevin Yen, Fei Tan, Kyumin Lee, and Serim Park. 2020. HABERTOR: An efficient and effective deep hatespeech detector. *arXiv preprint arXiv:2010.08865* (2020).

[22] Tianlu Wang, Xuezhi Wang, Yao Qin, Ben Packer, Kang Li, Jilin Chen, Alex Beutel, and Ed Chi. 2020. Cat-gen: Improving robustness in nlp models via controlled adversarial text generation. *arXiv preprint arXiv:2010.02338* (2020).

[23] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. 2020. Adversarial weight perturbation helps robust generalization. *arXiv preprint arXiv:2004.05884* (2020).

[24] Jin Yong Yoo and Yanjun Qi. 2021. Towards Improving Adversarial Training of NLP Models. *arXiv preprint arXiv:2109.00544* (2021).

[25] Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv:1903.08983* (2019).

[26] Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019. Learning to discriminate perturbations for blocking adversarial attacks in text classification. *arXiv preprint arXiv:1909.03084* (2019).

[27] Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2019. Freelb: Enhanced adversarial training for natural language understanding. *arXiv preprint arXiv:1909.11764* (2019).