Probabilistic Robustness for Data Filtering

Yu Yu* and **Abdul Rafae Khan*** and **Shahram Khadivi** † and **Jia Xu***
* School of Engineering and Science, Steven Institute of Technology, NJ 07030, USA
† eBay Inc., Aachen 52064, Germany
yyu50@stevens.edu, akhan4@stevens.edu, skhadivi@ebay.com, jxu70@stevens.edu

Abstract

We introduce our probabilistic robustness rewarded data optimization (PRoDO) approach as a framework to enhance the model's generalization power by selecting training data that optimizes our probabilistic robustness metrics. We use proximal policy optimization (PPO) reinforcement learning to approximately solve the computationally intractable training subset selection problem. The PPO's reward is defined as our $(\alpha, \epsilon, \gamma)$-Robustness that measures performance consistency over multiple domains by simulating unknown test sets in real-world scenarios using a leaving-one-out strategy. We demonstrate that our PRoDO effectively filters data that lead to significantly higher prediction accuracy and robustness on unknown-domain test sets. Our experiments achieve up to +17.2% increase of accuracy (+25.5% relatively) in sentiment analysis, and -28.05 decrease of perplexity (-32.1% relatively) in language modeling. In addition, our probabilistic $(\alpha, \epsilon, \gamma)$-Robustness definition serves as an evaluation metric with higher levels of agreement with human annotations than typical performance-based metrics.

1 Introduction

Modern machine learning works with massive amounts of data on a range of tasks like language modeling, object detection, and data mining. Using large amounts of training set to build machine learning systems requires extensive computational resources and creates problems like domain shifts and input noise. These unfiltered training data harm model learning robustness (Frénay and Verleysen, 2013) that leads to prediction errors and serious consequences like self driving car fatality and medical misdiagnosis (Tian et al., 2018).

One problem causing this model instability is that the model learning is opt for the system's quality, which is typically evaluated by measuring how close this system's output of a test set is from its human label using metrics such as accuracy, error rate, perplexity, human evaluation score, and so on. However, such a system performance metric highly depends on the test set's choice and is thus unreliable. For instance, if our training set is drawn from the news domain, then the performance on a test set from the news domain (in-domain test set) is usually much higher than that from the Twitter domain (out-of-domain test set). As a result, in NLP, while some systems produce human parity results like the use of a pre-trained Transformer (Hendrycks et al., 2020) on in-domain test sets, these systems are easily corrupted by out-of-domain (OOD) samples from the real world.

Existing studies on data selection and robust learning demonstrate a need for test domain knowledge during training. Some data selection work (Moore and Lewis, 2010; Kirchhoff and Bilmes, 2014; van der Wees et al., 2017; Fan et al., 2017; Qu et al., 2019; Liu et al., 2019; Kang et al., 2020) chooses critical in-domain data for domain adaptation, and other work defends against adversarial attacks but offers little help for out-of-domain robustness (Taori et al., 2020) under natural distributional shifts (Wang et al., 2021) that occurs more frequently than extreme adversarial cases. This out-of-domain robustness is often measured by testing on a specific domain and a single task like sentiment classification (Müller et al., 2019; Hendrycks et al., 2020). The problem with these existing approaches is that the target domain knowledge is often unknown, as is the case for most real-world applications that do not know the domain of test data they will receive before launch.

To address these challenges, our goal is to select training data to achieve high accuracy on OOD test set, without requiring any target domain insight during data selection or model training process. We distinguish the out-of-domain and *unknown-domain test sets* by assigning the out-of-domain as the test domain known during the training, and the

Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 2950–2959

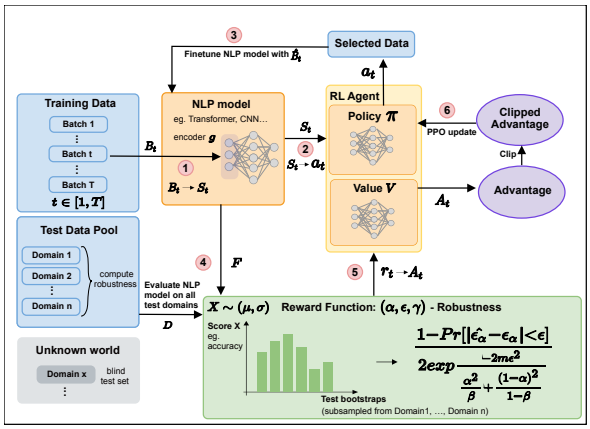

Figure 1: Probabilistic Robustness Rewarded Data Optimization (PRoDO) Framework.

unknown-domain as the test domain along with its information that are not known during the training. Practical applications often apply the latter case where we do not have any target domain knowledge, a condition we call "unknown-domain" robustness. To move our understanding forward, there is an urgent need to revisit out-of-domain robustness in "unknown worlds" to bridge the gap between laboratory observations and real-life results.

