Can Data Diversity Enhance Learning Generalization?

Yu Yu[*] 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, skhadivi@ebay.com, jxu70@stevens.edu

Abstract

This paper introduces our Diversity Advanced Actor-Critic reinforcement learning (A2C) framework (DAAC) to improve the generalization and accuracy of Natural Language Processing (NLP). We show that the diversification of training samples alleviates overfitting and improves model generalization and accuracy. We quantify diversity on a set of samples using the max dispersion, convex hull volume, and graph entropy based on sentence embeddings in high-dimensional metric space. We also introduce A2C to select such a diversified training subset efficiently. Our experiments achieve up to +23.8 accuracy increase (38.0% relatively) in sentiment analysis, -44.7 perplexity decrease (37.9% relatively) in language modeling, and consistent improvements in named entity recognition over various domains. In particular, our method outperforms both domain adaptation and generalization baselines without using any target domain knowledge.

1 Introduction

We introduce the Diversity Advanced Actor-Critic reinforcement learning framework (**DAAC**) to improve the generalization and accuracy of Natural Language Processing (NLP). Training data plays a crucial role in Deep Learning (DL)-based NLP. Investigations (Hendrycks et al., 2020; Dodge et al., 2020; Ramponi and Plank, 2020) show that training data quality is imperative to the machine learning model's performance. Good data are effective and easy to generalize, while bad data bring noise and overfits on out-of-domain test sets.

Decades of work (van der Wees et al., 2017; Aharoni and Goldberg, 2020; Guo et al., 2020; Axelrod et al., 2011) have been devoted to finding the best data, with the most commonly used method we call domain adaptation (Qu et al., 2019; Liu et al., 2019) that uses target domain knowledge. Despite large improvements on specific domains, these adapted models lack robustness and are prone to rare events. For example, when models shift towards one target domain like Twitter, they may become erroneous for other domains like medical or travel, and real-world applications on news report data, for example, do not know which queries to receive before they launch, resulting in a critical performance gap between laboratory findings and reality. Therefore, we move away from the independent and identically distributed assumption (i.i.d.) and focus on reducing generalization errors by studying the inter-dependencies among samples.

In this paper, we ask three questions (RQs) and propose our answers. The first question (**RQ 1**) is: *"does higher diversity among training samples lead to better generalized NLP model learning?"*. Semantic diversity is a desirable aspect in human annotation, where training data sets should be large and diverse enough to learn the many ways the objects differ (Shankar et al., 2017; Wu et al., 2018; Merler et al., 2019; Schumann et al., 2021), like intrinsic facial diversity in face recognition data, since every face is different. We show that diversity ensures that the training data can provide sufficiently discriminative information for the model (Gong et al., 2019), and thus give a more accurate prediction. Intuitively, for a fixed number of samples, the more semantic meaning they cover, the more information they contain, and thus the more effective they are for learning. We will use two examples to elaborate on our rationale.

Let us first take one explanatory example of $V =${be, cheerful, happy, stay}, a four single-word dataset that forms two semantically close clusters: $A =${be, stay} and $B =${happy, cheerful}. Our hypothetical NLP task is to generate sentences from V. From $\binom{4}{2}$ word combinations, four sentences are meaningful: "stay happy", "stay cheerful", "be happy", "be cheerful". Our dataset selection task is to find two single-word samples from V that generate the most sentences. If we select both single-word samples from A, which are semanti-

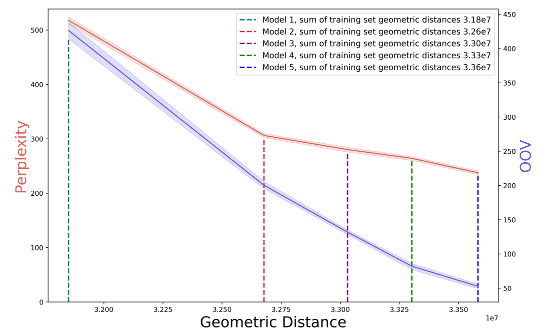

Figure 1: Higher training set diversity (geometric sum of pairwise Cosine distance), better learning generalization, w.r.t. PPL/OOV on a random test set.

cally close, we will generate no meaningful sentence (analogously that from B). However, we can generate one of the four meaningful sentences if we select one word from A and the other word from B, e.g., “stay” and “happy”, which are semantically distant. In fact, since these four sentences can be treated as paraphrasing, generating any of them will roughly cover the semantic meaning for all. In this example, the diversified subset $\{stay, happy\}$ is more effective for learning generalization.

Our second example is conducted on a real dataset. Figure 1 shows our experimental results on the Penn Treebank dataset. We equipartition 42K training sentences into five subsets and compute their diversity using the sum of pairwise Cosine distances of sentence embeddings (i.e., our *first method*, see next paragraph and Section 2.2.1). We sort the subsets according to their diversity scores. The first subset has the lowest diversity ($3.18e^7$), the second subset has the second-lowest diversity ($3.26e^7$), etc. We then train five language models on each subset[1], and computes the perplexity (PPL) and out-of-vocabulary (OOV) on ten randomly sampled subsets of the test set. PPL and OOV are typical indicators of how training data is generalized (Rastogi et al., 2020; Müller et al., 2019). Our results show that the diversity of the subset strongly negatively correlates to both PPL and OOV rate, where the subset with the highest diversity is more likely to generalize on a random test and has fewer unseen words.

With the preliminary evidence showing that a training set with higher diversity leads to more generalized learning, we propose our second question (**RQ 2**): *how to measure the diversity of a sample set?* We introduce three methods to quantify training dataset diversity. The *first method* is based on the notion of dispersion, also known as Max-Sum Diversification (Cevallos et al., 2016). By viewing each training sample as a data point in the sentence embedding space, we maximize the dispersion of the training set, which is the sum of pairwise Cosine distances of sentence embeddings, as in Figure 2 (2). The larger the sum, the more informative the subset is.

The pairwise geometric distance method has a worst-case complexity of $\mathcal{O}(n^2)$, and to accelerate the computation, we introduce our *second method* (Figure 2 (3)) that measures diversity using the volume of a convex hull around the samples, namely the volume that encloses all vertices, taking each sentence embedding as a vertex. Its quickhull (Barber et al., 1996) implementation has an average-case complexity of $\mathcal{O}(n \log n)$ and a worst-case complexity of $\mathcal{O}(n^2)$.

The above two methods compute diversity based on distances in the metric space. We can obtain global information content by measuring the uncertainty of the semantic distances between samples based on frequencies. Our *third method* combines our view of geometric distance and entropy in NLP. As depicted in Figure 2 (4), we apply Graph entropy, $H(G, P)$, an information-theoretic function on graph G with a probability distribution P on its vertex set (Dehmer and Mowshowitz, 2011; Rezaei, 2013). Here, each sentence in the subset is a node in the graph, and the Cosine distance of the two-sentence embeddings is the weighted edge between the two nodes. Empirically, all these three diversity measures consistently outperform the baselines. The Max Dispersion (MD) and Graph Entropy (GE) are more robust in various domains with the best performance overall because they account for every pair of distances in the training set.

After we have defined our diversity measure, we ask our third question (**RQ 3**): *“how can we optimize the subset to maximize the information content?"* Subset selection is NP-hard in general (Qian et al., 2016; Davis et al., 1997). We conjecture that this holds true for every objective function, including the notion of diversity introduced in this paper, and is the reason why we use **actor-critic** reinforcement learning (Konda and Tsitsiklis, 2000). We equipartition the training data into mini-batches

[1]The language model is trained on Transformer with 2 heads, 2 hidden layers, 200 embedding size, and 200 hidden units on each subset for 10 epochs. Each test set subset contains 1000 sentences.

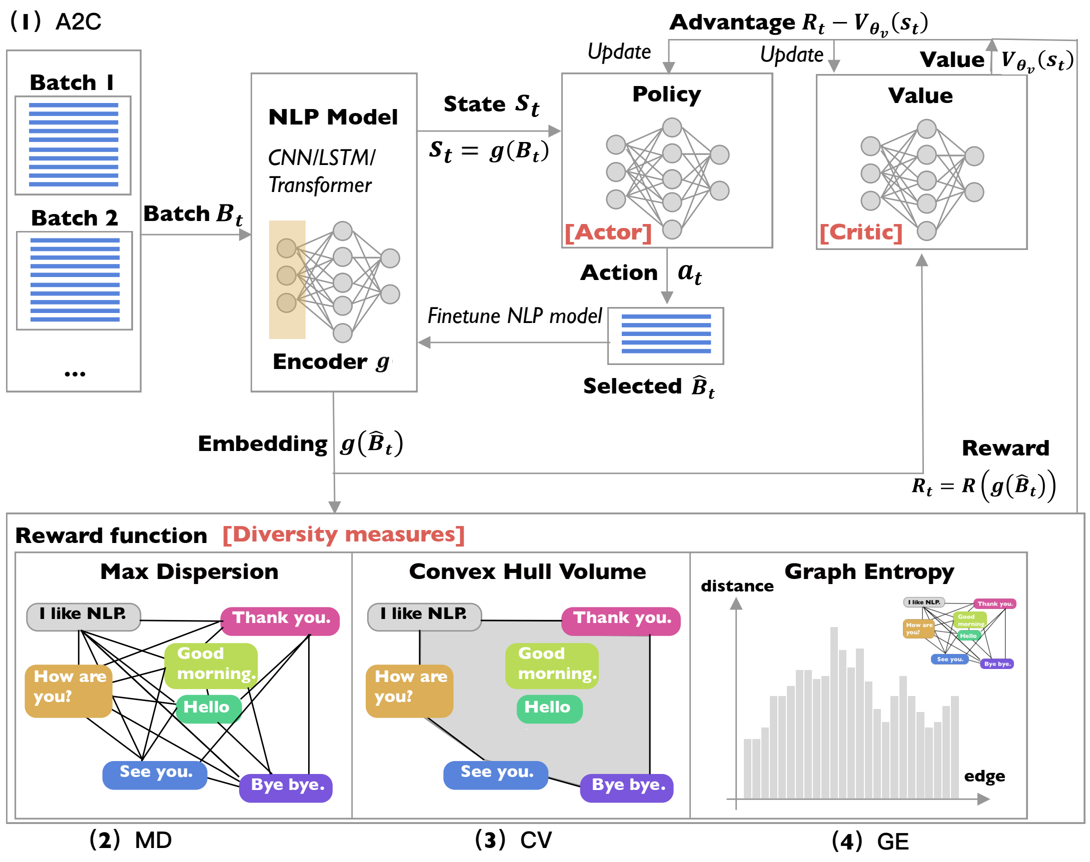

Figure 2: Diversity Advanced Actor-Critic Framework.

and simultaneously learn a policy network (as an actor) to select data and a value network (as a critic) to estimate future returns using the diversity measure as an evaluation reward. Our actor-critic data selection method has the advantages of low variance updates as well as credit assignment and significantly outperforms domain adaptation (Liu et al., 2019) on various domains without requiring any target domain knowledge. The architecture is shown in Figure 2 and Section 2.

In summary, our work mainly contributes to the following: (1) Modeling sample dependency for effective data sampling using diversity; (2) Measure dataset diversity using max dispersion, convex hull volume, and graph entropy; (3) Introduction of the Advantage Actor-Critic reinforcement learning framework to select informative samples and enhance NLP accuracy and generalization.

The rest of the paper is organized as follows. In Section 2.1, we detail Diversity Advanced Actor-Critic reinforcement learning (DAAC). Then in Section 2.2, we introduce different diversity measures. Afterward, in Section 3, we empirically verify the generalization and accuracy improvement using DAAC. We discuss related work in Section 4 and conclude the paper in the last section.

2 Diversity Advanced Actor-Critic

We now present the details of our Diversity Advanced Actor-Critic (DAAC) algorithms. The details of data selection and fine-tuning process are depicted in Figure 2. Our task model $\mathcal{F}$ (any Deep Learning-based NLP model, such as Transformer, etc.,) is pre-trained 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, as in Liu et al. (2019), we shuffle and randomly partition $\mathcal{X}$ into T disjoint data batches so that $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$, with $\mathcal{B}_t = \{x_{(t-1)n|T+1}, x_{(t-1)n|T+2}, ..., x_{tn|T}\}$. $n|T$ is the integer division of n by T, and $T \leq t$. If $\bmod\ (n,T) \neq 0$, then the last batch has a variable size of $\bmod\ (n,T)$ and collects the remaining sentences. For each batch, we select and denote $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | x_i \in \mathcal{B}_t\}$ as the selected data with size o. After obtaining $\hat{\mathcal{B}}_t$, we use $\hat{\mathcal{B}}_t$ to fine-tune $\mathcal{F}$. $\mathcal{F}$ and its encoder g is updated on $\hat{\mathcal{B}}_t$ for T times in an epoch, and each update is based on the previous checkpoint.