In our approach to the measurement of robustness on unknown domains, we define robustness as the consistency of the behavior of a machine learning system. The more a machine learning system's behavior deviates from the typical, the less robust the system is defined to be. Notice that this definition does not necessarily give a notion of whether system performance is good or bad. For example, in terms of sentiment analysis, this definition refers to consistency in prediction accuracy for a trained classifier.

To measure a system's performance consistency, we combine test sets for evaluation from various domains, like news, biomedical, and Twitter; randomly sample their subsets, and take each subset as this system's input and obtain an output. Then, we measure the performance of each output. If these output performances are close to each other, we say they are consistent. To quantify this consistency, we define our robustness metric as a notion of a probabilistic definition on the distribution of performance across different test domains, called **$(\alpha, \epsilon, \gamma)$-Robustness**, where the higher the probability of the consistent prediction performance, the more robust the system.

Our objective is to measure the probability of upper bounding the prediction accuracy gap between any test subset and its average. More specifically, we call an NLP system $(\alpha, \epsilon, \gamma)$-robust, if for every subset uniform-randomly drawn from a distribution, its prediction error, a combination of the target domain error and the source domain error weighted by the parameter α, is centered around the mean error, which is bounded through a parameter ϵ with a probability depending on γ, an indicator of how robust a system is, see definition in Figure 1 (in green) and details in Section 2.2.

In our approach, we *do not need any target-domain data* since we "simulate" unknown target-domain test sets using the *leave-one-out* error stability (Mukherjee et al., 2006). We assume that the non-left-out test sets are the simulated target domain while the left-out test is the real target domain for evaluation. For example, given biomecial as our unknown target domain to evaluate, we take the training data as the source-domain set and sample different subsets from a combination of other test domain data (e.g., news, TED talks, etc.) to simulate our target-domain test sets. The hyper-parameter α is the target-domain error weight and offers flexibility to balance the trade-offs between source- and target-domain errors.

The $(\alpha, \epsilon, \gamma)$-Robustness takes a new direction away from adversarial robustness to a general consistency of a model's quality as meta-evaluation methods that measure the consistency of user-defined quality metrics. Thus, *any standard performance evaluation metrics, such as accuracy, can be used within the definition of our robustness*.

After defining our robustness metric as our data optimization goal, we ask, *how should we select a subset of data that can maximize the robustness?* We assert that in general, the subset selection problem is computationally intractable. We conjecture that this condition holds true for every objective function, including our notion of probabilistic robustness, and is the reason why we use Proximal Policy Optimization (PPO) (Schulman et al., 2017) deep reinforcement learning to optimize the training set, which has the advantages of low variance, monotonic policy improvement, and sampling efficiency (Schulman et al., 2015; Uc-Cetina et al., 2021) compared to A2C (Konda and Tsitsiklis, 2000) and policy gradient (Sutton et al., 1998). Our Probabilistic Robustness rewarded Data Optimization (PRoDO) framework equipartitions the training data into mini-batches and simultaneously learns a policy network to select data iteratively

and a value network to estimate future returns using our $(\alpha, \epsilon, \gamma)$-Robustness as the reward functions illustrated in Figure 1 .

Our empirical results on sentiment analysis and language modeling show that the use of our robustness definition consistently and significantly enhances a model's out-of-domain performance. The main contributions of this work include:

1. Probabilistic Robustness definition of $(\alpha, \epsilon, \gamma)$-Robustness;
2. The creation of PPO Deep Reinforcement Learning framework for data selection;
3. The improvement of NLP model accuracy and out-of-domain generalization on showcase applications of sentiment analysis and language modeling.

The rest of the paper is as follows: In Section 2.1, we describe our PRoDO framework. In Section 2.2, we introduce the $(\alpha, \epsilon, \gamma)$-Robustness. Section 3 introduces experimental details including baselines, NLP tasks and ablation study. In Section 4, we discuss the previous literature on robustness in machine learning. Section 5 concludes the paper.

2 Method

Our goal is to enable our task model $\mathcal{F}$ (any Deep Learning-based NLP model, such as Transformer, etc.,) to achieve consistent while superior performance on any unknown test domain $\mathcal{D}_x$ whose distribution is different from the source training set $\mathcal{X}$, by learning an effective subset of $\mathcal{X}$ that can maximize the robustness of model $\mathcal{F}$.

The entire process details are depicted in Figure 1. Our method consists of reinforcement learning (RL), data selection, and training NLP models using the selected data. Specifically, we use reinforcement learning to train a data selection policy, and we use the data selection policy to select a subset of training data to fine-tune NLP models. Following Yu et al. (2022a), we pre-train the task model $\mathcal{F}$ on the full training data set $\mathcal{X} = \{x_i\}_{i=1}^{N}$, where x_i is a sentence, N is training set size. Then, the training set $\mathcal{X}$ is shuffled and randomly partitioned into T disjoint data batches such that $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$, with $\mathcal{B}_t = \{x_{(t-1)N|T+1}, x_{(t-1)N|T+2}, ..., x_{tN|T}\}$, where $N|T$ is the integer division of N by T, and $T \leq t$. For each batch, we select a subset of data $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | x_i \in \mathcal{B}_t\}$ with size o according to the data selection policy trained by reinforcement learning, and use it to fine-tune the model $\mathcal{F}$. In general, $\mathcal{F}$ and its encoder g are updated on $\hat{\mathcal{B}}_t$ for T times in an epoch, and each update is based on the previous checkpoint. Besides training set, we use a test data pool $\mathcal{D}$ containing n test domains $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n\}$ to simulate the real world scenario and compute the robustness score with it.