Our goal is to improve our task model $\mathcal{F}$ on any test domain whose distribution is different from the source training set $\mathcal{X}$, by learning an effective subset of $\mathcal{X}$ to fine-tune $\mathcal{F}$. Figure 2-(1) shows our new model.

Our framework is based on Actor-Critic reinforcement learning. Our Actor-Critic method has the following properties, which are very desirable to achieve our goals: variance deduction, efficient sampling, and credit assignment (Konda and Tsitsiklis, 2000; Grondman et al., 2012). 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. In the following context, we will first introduce our Actor-Critic algorithm (Section 2.1) and then our three different reward functions based on diversity (Section 2.2).

2.1 Advantage Actor Critic

2.1.1 Markov Decision Process

We cast the data selection as a reinforcement learning problem with a Markov Decision Process (MDP). Our data selection policy π, a mapping from states to actions, serves as an agent to interact with an environment that constitutes an NLP model, we call task model $\mathcal{F}$ over T time steps. At each time step t, a state s_t, an action a_t and a reward r_t are collected.

First, the encoder g inside the NLP model (e.g. an embedding layer in LSTM, or an encoder in transformer) transforms a batch of data $\mathcal{B}_t$ into its embedding s_t ($s_t = g(B_t)$ in Fig 2, e.g. vector representations of the encoder output of a sentence).

Secondly, the policy π outputs a probability distribution for the batch of state 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. For example, for the k-th sentence in the batch with probability p_k, we generate a random number between 0 and 1. If p_k is larger than the random number, then the k-th sentence is selected; otherwise, it is not selected. The result of Bernoulli sampling is represented as a 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, task model $\mathcal{F}$ as well as encoder g are finetuned by the selected subset $\hat{\mathcal{B}}_t$. In the meantime, a scalar reward $r_t = \mathcal{R}(g(\hat{\mathcal{B}}_t))$ is calculated by designed diversity reward functions $\mathcal{R}$ which we give definitions in Section 2.2.

Finally, the policy agent π updates its weights using the collected s_t, a_t and r_t, where state s_t is the encoded representation of a batch of data $\mathcal{B}_t$, action a_t is a vector with each value of either 0 or 1 representing each sentence in the batch not being or being selected, and the scalar reward r_t measures the diversity of selected batch $\hat{\mathcal{B}}_t$. This optimization process is expanded in next section 2.1.2.

2.1.2 Training algorithm

We employ the Advantage Actor Critic algorithm that 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 a_t in state s_t is defined in Mnih et al. (2016) as

$$A(s_t, a_t) = \mathcal{Q}(s_t, a_t) - \mathcal{V}(s_t) \quad (1)$$

$$\approx \sum_{j=0}^{T-t} \gamma^j r_{t+j} - \mathcal{V}(s_t) \quad (2)$$

where $\gamma \in (0, 1]$ is the discounting factor, and we set its value as 0.99. $\mathcal{V}$ is the value function (critic) implemented as a value network in Figure 2-(1).

The data selection policy π (actor) is implemented as a policy network, whose training objective is

$$\nabla_\theta \mathcal{J}(\theta) = E_{\pi_\theta} \nabla_\theta \log \pi_\theta(a_t|s_t;\theta) A(s_t, a_t; \theta, \theta_v).$$

The parameters of the policy network θ are updated by:

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \log \pi(a_t|s_t;\theta_t) A(s_t, a_t; \theta_t, \theta_{vt}) \quad (3)$$

where α is the learning rate, and Equation 2 estimates $A(s_t, a_t; \theta, \theta_v)$.

The objective of value network is:

$$\nabla_{\theta_v} \mathcal{V}(\theta_v) = E_{\pi_\theta} \nabla_{\theta_\mathcal{V}} (r_t - \mathcal{V}(s_t; \theta_v))^2$$

The parameters of value function θ_v is updated by:

$$\theta_{v(t+1)} = \theta_{vt} + \alpha \nabla_{\theta_{vt}} (r_t - \mathcal{V}(s_t; \theta_{vt}))^2 \quad (4)$$

The policy network $\pi(a_t|s_t;\theta)$ is a two-layer network that has two nodes in the first layer and one hidden output produced by the $\tanh$ activation function and one softmax output. The value network is a two-layer network that has eight nodes

Algorithm 1 DAAC Training Algorithm

```
Input: Epoch $L$, learning rate $\alpha$, discount factor $\gamma$, training set $\mathcal{X}$, pre-trained task model $\mathcal{F}$ (including encoder $g$), reward function $\mathcal{R}$ (discussed in section 2.2)
Output: selected data, fine-tuned $\mathcal{F}$, policy $\pi_\theta$, data value estimator $\mathcal{V}_{\theta_v}$
1:  Initialize data selection policy $\pi_\theta$ and value estimator $\mathcal{V}_{\theta_v}$
2:  for episode $l = 1$ to $L$ do
3:      Shuffle (uniformly at random) all training samples;
4:      Equipartition $\mathcal{X}$ into $T$ (disjoint) sets with same size $n|T$: $\mathcal{X} = \{\mathcal{B}_t\}_{t=1}^{T} = \{\mathcal{B}_1, \mathcal{B}_2, ..., \mathcal{B}_T\}$;
5:      Initialize an empty list: episode history $\Upsilon$
6:      for all $\mathcal{B}_t \in \mathcal{X}$ (uniform transition probability) do
7:          $s_t = g_t(\mathcal{B}_t)$;
8:          Obtain batch action $a_t$ by sampling based on $\pi_\theta(s_t)$;
9:          $\hat{\mathcal{B}}_t = \{(x_i)_{i=1}^{o} | a_i = 1\}$, where $o$ is selected sample size;
10:         Update task model $\mathcal{F}(g_t)$ by fine-tuning on $\hat{\mathcal{B}}_t$;
11:         $r_t = \mathcal{R}(\hat{\mathcal{B}}_t, g)$;
12:         Store $(s_t, a_t, r_t)$ to episode history $\Upsilon$;
13:     end for
14:     for all $(s_t, a_t, r_t) \in \Upsilon$ do
15:         Obtain $A(s_t, a_t)$ for each batch (Eq. 2);
16:         Update policy weights $\theta$ (Eq. 3);
17:         Update value estimator weights $\theta_v$ (Eq. 4);
18:     end for
19:     Clear episode history $\Upsilon$;
20: end for
21: return $\mathcal{F}$,$\pi_\theta$ and $\mathcal{V}_{\theta_v}$
```

in the first layer and one hidden output produced by the ReLU activation function, and one linear output for the value network $\mathcal{V}(s_t; \theta_v)$ following Mnih et al. (2016). The training algorithm is shown in Algorithm 1.

2.2 Diversity Measures

The reward function in Section 2.1 is the diversity of the selected batch subset. We do not need any target domain knowledge, and the diversity is measured with the following three methods.

2.2.1 Max Dispersion

The dispersion of a set is the sum of all pair-wise distances within the set (Cevallos et al., 2016). Intuitively, maximizing the dispersion (denote as **MD**) of a set can enlarge the semantic coverage of the set, and thus diversify the content of the set. Formally, x_i and x_j are any two training sentences in a training set G, and their sentence embeddings are $g(x_i)$ and $g(x_j)$, respectively, we define the $dispersion$ as

$$\mathcal{D}(G) = \sum_{(x_i, x_j) \in G} d(x_i, x_j), \tag{5}$$

where $d(x_i, x_j) = 1 - \frac{g(x_i) \cdot g(x_j)}{\|g(x_i)\| \|g(x_j)\|}$. The diversity of a set of training samples (sentences) is computed as the sum of each pairwise sample distance, which can be measured by any distance metrics, i.e., Cosine distance. We take the NLP task model encoder output as the sentence embedding to compute pairwise sample distance. The sentence embedding preserves pairwise sentence distance from the semantic space to the high-dimensional vector space. More details of the algorithm are shown in Algorithm 2, with a worst time complexity of $O(n^2)$, where n is the number of samples in the set.

2.2.2 Convex Hull Volume

Instead of looking at each pair of sentences, we can simplify the computing process by looking at the boundary of a set of training samples in the high dimensional space and compute the volume of such space using Convex Hull Volume (denote as **CV**). Specifically, we treat each sentence as a point in the embedding space and approximate the volume of a convex body. In a Euclidean plane, given a finite set of points Q, it is sometimes interesting to determine its convex hull, namely the minimum convex polygon so that any point of Q is either inside this polygon or at its border. There are a number of algorithms to compute the convex hull. Since our embedding is in high dimension, i.e., 256, we consider N-dimensional quickhull, which was introduced in Barber et al. (1996) . Just like the quicksort algorithm, it has the expected time complexity of $O(n \log n)$ by divide and conquer approach, but may degenerate to $O(n^2)$ in the worst case. Details shown in appendix Algorithm 3.

2.2.3 Graph Entropy

Applying entropy on the distances, we introduce Graph Entropy (**GE**), following the so-called local measure of distance-based substructure entropy, details in Dehmer and Mowshowitz (2011). As shown in Figure 2, graph entropy measures the uncertainty of the pairwise distance of sentence (or word) embeddings in the geometric space. To compute graph entropy of a set of training samples, we cast the sentence embeddings as vertices and distances between sentence embeddings as edges to form a clique. The distance entropy $\mathbf{I}$ of each sample x is defined as

$$\mathbf{I}(x) = -\sum_{i=1}^{M} \mathbf{f}(d(x, x_i)) \log(\mathbf{f}(d(x, x_i))).$$

M is the number of sentences in the set, $d(x, x')$ is the Cosine distance between the encoder output vector representation of x and x', $\mathbf{f}(d(x, x')) = \frac{d(x,x')}{\sum_{i=1}^{M} d(x,x_i)}$ is the relative distance of edge $d(x, x')$ (instead of words). The graph entropy of entire set is then defined as the sum of all distance entropy $\mathbf{I}$:

$$\mathcal{G}(G) = \sum_{i=1}^{M} \mathbf{I}(x_i) \tag{6}$$

For a set with n data samples, there are $\binom{n}{2}$ edges in the resulting graph. The time complexity of graph entropy is thus $\mathcal{O}(n^2)$ by looping through each edge in the graph. Details are shown in Algorithm 5.

3 Experiments

We describe our experimental details and demonstrate that our DAAC improves baselines in three NLP applications, including two recognition tasks: sentiment analysis, named entity recognition (NER), and one generation task of language modeling, without the knowledge of the target domain. We give hyperparameters of DAAC in appendix.

Finetuning vs. Data Selection We introduce our four models based on DAAC. The first three models finetune a pretrained NLP model using max dispersion **MD**, convex hull volume **CV** and graph entropy **GE** as reward function. As a comparison, we add the fourth model trained on selected data from scratch that does not rely on a pretrained model, denoted as data selection **DS**. Specifically, we train A2C using MD as reward function for 150 epochs first, and use the trained A2C to select data samples from all source data, then use only the selected data to train the NLP model from scratch. We select 3833 (63.8%) and 7630 (54.3%) samples from the source domain of sentiment analysis and NER respectively, and 21017 (49.9%) samples from Penn Treebank and 18043 (49.1%) samples from WikiText-2 in language modeling experiment.

	auto	beauty	food	instrs	office	comptr	tools
All	56.4	54.0	56.0	59.1	56.8	55.1	55.9
Rand	57.0	54.9	56.9	59.5	59.2	56.1	56.0
Mtl	59.4	55.8	60.7	61.0	61.8	56.2	62.5
Meta	56.3	52.2	58.4	56.4	58.1	52.9	48.8
Sim	73.4	62.9	77.3	79.4	78.4	63.0	81.2
MD	76.8	65.2	80.5	82.9	82.8	66.0	85.7
CV	76.4	65.4	80.4	82.8	82.2	66.0	85.7
GE	75.4	65.0	78.9	81.4	80.5	65.5	83.3
DS	**78.0**	**65.9**	**81.7**	**84.4**	**83.7**	**67.5**	**86.3**
+%	18.6	10.1	21.0	23.4	21.9	11.3	23.8

Table 1: Sentiment analysis accuracy [%] on amazon unprocessed domains. Full results of 21 domains in appendix. Baselines **All**, **Rand**, marginal transfer learning **Mtl** (Blanchard et al., 2021), Metareg **Meta** (Balaji et al., 2018) and our four measures are trained on the joint dataset of books, dvd & kitchen domain, and do not use any test/target domain data, while **Sim** (Liu et al., 2019) uses target domain electronics. Results in each domain are averaged over five random seeds. Last column: absolute improvement between **DS** and **Mtl**.