In the following sections, we will first introduce how our PRoDO framework learns to select data (Section 2.1) and then our reward function based on the probabilistic robustness definition (Section 2.2).

2.1 PRoDO framework

We now present the details of our Probabilistic Robustness Rewarded Data Optimization (PRoDO) framework.

2.1.1 RL training

The goal of our reinforcement learning agent is to learn an optimal data selection policy π to maximize the expected return $\mathcal{R}_t = \sum_{j=0}^{T-t} \gamma^j r_{t+j}$ from each state s_t, where the scalar reward r_t measures how good the action a_t taken by the policy is at the time step t and $\gamma \in [0, 1]$ is the discount factor. Specifically, each time step can be split into six steps as in Figure 1. Firstly, the encoder (e.g. an embedding layer in LSTM, or an encoder in transformer) inside the NLP model transforms the batch of raw data $\mathcal{B}_t$ into a batch of (document) embeddings, denoted as s_t. Secondly, the agent takes action a_t based on state s_t. The agent takes the state s_t as input and outputs a probability distribution for s_t, so that each sentence is associated with a probability, representing how likely it is going to be selected. The selected subset, denoted as $\hat{\mathcal{B}}_t$, is then obtained by Bernoulli sampling each sentence in the state s_t. The result of Bernoulli sampling is represented as an action vector a_t, where each value in it is either 0 or 1 representing each sentence in the batch not being or being selected. Thirdly, as soon as we obtain $\hat{\mathcal{B}}_t$, the NLP model $\mathcal{F}$ as well as encoder g are fine-tuned by the selected subset $\hat{\mathcal{B}}_t$. Then, the scalar reward $r_t = \mathcal{R}(\mathcal{D}, \mathcal{F})$ is calculated by our reward functions $\mathcal{R}$ (defined in Section 2.2) based on all available test domains D and current NLP model $\mathcal{F}$. Next, the advantage A_t over action a_t is computed by the difference of reward r_t and the output of value function $V(s_t)$. Finally, we update the policy function following the gradient with regard to the objective in PPO

(details in next section 2.1.2) using the advantage A_t.

2.1.2 PPO

In the conventional policy based (Sutton et al., 1998) or actor-critic (policy-value) (Mnih et al., 2016) based reinforcement learning, the precision of the value function often corrupts the policy optimization process for two reasons. First, some collected states might introduce noise to the prediction of $V(s_t)$, and thus lead to an inaccurate estimate of advantage A_t following with an inaccurate update of policy gradient. Secondly, a trajectory of interaction (consider a large dataset with a large interaction horizon $\mathcal{T}$) might take long time, while one collection of data can only be used to update the policy once, thus leading to severe sample inefficiency.

Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) and Proximal Policy Optimization (PPO)(Schulman et al., 2017) are proposed to solve the aforementioned problems by introducing importance sampling and advantage clipping. We adopt PPO as our framework since it is much simpler to implement than TRPO. PPO has the following properties, which are very desirable to achieve our goals: low variance, monotonic policy improvement and sampling efficiency (Grondman et al., 2012; Schulman et al., 2017). Our framework consists of policy and value networks jointly and dynamically learned together with the task model using the advantage error computed from the reward function, as shown in Algorithm 1. PPO uses $A(s_t, a_t)$, the advantage of action a_t in state s_t to scale the policy gradient. Specifically, the advantage of action (Mnih et al., 2016) a_t in state s_t is defined as

$$A(s_t, a_t) = \mathcal{Q}(s_t, a_t) - \mathcal{V}(s_t) \approx \sum_{j=0}^{T-t} \gamma^j r_{t+j} - \mathcal{V}(s_t), \tag{1}$$

where $\gamma \in (0, 1]$ is the discounting factor set as 0.99. $\mathcal{V}$ is the value function implemented as a value network.