Baselines We compare our models with five baselines: 1) **All** The models are trained on all source data; 2) **Rand** The models are trained on randomly selected 50% source data; 3) Marginal transfer learning (**Mtl**) by Blanchard et al. (2021), a domain generalization framework using kernel methods to augment feature space. 4) MetaReg (**Meta**) by Balaji et al. (2018), a domain generalization method using meta-learning. 5) **Sim** by Liu et al. (2019) that uses similarity features between target test domain and source training domains. To be noted, **Sim** is a domain adaptation method, which requires the usage of target domain data, while other baselines and our four models do not use target domain data.

3.1 Sentiment Analysis

Settings We use the Amazon product review dataset (Blitzer et al., 2007) for the sentiment analysis task. The sentiment analysis baseline is a CNN classifier (Kim, 2014). We pretrain the CNN for two epochs as (Liu et al., 2019) for a fair comparison.

Results Table 1 shows the results on amazon domains. Max dispersion outperforms other diversity measures in 11 out of all 22 unseen domains, and the Data Selection achieves a most boost of 38.0% (relative) and 23.8% (absolute) percent in the tools domain. This observation aligns with previous findings that 60%-70% important samples can perform similarly or better than training on the entire dataset

	politics	science	music	literature	ai
All	26.49	19.84	12.26	16.38	13.92
Rand	26.07	19.68	12.81	17.47	13.68
Mtl	28.47	22.47	13.49	18.97	15.68
Meta	29.16	22.57	13.75	20.92	15.74
Sim	29.37	22.51	15.31	20.63	16.00
MD	29.64	23.09	15.69	21.21	16.72
CV	29.42	22.85	15.71	21.30	**17.13**
GE	**29.75**	**23.75**	**16.03**	**21.58**	16.91
DS	29.74	23.18	15.22	20.56	15.97

Table 2: NER F1-scores. Results are averaged over three runs.

	WikiText-2		Penn Treebank	
	IWSLT'17	Bio'21	IWSLT'17	Bio'21
All	328.23	259.47	147.03	117.17
Rand	515.22	456.78	234.14	157.39
MD	323.89	251.45	144.78	114.68
CV	325.22	252.17	145.43	115.31
GE	324.57	251.91	144.95	114.89
DS	**321.05**	**222.78**	**141.02**	**72.52**

Table 3: Language modeling: Perplexity scores on two test domains. First row: source training domain; Second row: test domains.

(Yoon et al., 2019; Fan et al., 2017). We also conduct a domain adaptation experiment which aims to maximize the test accuracy on one specific target domain and outperform all baselines with a significant test $p = 0.00038$, as shown in table 8 in appendix.

3.2 Named Entity Recognition

Settings We use the CoNLL2003 English NER dataset (Sang and Meulder, 2003) as source training set and the five domains from CrossNER dataset (Liu et al., 2020) as test sets, which has specialized entity categories for each domain. We finetune the pretrained BERT model (Devlin et al., 2018) on source training set by adding a linear layer on top of the hidden-states output and then evaluate the F1-scores on five test domains.

Results The result is reported in Table 2. Graph Entropy achieves best performance among all the three diversity measures. In particular, Data Selection performs better than training with all data directly (All) and random selection (Rand), which means important samples are selected and noisy samples are filtered out.

3.3 Language Modeling

Settings Our baseline is a Transformer language model (Vaswani et al., 2017) with default hyperparameters. We experiment with two moderate size datasets WikiText-2 (Merity et al., 2016) and Penn Treebank. As for evaluation, we report perplexity scores on two translation datasets from different domains, IWSLT' 17 (TED talk english) and WMT Biomedical' 21 (english). 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.

Results The evaluation results are shown in Table 3. The perplexity on two test domains have been improved. Specifically, test perplexity of Bio' 21 has an improvement of 44.65 (37.9% relative improvement) while trained by our selected data. The comparison with randomly selected result (Rand) further proves the effectiveness of data selected by DAAC.

3.4 Ablation Analysis

MD vs. GE In our three experiments we observe Max Dispersion (MD) performs best in sentiment analysis and language modeling, and Graph Entropy measure (GE) achieves the best performance in named entity recognition. We infer the smaller improvement of convex hull volume might be due to error propagation of the quickhull algorithm.

Regarding the reason why GE performs best in named entity recognition, we first analyze the difference of operations on sentence embeddings for GE and MD. Since GE is calculated by the sum of distance entropy of each sentence (sample), there are more operations (eg. $\log$ and normalization) depending on the sentence embedding input than MD. Thus, performance of GE is more relying on the precision of sentence embeddings compared to that of MD. Then looking into the key difference between the task of NER and other tasks, the source of sentence embeddings are apparently different. In NER, sentence embeddings are trained by BERT, while in sentiment analysis they are trained by CNN. As a result, we conjecture the more precise sentence embeddings trained by BERT lead to the better performance of GE on NER task. To verify this assumption, we quantitatively evaluate the quality of sentence embeddings generated by finetuned CNN and finetuned BERT on semantic textual similarity (STS) benchmarks STS-B-Dev (Cer et al., 2017), STS14 and STS15 (Agirre et al., 2014, 2015). We report the Spearman's rank correlation between the cosine similarity of sentence embeddings using the SentEval toolkit (Conneau and Kiela, 2018). The results in table 4 show preliminary evidence that our conjecture holds. In short, MD can be consid-

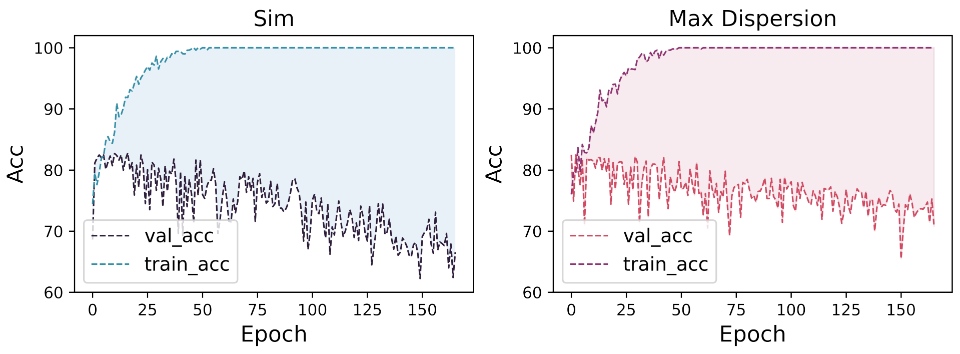

Figure 3: Training/Validation accuracy change over epochs. Validate set is gourmet food domain of unprocessed Amazon product reviews. Training set is a joint set of dvd, kitchen and books, and target domain in **Sim** is electronics. **MD** can help alleviate overfitting on test domains compared to the domain adaptation method **Sim** (Liu et al., 2019)

	STS-B-Dev	STS14	STS15
Finetuned-CNN	29.92	23.26	19.77
Finetuned-BERT	32.02	24.54	23.06

Table 4: Spearman's rank correlation between the cosine similarity of sentence embeddings generated by finetuned-CNN and finetuned-BERT. Numbers are reported as $\rho\times 100$.

ered for general tasks, however, when there is a reliable sentence embedding source, GE can be a better choice than MD.

Overfitting We plot the validation accuracy curve of the sentiment classifier on one test domain, as shown in Fig 3. The validation accuracy of Sim (Liu et al., 2019) model degrades faster and more significant than DAAC. The gap of blue and red area, which is the gap between training accuracy and validation accuracy of models finetuned by Sim and MD, suggests DAAC can help alleviate overfitting on test domains compared to the domain adaptation method Sim.

Time As discussed in 2.2, MD and GE both take account for the internal structure of a set of training samples and take time $O(n^2)$, where n is size of the set. CV can reduce time to $O(n \log n)$ by utilizing divide and conquer, while accompanying with the degradation of performance. In theory, the entire training time of DAAC-finetuning is approximately T times compared to training with all data, where T is the time steps in Markov Decision Process, or the number of batches. In the finetuning setting, we follow Liu et al. (2019) and Yoon et al. (2019), using $T \in [2, 4]$ and finetune NLP model for 200 time steps. However, in the data selection (DS) setting, the training time is roughly 20 times more compared to the finetuning setting, since we train A2C for extra epochs (10000 time steps with $T = 60$, batch size 100) to ensure both the value and policy network converge. In practice, training with all source data directly (All) in sentiment analysis takes 131 seconds while finetuning with MD takes 217 seconds, and selecting data out then training (DS) takes 4774 seconds on one Tesla V100 GPU.

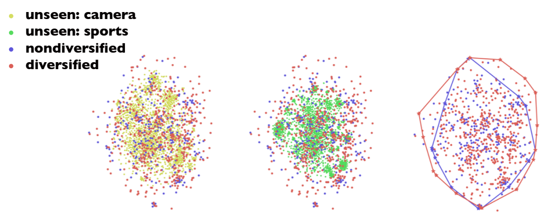

Figure 4: t-SNE plot of diversified/nondiversified training data and their coverage with unseen domains.

	Camera	Sports	Food
Diversified	7956	8443	5158
Nondiversified	8348	8886	5469

Table 5: Out-of-vocabulary of diversified and nondiversified set

Visualization Figure 4 visualizes data samples selected by our DAAC (diversified, red) and those randomly selected (nondiversified, blue), as well as the test data samples (unseen domains) in the embedding space. We observe that our DAAC selected samples are more widely spread, thus covering more semantic meaning for most test sets. Figure 4 (right) shows that DAAC selected samples has a larger convex hull volume than random selected samples after removing outliers. Furthermore, table 5 shows that our DAAC has a smaller size of out-of-vocabulary than the baseline on the test domains. Data samples are selected from the magazines domain of Amazon product reviews (Blitzer et al., 2007). We use max dispersion (MD) diversity measure, sentence-transformer toolkit (Reimers and Gurevych, 2019), and plot the embedding into two-dimension t-SNE.

4 Related work

Existing domain generalization methods related to data focus on data augmentation (Zhou et al., 2020; Qiao et al., 2020) and data generation (Wang et al., 2021). These methods increase the quantity of training set while our method utilizes existing data maximally by modeling the intrinsic sample dependency.

There have been several influential works (Moore and Lewis, 2010; Axelrod et al., 2011; Ruder and Plank, 2017a) on data selection that significantly contributed to today's NLP state-of-the-arts. Fan et al. (2017) proposes a data filter based on deep reinforcement learning on image and sentiment classification task. Feng et al. (2018) implemented an instance selector using reinforcement learning to filter noisy data to improve the accuracy on natural language inference (Qu et al., 2019), sentiment analysis, part-of-speech tagging and dependency parsing (Liu et al., 2019). These work select the training data close to a given target domain for domain adaptation. In contrast, we aim to enhance the model generalization and increase the accuracy on any arbitrary domain.

As for diversity measures, Ruder and Plank (2017b) examines the effects of Shannon entropy in the transfer learning setting, but they do not consider content semantic meaning. Shi et al. (2021) uses determinantal point processes to select diverse data to reduce the labor of annotating training examples for dependency parsing, but the proposed diversity measure cannot be generalizaed to other NLP tasks. Other works incorporate the notion of diversity into topics like language representation (Chubarian et al., 2021), and ensemble language modeling (Duan et al., 2021).

5 Conclusion

We introduce Actor-Critic reinforcement learning rewarded with diversity measures to select effective training data that significantly enhances the accuracy of sentiment analysis, named entity recognition, and language modeling tasks across various domains. Without any target domain knowledge, our method outperforms the CNN, Transformer and BERT baselines. Our experiments show that modeling sample dependency by increasing training set diversity enhances the learning generalization and prediction accuracy.

Acknowledgments

We appreciate the National Science Foundation (NSF) Award No. 1747728 and NSF CRAFT Award, Grant No. 22001 to fund this research. We are also thankful for the support of the Google Cloud Research Program.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 252–263.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M Cer, Mona T Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *SemEval@ COLING*, pages 81–91.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. 2018. Metareg: Towards domain generalization using meta-regularization. *Advances in Neural Information Processing Systems*, 31:998–1008.