Let $\mathbf{r}_t(\theta)$ denote the probability ratio $\mathbf{r}_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta\,old}(a_t|s_t)}$, the objective of PPO is defined in Schulman et al. (2017) as

$$A_{t_{clipped}} = clip(\mathbf{r}_t(\theta), 1-\epsilon, 1+\epsilon) A_t \tag{2}$$

$$\mathcal{J}(\theta) = \mathbb{E}_t \left[\min(\mathbf{r}_t(\theta) A_t, A_{t_{clipped}}) \right], \tag{3}$$

Algorithm 1 PRoDO Training Algorithm

Input: Epoch L, learning rate α, discount factor γ, training set $\mathcal{X}$, pre-trained task model $\mathcal{F}$(including encoder g), reward function $\mathcal{R}$ (discussed in section 3.2)
Output: selected data, fine-tuned $\mathcal{F}$, policy π_θ, data value estimator $\mathcal{V}_{\theta_v}$

Initialize data selection policy π_θ and value estimator $\mathcal{V}_{\theta_v}$
for episode $l = 1$ to L **do**
  Shuffle (uniformly at random) all training samples;
  Equipartition $\mathcal{X}$ into T (disjoint) sets with same size $n|T$: $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$;
  Initialize an empty list: episode history Υ
  for all $\mathcal{B}_t \in \mathcal{X}$ (uniform transition probability) **do**
    $s_t = g_t(\mathcal{B}_t)$;
    Obtain batch action a_t by sampling based on $\pi_\theta(s_t)$;
    $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | a_i = 1\}$, where o is selected sample size;
    Update task model $\mathcal{F}(g_t)$ by fine-tuning on $\hat{\mathcal{B}}_t$;
    $r_t = \mathcal{R}(\hat{\mathcal{B}}_t, \mathcal{F})$;
    Store (s_t, a_t, r_t) to episode history Υ;
  end for
  for all $(s_t, a_t, r_t) \in \Upsilon$ **do**
    Obtain $A(s_t, a_t)$ for each batch;
    Update policy weights θ and value estimator weights θ_v;
  end for
  Clear episode history Υ;
end for
return $\mathcal{F}$,π_θ and $\mathcal{V}_{\theta_v}$

where ϵ is a hyperparameter, and we set it as 0.2. The objective function clips the range of change of policy gradient into $[1-\epsilon, 1+\epsilon]$, which forces the new policy to not deviate too much from the old policy. This clipping design of PPO ensures monotonic improvement and thus it has the advantages of sample efficiency and ease of tuning compared to other policy-based algorithms.

The objective of value network is:

$$\nabla_{\theta_v} \mathcal{V}(\theta_v) = \mathbb{E}_{\pi_\theta} \nabla_{\theta_\mathcal{V}} (r_t - \mathcal{V}(s_t; \theta_v))^2 \tag{4}$$

The parameters of value function θ_v is updated by:

$$\theta_{v(t+1)} = \theta_{vt} + \alpha \nabla_{\theta_{vt}} (r_t - \mathcal{V}(s_t; \theta_{vt}))^2 \tag{5}$$

2.2 Reward function: $(\alpha, \epsilon, \gamma)$-Robustness

The reward function in Section 2.1 is the robustness of the NLP model $\mathcal{F}$.

In the real world, the out-of-domain data are often much less than the in-domain data. Formally, consider a pool of m samples where βm samples are drawn from *target* domain D_t and $(1-\beta)m$ samples are drawn from *source* domain D_s where $\beta \in [0, 1)$, β is often small in many scenarios. Thus, the minimization of empirical target error $\hat{\epsilon}_t$ becomes hard with the constraint of limited target data. To solve this, we instead try to minimize a convex combination of empirical source and target error:

$$\hat{\epsilon}_\alpha = \alpha \hat{\epsilon}_t + (1-\alpha) \hat{\epsilon}_s \tag{6}$$

where $\alpha \in [0, 1]$. We also denote ϵ_α as the weighted combination of the true source and target errors measured with respect to source domain D_s and target domain D_t, as shown in Figure 2.

According to the learning theory of Ben-David et al. (2010), the probability that the difference between the weighted empirical error $\hat{\epsilon}_\alpha$ and the weighted true error ϵ_α of an NLP system exceeds a given threshold has an upper bound as shown in Equation 7. Since we are more interested in bounding the difference between the weighted empirical error $\hat{\epsilon}_\alpha$ and the weighted true error ϵ_α to some threshold, so as to consider the system as "robust", we can transform the inequality to derive a lower bound of the probability as shown in Equation 9:

$$Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| \geq \epsilon] \leq 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}}} \quad (7)$$

$$\Leftrightarrow 1 - Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| \geq \epsilon] \geq 1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}}} \quad (8)$$

$$\Leftrightarrow Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| < \epsilon] \geq 1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}}} \quad (9)$$

The right hand side of Equation 9 is the lower bound of the probability that the difference of the empirical error and the true error of an NLP system is smaller than some threshold ϵ. We can introduce the robustness factor γ ($\gamma \in [0, 1]$) to the right hand side to control how tightly we would like to bound the probability of error difference:

$$Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| < \epsilon] \geq 1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}}} \cdot \gamma \quad (10)$$

$$\Leftrightarrow \gamma \geq \frac{1 - Pr[|\hat{\epsilon}_\alpha - \epsilon_\alpha| < \epsilon]}{2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}}}} \quad (11)$$

In consequence, we give the formal definition of $(\alpha, \epsilon, \gamma)$-robustness as:

Definition *We call an NLP system $(\alpha, \epsilon, \gamma)$-robust, if for any source domain D_s and target domain D_t, the difference between the empirical error $\hat{\epsilon}_\alpha$ and the true error ϵ_α is bounded through a threshold parameter ϵ with a probability of $1 - 2e^{\frac{-2m\epsilon^2}{\frac{\alpha^2}{\beta} + \frac{(1-\alpha)^2}{1-\beta}}} \cdot \gamma$, where $\alpha \in [0, 1]$ is the weight of target domain error and $\beta \in [0, 1)$ is the ratio of target data within all data.*

Based on this definition, we can interpret $\gamma \in [0, 1]$ **as the inverse indicator of robustness** for an NLP system. For example, a larger γ indicates a lower probability that the difference between empirical error $\hat{\epsilon}_\alpha$ and true error ϵ_α is within our expected threshold, so that the empirical error is probably with large variance, and thus the large γ indicates a less robust system.

2.2.1 Compute robustness by bootstrapping

We estimate robustness metrics using the *leave-one-out* error stability (Mukherjee et al., 2006) by excluding a left-out test set from all available datasets where we measure robustness. More precisely, for a given model, we randomly select one leaving-one-out test set and then combine all other tests to compute the robustness of the left-out datasets. Specifically, we consider the test domain as the target domain (an unknown domain that will not be used in training) and use all other test domains as available resources to compute mean and variance.

In practice, it often occurs that only limited test domains are available. Thus, the test scores of test domains are discrete values and hard to form a distribution. To solve this problem, we seek to a modified bootstrap algorithm to construct a pool of subsamples from the combined test set, as proposed by Yu et al. (2022b). For each subsample, instead of random sampling from the complete test pool, we randomly sample from the elements not present in the current subsample to minimize the intersection between each.

3 Experiments

3.1 Baselines

We compare our measure with four baselines. The first baseline denoted as **All** is the normal NLP model trained on all training samples. **Minmax** is derived from classic min-max objective in robust optimization. **Diff** (Zhang et al., 2022) and **Ratio** (Niu et al., 2020) are recently introduced robustness metrics defined by input perturbations.

Min-max robustness The notion of robustness can be originated from robust optimization (Ben-Tal and Nemirovski, 1998; Bertsimas and Sim, 2004) in which the optimization goal is to find a solution h satisfying

$$\min_h [\max_{\delta_1, \ldots \delta_n \in \Delta} \sum_{i=1}^{n} l(h, x_i + \delta_i)] \quad (12)$$

where x_i are the observed training samples, δ_i are deviations or perturbations from the observed samples, $l(;)$ is the loss function, and Δ are all possible

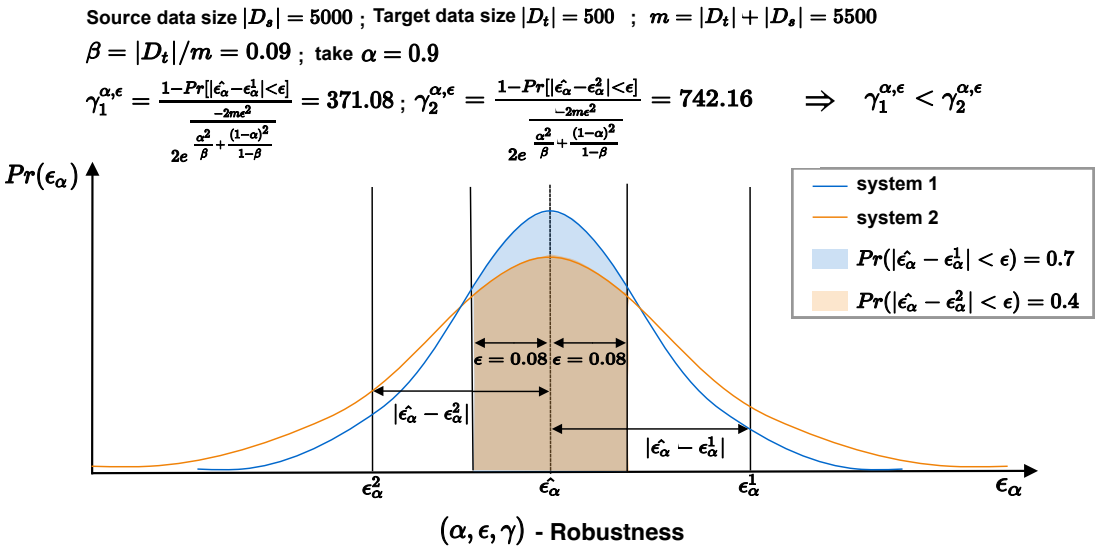

Figure 2: Illustrations of $(\alpha, \epsilon, \gamma)$-Robustness for two NLP systems. Orange curve and blue curve are the probability density function (PDF) of the test scores (X) of two NLP systems. The inference of robustness comparison is shown in top.

perturbations. Robust algorithms satisfying this definition are expected to minimize the empirical error under the worst possible perturbation. Since min-max objective targets the "worst possible perturbation", we consider the most challenging test set that will degrade the NLP performance most as "the worst possible perturbation" and use the evaluation score of such test set as a quantitative measure to denote model robustness under the min-max objective.