C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4):469–483.

Gilles Blanchard, Aniket Anand Deshmukh, Ürün Dogan, Gyemin Lee, and Clayton D. Scott. 2021. Domain generalization by marginal transfer learning. *J. Mach. Learn. Res.*, 22:2:1–2:55.

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.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Alfonso Cevallos, Friedrich Eisenbrand, and Rico Zenklusen. 2016. Local search for max-sum diversification. *CoRR*, abs/1607.04557.

Karine Chubarian, Abdul Rafae Khan, Anastasios Sidiropoulos, and Jia Xu. 2021. Grouping words with semantic diversity. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3217–3228, Online. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*.

Geoff Davis, Stephane Mallat, and Marco Avellaneda. 1997. Adaptive greedy approximations. *Constructive approximation*, 13(1):57–98.

Matthias Dehmer and Abbe Mowshowitz. 2011. A history of graph entropy measures. *Information Sciences*, 181(1):57–78.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping.

Zhibin Duan, Hao Zhang, Chaojie Wang, Zhengjue Wang, Bo Chen, and Mingyuan Zhou. 2021. EnsLM: Ensemble language model for data diversity by semantic clustering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2954–2967, Online. Association for Computational Linguistics.

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

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *Proceedings of the aaai conference on artificial intelligence*, volume 32.

Zhiqiang Gong, Ping Zhong, and Weidong Hu. 2019. Diversity in machine learning. *IEEE Access*, 7:64323–64350.

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.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2020. Multi-source domain adaptation for text classification via distancenet-bandits.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

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

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

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.

Zihan Liu, Yan Xu, Tiezheng Yu, Wenliang Dai, Ziwei Ji, Samuel Cahyawijaya, Andrea Madotto, and Pascale Fung. 2020. Crossner: Evaluating cross-domain named entity recognition. *CoRR*, abs/2012.04373.

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

Michele Merler, Nalini Ratha, Rogerio S Feris, and John R Smith. 2019. Diversity in faces. *arXiv preprint arXiv:1901.10436*.

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.

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

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*.

Chao Qian, Jing-Cheng Shi, Yang Yu, Ke Tang, and Zhi-Hua Zhou. 2016. Parallel pareto optimization for subset selection. In *IJCAI*, pages 1939–1945.

Fengchun Qiao, Long Zhao, and Xi Peng. 2020. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12556–12565.

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.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in nlp—a survey.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Seyed Saeed Changiz Rezaei. 2013. Entropy and graphs. *arXiv preprint arXiv:1311.5632*.

Sebastian Ruder and Barbara Plank. 2017a. Learning to select data for transfer learning with bayesian optimization. *CoRR*, abs/1707.05246.

Sebastian Ruder and Barbara Plank. 2017b. Learning to select data for transfer learning with Bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382, Copenhagen, Denmark. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, cs.CL/0306050.

Candice Schumann, Susanna Ricco, Utsav Prabhu, Vittorio Ferrari, and Caroline Pantofaru. 2021. A step toward more inclusive people annotations for fairness. *arXiv preprint arXiv:2105.02317*.

Shreya Shankar, Yoni Halpern, Eric Breck, James Atwood, Jimbo Wilson, and D Sculley. 2017. No classification without representation: Assessing geodiversity issues in open data sets for the developing world. *arXiv preprint arXiv:1711.08536*.

Tianze Shi, Adrian Benton, Igor Malioutov, and Ozan Irsoy. 2021. Diversity-aware batch active learning for dependency parsing. *CoRR*, abs/2104.13936.

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.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin. 2021. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*.

Baoyuan Wu, Weidong Chen, Peng Sun, Wei Liu, Bernard Ghanem, and Siwei Lyu. 2018. Tagging like humans: Diverse and distinct image annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7967–7975.

Jinsung Yoon, Sercan Ömer Arik, and Tomas Pfister. 2019. Data valuation using reinforcement learning. *CoRR*, abs/1909.11671.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. 2020. Learning to generate novel domains for domain generalization. In *European conference on computer vision*, pages 561–578. Springer.

A Appendix

A.1 Notations

Notation	Meaning
$\mathcal{F}$	NLP model
g	encoder
$\mathcal{X}$	Training data set
x_i	sentence
$\|\mathcal{X}\|$	Training set size
T	number of batches; maximum training steps in an epoch
B_t	batch
$\hat{B}_t$	selected batch
s_t	batch state
s_k	single sentence state
a_t	action on batch B_t at time step t
a_k	action on single sample x_k at time step t
$\|B_t\|$	batch size
π	policy
r_t	reward at time step t
R_t	total future reward from t to T
$Q^\pi(s_t, a_t)$	action value
$V^\pi(s_t)$	expected future return following π since time step t
$b(s_t)$	baseline function
θ	parameters of policy network
∇_θ	$\frac{\partial X}{\partial \theta}$
$\nabla_\theta \mathcal{J}(\theta)$	objective function of policy
L	epoch number
α	learning rate
γ	discount factor
$\mathcal{F}'$	pretrained task model (including encoder g)
$\mathcal{E}$	episode record, including s_t, a_t, r_t
G	a set of samples
d	cosine distance of embeddings
M	number of sentences in a set G
J	sentence length
$h(\cdot; n)$	n-gram entropy

Table 6: Notation table

A.2 Hyperparameters of A2C

Hyperparameter	Value
learning rate	$7e-4$
discount factor	0.99
entropy coefficient	0
value function coefficient	0.5
RMSProp epsilon	$1e-5$
number of steps (finetuning)	200
number of steps (data selection)	15000

Table 7: Hyperparameters of A2C

A.3 Diversity algorithms

See 2, 3 and 5.

A.4 Sentiment analysis full results

See table 9 and 8.

Algorithm 2 Max Dispersion set diversity [**MD**]

```
Input: A batch of training samples $G = \{(s_i)_{i=1}^M\}$ of size $M$.
Output: Dispersion of the batch $D(G)$
1: Initialize $D(G) = 0$;
2: Initialize an empty set $S$;
3: for all $s_i \in G$ do
4:     Obtain sentence embedding $v(s_i)$ by averaging all word embedding in $s_i$;
5:     Normalize $v(s_i)$ by softmax function;
6:     for all $s_j \in G$ if $s_j \neq s_i$ and tuple $(i, j)$ not in $S$ and tuple $(j, i)$ not in $S$ do
7:         Obtain sentence embedding $v(s_j)$ by averaging all word embedding in $s_j$;
8:         Normalize $v(s_j)$ by softmax function;
9:         Compute Cosine distance $d(v(s_i), v(s_j)$;
10:        $D(G) = D(G) + d(v(s_i), v(s_j))$;
11:        Add tuple $(i, j)$ to $S$;
12:    end for
13: end for
14: return $D(G)$
```

Algorithm 3 Quickhull algorithm

```
Input: A batch of training samples $G = \{(s_i)_{i=1}^M\}$ with size $M$; $M \geq 2$
Output: Convex hull set $h$
1: Initialize empty set convex hull $h$;
2: Initialize empty dictionary $d$;
3: for all $s \in G$ do
4:     for all $s' \in G$ that $s' \neq s$ do
5:         Compute Cosine similarity $d(v(s), v(s'))$ between sentence embedding of $s$ and $s$;
6:         $d[(s, s')] = d(v(s), v(s'))$
7:     end for
8: end for
9: Sort $d$ in descending order
10: Add the two sentence $s_p$ and $s_q$ that has the max distance to convex hull set $h$. Line formed by $s_p$ and $s_q$ segment the space into left half $S_1$ and right half $S_2$.
11: Call subroutine FindHull($S_1$, $s_p$, $s_q$ )
12: Call subroutine FindHull($S_2$, $s_q$, $s_p$ )
13: return Convex hull set $h$
```

		books	dvd	electronics	kitchen
w/o tgt	All	77.58	79.88	83.50	84.50
	Rand	76.39	79.30	82.99	84.01
	Mtl	77.20	79.90	84.25	85.50
	Meta	76.10	79.60	83.90	85.10
w/o tgt	MD	79.67	**84.08**	85.29	87.38
	CV	79.47	83.08	84.58	87.48
	GE	**80.08**	83.58	**85.68**	**87.88**
w/ tgt	Sim	78.97	82.07	82.28	86.18

Table 8: Sentiment Analysis accuracy on one test domain[%]. The "book" column is tested on the book domain, while using other three domains for training.

Algorithm 4 Subroutine: FindHull

```
Input: A batch of training samples $S$; point $p$ and $q$.

1: From the given set of points in $S$, find farthest point $f$
2: Add point $f$ to convex hull set $h$. Three points $p$, $q$, and $f$ partition the remaining points of $S$ into 3 subsets: $S_0$, $S_1$, and $S_2$, where $S_0$ are points inside triangle $pqf$, $S_1$ are points on the right side of the line from $p$ to $f$, and $S_2$ are points on the right side of the line from $f$ to $q$.
3: Call FindHull($S_1$, $p$, $f$)
4: Call FindHull($S_2$, $f$, $q$)
```

Algorithm 5 Graph entropy based set diversity [GE]

```
Input: A batch of training samples $G = \{(s_i)_{i=1}^{M}\}$ with size $M$
Output: Graph entropy $\mathcal{G}(G)$
1: Initialize $\mathcal{G}(G) = 0$;
2: Initialize empty dictionary $d$;
3: for all $s \in G$ do
4:     for all $s' \in G$ that $s' \neq s$ do
5:         Compute Cosine similarity $d(v(s), v(s'))$ between sentence embedding of $s$ and $s$;
6:         $d[s] = d[s] + d(v(s), v(s'))$;
7:     end for
8: end for
9: for all $s \in G$ do
10:    Initialize distance entropy of $s$: $\mathbf{I}(s) = 0$;
11:    for all $s' \in G$ that $s' \neq s$ do
12:        Compute Cosine similarity $d(v(s), v(s'))$ between sentence embedding of $s$ and $s'$;
13:        Compute relative frequency of distance $\mathbf{f}(d(v(s), v(s')))$
14:        Update $\mathbf{I}(s)$ according to equations in section 2.3.3
15:    end for
16:    $\mathcal{G}(G) = \mathcal{G}(G) + \mathbf{I}(s)$;
17: end for
18: return $\mathcal{G}(G)$
```

Domain	All	Rand	Mtl	Meta	MD	CV	GE	DS
apparel	49.43	50.47	47.65	49.92	50.53	50.48	**50.78**	49.95
auto	56.48	57.07	59.46	56.32	76.81	**76.94**	75.49	78.03
baby	50.42	51.09	50.80	50.06	50.86	52.95	**53.12**	52.81
beauty	54.03	54.93	55.89	52.26	65.43	65.43	65.03	**65.97**
camera	49.82	50.30	50.05	49.78	49.91	49.98	49.66	**50.32**
phones	52.79	53.77	53.93	52.51	62.13	61.81	61.18	**61.98**
computer	55.17	56.16	56.20	52.99	66.09	66.00	65.54	**67.54**
food	56.02	56.93	60.73	58.41	80.57	80.40	78.94	**81.76**
grocery	53.06	54.81	57.86	57.73	72.23	72.04	71.59	**73.10**
health	50.92	51.10	49.95	51.03	49.94	49.96	**51.11**	49.68
jewelry	56.63	57.92	58.98	58.15	75.77	75.56	74.84	**76.81**
magazines	50.78	50.56	50.42	50.03	51.15	**51.23**	50.94	50.97
music	50.05	50.27	50.08	49.60	50.08	**50.20**	50.20	50.03
instrs	59.15	59.55	61.88	56.41	82.92	82.82	81.40	**84.45**
office	56.84	59.28	61.87	58.14	82.92	82.23	80.58	**83.73**
outdoor	55.81	57.39	58.23	54.25	72.92	72.60	72.44	74.37
software	49.70	50.45	50.73	49.92	52.30	**52.37**	52.12	51.90
sports	51.08	51.13	49.97	49.98	50.66	50.68	50.70	50.25
tools	55.95	56.04	62.50	48.81	85.71	85.71	83.33	**86.31**
toys	50.07	50.82	50.20	50.35	51.03	51.02	51.38	50.17
video	51.65	50.72	50.08	50.68	50.50	50.70	50.85	**50.20**

Table 9: Sentiment analysis accuracy [%] on unknown domains.