3.2 Improvements on NLP tasks

We experiment on a typical classification task of sentiment analysis and a generation task of language modeling.

3.2.1 Sentiment Analysis

In the task of sentiment analysis, we train a CNN classifier (Kim, 2014) using Amazon product review dataset (Blitzer et al., 2007). Specifically, we use the combined set of DVD, kitchen and books domains as source data. To compute robustness metrics, we use the full pre-processed Amazon product review dataset, which contains other domains such as grocery, tools, beauty and computer. We treat the test accuracy on the bootstrapped subsamples as a distribution and compute the $(\alpha, \epsilon, \gamma)$-Robustness by setting ϵ as 0.001 and α as 0.9. From Table 1, the classifiers optimized with $(\alpha, \epsilon, \gamma)$-Robustness outperforms all baselines. Specifically, the Min-max robustness achieves the second highest average accuracy score on test domains.

3.2.2 Language Modeling

Our baseline is a Transformer language model (Vaswani et al., 2017) with default hyper-parameters. We experiment with two moderate size datasets WikiText-2 (Merity et al., 2016) and Penn Treebank. As for evaluation, we report perplexity scores on four translation datasets from different domains, IWSLT'17 (TED talk) (Cettolo et al., 2012), Biomedical'21 (medical) (Yeganova et al., 2021), MTNT'18 (Reddit) (Michel and Neubig, 2018) and WMT'15 (news). The baseline models are trained using the fairseq toolkit (Ott et al., 2019) and stop training until the validation perplexity score does not improve for 5 epochs. The evaluation results are shown in Table 2. The perplexity (PPL) on all test domains have been improved. Specifically, test perplexity of MTNT'18 has an improvement of 28.05 (32.1% relative improvement) compared to the best baseline (Niu et al., 2020).

3.3 Ablation study

3.3.1 Comparison with human evaluation

We give a case study to compare the robustness of four language models trained on four different domains from the OPUS dataset (Tiedemann, 2012) using proposed robustness metrics. We combine MTNT'18, BIO'21, WMT'15 and IWSLT'17 test sets as a test pool, then subsample 50% of the test pool for 30 times. Next, we collect PPL scores on each subsampled bootstrap, and compute the mean and variance for the PPL score distribution. Fig-

	auto	beauty	food	instruments	office	computer	tools	phones	grocery	jewelry	outdoor	avg
All	56.48	54.03	56.02	59.15	56.84	55.17	55.95	52.79	53.06	56.63	55.81	55.63
Minmax	72.56	55.91	76.24	74.70	80.08	60.90	82.50	57.68	69.01	60.56	70.50	69.14
Diff	51.77	63.78	75.99	66.77	79.01	47.23	81.93	50.39	64.28	71.74	63.48	65.12
Ratio	71.06	63.92	75.25	82.01	67.45	63.19	78.57	58.83	68.79	65.53	62.80	68.85
$(\alpha, \epsilon, \gamma)$	**79.26**	**67.07**	**82.28**	**85.47**	**84.67**	**68.68**	**89.29**	**62.60**	**73.45**	**77.25**	**75.45**	**76.86**
+%	8.20	3.15	7.03	3.46	17.22	5.49	10.72	3.77	4.66	11.72	12.65	8.01

Table 1: Sentiment analysis accuracy [%] on amazon unprocessed domains. Last row: absolute improvement between $(\alpha, \epsilon, \gamma)$ and Ratio (Niu et al., 2020). Last column: average accuracy over all domains.

	WikiText-2				Penn Treebank			
	IWSLT	BIO	MTNT	WMT	IWSLT	BIO	MTNT	WMT
All	328.23	259.47	274.17	296.27	147.03	117.17	93.82	104.55
Minmax	189.03	140.35	160.26	169.03	142.84	82.55	91.59	101.91
Diff	193.06	136.94	167.55	168.90	140.18	79.04	92.62	98.70
Ratio	195.73	134.15	158.35	160.34	143.37	80.66	87.21	102.18
$(\alpha, \epsilon, \gamma)$	**175.91**	**123.47**	**142.34**	**154.38**	**118.96**	**71.45**	**59.16**	**87.70**

Table 2: Language modeling: Perplexity on four test domains. First row: source training domain; Second row: test domains.

ure 3 shows the normalized γ values against each corresponding epsilon value for the four language models. We can observe the γ values for the model trained by medical are much smaller than the values for other models, and the γ values for office are the highest. This shows that the model trained by medical is the most robust among all models, while the model trained by office is the least robust.

Model 1	Model 2	γ	Human	Agree?
edu	office	edu	edu	YES
edu	medical	medical	medical	YES
edu	books	books	books	YES
office	medical	medical	medical	YES
office	books	books	books	YES
books	medical	medical	medical	YES

Table 3: Robustness Metrics pair-wise comparison on each two models.

To evaluate how our robustness measures match human judgments, we compare the rank given by Figure 3 with the rank given by perplexity score and the rank given by human annotators, as shown in Table 4. Each human annotator is assigned with two language models selected at random. The human annotators do not know any details about the model or the training data used for each model. She/He can only use these models to get two generated sentences for the same input prompt. This step can be repeated as many times as possible until the human annotator decides which model is more *consistent* in its generations. The decision is based on consistency of generation quality but rather the quality itself. The results of pair-wise comparison

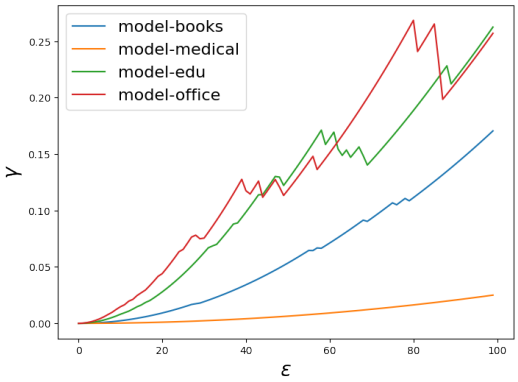

Figure 3: $(\alpha, \epsilon, \gamma)$-Robustness plot on four language models. "model-books" denotes the language model trained by the books domain from the OPUS dataset (Tiedemann, 2012)

between two language models are shown in Table 3. Our $(\alpha, \epsilon, \gamma)$-robustness perfectly aligns with decisions made by human annotators, while the rank based on perplexity score fails to match human evaluation results.

3.3.2 Domain distance

Many data selection work use target domain knowledge to select best in-domain data samples that are close to target domain (Aharoni and Goldberg, 2020; Liu et al., 2019; Ma et al., 2019). Our method does not use target domain knowledge, thus, we question whether our selected data resembles target domain under such zero-knowledge setting. We follow Aharoni and Goldberg (2020) using DistilBert (Sanh et al., 2019) to embed all domain data, Penn Treebank data and our $(\alpha, \epsilon, \gamma)$-Robustness

Model	CE	PPL	rank$_{CE}$	rank$_{PPL}$	rank$_{\gamma}$	rank$_{Human}$
edu	11.34	2474.16	1	1	3	3
books	12.03	4039.17	2	2	2	2
office	12.06	4181.39	3	3	4	4
medical	13.41	9426.04	4	4	1	1

Table 4: $(\alpha, \epsilon, \gamma)$-Robustness 100% agrees with human ranking, while perplexity (PPL) and cross entropy (CE) 25% agree with human. "office" means the model is trained on the office domain.

	BIO	MTNT	WMT	IWSLT
All	89.95	92.94	93.35	92.99
$(\alpha, \epsilon, \gamma)$	92.06	94.25	94.55	93.96

Table 5: Cosine similarity [%] between (row,column) data set. With zero-knowledge of four target domains, our method selects subsets of training data that are more close to the target domain.

selected PTB data into sentence embeddings and compute pairwise cosine similarity between the centroid of each domain. The result is shown in Table 5. With zero-knowledge of four target domains, our method selects subsets of training data that are more close to the target domain. Furthermore, we find the larger domain distance between the source domain and the target domain, the larger improvement will be, by computing the pearson correlation coefficient (0.83) between the domain distance and the relative improvement normalized by test set size.

4 Related Work

In previous years, a crucial direction of work on robustness of NLP models lies on the vulnerability of NLP models to input perturbations, such as crafted noises (Song et al., 2020; Boucher et al., 2022; Li et al., 2020; Schwinn et al., 2021). For instance, Cheng et al. (2018) proposes an adversarial stability training objective to enable neural machine translation models robust to input perturbations. Niu et al. (2020) evaluates robustness to input perturbations for neural machine translation. Compared to them, our approach handles new test inputs with distribution shifts from any unknown domains.

Some literature propose evaluation metrics for robustness from the perspectives of statistics or input perturbations (Weng et al., 2018; Niu et al., 2020; Mangal et al., 2019; Couellan, 2021). However, they either focus on the worst-case scenario of adversarial inputs or sampling single instances without considering the full distribution of the system performance.

5 Conclusion

We introduce probabilistic robustness rewarded proximal policy data optimization (PRoDO) framework to improve NLP model's generalization by selecting training data. Our framework is rewarded by the $(\alpha, \epsilon, \gamma)$-Robustness to measure an NLP model's performance consistency over multiple domains. Our experiments show the effectiveness of probabilistic robustness measure to enhance learning generalization and prediction accuracy. Our work also demonstrates a successful step towards general robustness evaluation and data selection without target domain insight.

6 Limitations

Time efficiency is one limitation of this work. For one thing, like other data selection work with reinforcement learning (RL), introducing RL requires convergence of the policy network and the value network, which takes quite a long time empirically. Referring to our time comparison results, our methods are roughly ten times slower than training with all source data directly. For another, our robustness measure requires the tuning process for hyperparameters α and ϵ, which also takes additional time.

Acknowledgments

We appreciate Amazon Alexa Prize, National Science Foundation (NSF) Award No. 1747728, and NSF CRAFT Award No. 22001 to fund this research.

References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. *arXiv preprint arXiv:2004.02105*.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning*, 79(1):151–175.

Aharon Ben-Tal and Arkadi Nemirovski. 1998. Robust convex optimization. *Mathematics of operations research*, 23(4):769–805.

Dimitris Bertsimas and Melvyn Sim. 2004. The price of robustness. *Operations research*, 52(1):35–53.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad characters: Imperceptible nlp attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1987–2004. IEEE.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. In *Conference of european association for machine translation*, pages 261–268.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. *arXiv preprint arXiv:1805.06130*.

Nicolas Couellan. 2021. Probabilistic robustness estimates for feed-forward neural networks. *Neural Networks*, 142:138–147.

Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. 2017. Learning what data to learn. *arXiv preprint arXiv:1702.08635*.

Benoît Frénay and Michel Verleysen. 2013. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869.

Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. *arXiv preprint arXiv:2004.06100*.

Xiaomian Kang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2020. Dynamic context selection for document-level neural machine translation via reinforcement learning. *arXiv preprint arXiv:2010.04314*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Katrin Kirchhoff and Jeff Bilmes. 2014. Submodularity for data selection in machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 131–141.

Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014.

Dianqi Li, Yizhe Zhang, Hao Peng, Liqun Chen, Chris Brockett, Ming-Ting Sun, and Bill Dolan. 2020. Contextualized perturbation for textual adversarial attack. *arXiv preprint arXiv:2009.07502*.

Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. 2019. Reinforced training data selection for domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1957–1968, Florence, Italy. Association for Computational Linguistics.

Xiaofei Ma, Peng Xu, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2019. Domain adaptation with BERT-based domain classification and data selection. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 76–83, Hong Kong, China. Association for Computational Linguistics.

Ravi Mangal, Aditya V Nori, and Alessandro Orso. 2019. Robustness of neural networks: A probabilistic and practical approach. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 93–96. IEEE.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Paul Michel and Graham Neubig. 2018. MTNT: A testbed for machine translation of noisy text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 543–553, Brussels, Belgium. Association for Computational Linguistics.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Sayan Mukherjee, Partha Niyogi, Tomaso Poggio, and Ryan Rifkin. 2006. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25(1):161–193.

Mathias Müller, Annette Rios, and Rico Sennrich. 2019. Domain robustness in neural machine translation. *arXiv preprint arXiv:1911.03109*.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. *arXiv preprint arXiv:2005.00580*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W Bruce Croft. 2019. Learning to selectively transfer: Reinforced transfer learning for deep text matching. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 699–707.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Leo Schwinn, René Raab, An Nguyen, Dario Zanca, and Bjoern Eskofier. 2021. Exploring misclassifications of robust neural networks to enhance adversarial attacks. *arXiv preprint arXiv:2105.10304*.

Liwei Song, Xinwei Yu, Hsuan-Tung Peng, and Karthik Narasimhan. 2020. Universal adversarial attacks with natural triggers for text classification. *arXiv preprint arXiv:2005.00174*.

Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. 2020. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599.

Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*, pages 303–314.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pages 2214–2218.

Victor Uc-Cetina, Nicolas Navarro-Guerrero, Anabel Martin-Gonzalez, Cornelius Weber, and Stefan Wermter. 2021. Survey on reinforcement learning for language processing. *arXiv preprint arXiv:2104.05565*.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Xuezhi Wang, Haohan Wang, and Diyi Yang. 2021. Measure and improve robustness in nlp models: A survey. *arXiv preprint arXiv:2112.08313*.

Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. 2018. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv preprint arXiv:1801.10578*.

Lana Yeganova, Dina Wiemann, Mariana Neves, Federica Vezzani, Amy Siu, Iñigo Unanue, Maite Oronoz, Nancy Mah, Aurélie Névéol, David Martinez, et al. 2021. Findings of the wmt 2021 biomedical translation shared task: Summaries of animal experiments as new test set. In *Sixth Conference on Machine Translation*.

Yu Yu, Shahram Khadivi, and Jia Xu. 2022a. Can data diversity enhance learning generalization? In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4933–4945, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yu Yu, Abdul Rafae Khan, and Jia Xu. 2022b. Measuring robustness for NLP. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3908–3916, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Yunxiang Zhang, Liangming Pan, Samson Tan, and Min-Yen Kan. 2022. Interpreting the robustness of neural nlp models to textual perturbations. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3993–4007.