PREDICTING OUT-OF-DOMAIN GENERALIZATION WITH LOCAL MANIFOLD SMOOTHNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding how machine learning models generalize to new environments is critical to their safe deployment. Recent work has proposed a variety of complexity measures that directly predict or theoretically bound the generalization capacity of a model. However, these methods rely on a strong set of assumptions that in practice are not always satisfied. Motivated by the limited settings in which existing measures can be applied, we propose a novel complexity measure based on the local manifold smoothness of a classifier. We define local manifold smoothness as a classifier's output sensitivity to perturbations in the manifold neighborhood around a given test point. Intuitively, a classifier that is less sensitive to these perturbations should generalize better. To estimate smoothness we sample points using data augmentation and measure the fraction of these points classified into the majority class. Our method only requires selecting a data augmentation method and makes no other assumptions about the model or data distributions, meaning it can be applied even in out-of-domain (OOD) settings where existing methods cannot. In experiments on robustness benchmarks in image classification, sentiment analysis, and natural language inference, we demonstrate a strong and robust correlation between our manifold smoothness measure and actual OOD generalization on over 4,000 models evaluated on over 100 train/test domain pairs.

1 INTRODUCTION

As deep neural networks find increasing use in safety-critical domains such as autonomous driving (Gupta et al., 2021) and healthcare (Wiens et al., 2019), it is important to develop methods to understand how these models generalize to new environments. Although models have been shown empirically to generalize in many settings (Hendrycks et al., 2020a; Allen-Zhu et al., 2018; Neyshabur et al., 2017a) , they also exhibit numerous failure cases. For example, models have been shown to overfit to a dataset's meta characteristics (Recht et al., 2019) or arbitrarily corrupted labels (Zhang et al., 2016), learn spurious correlations (Liang & Zou, 2022), and change their predictions even with small adversarial perturbations (Goodfellow et al., 2014; Papernot et al., 2017). Many methods have been proposed to mitigate these issues, but precisely characterizing the generalization properties of a model in diverse settings remains an open problem.

Directly estimating the generalization of a trained model on test data is one approach to this problem (Deng & Zheng, 2021; Jiang et al., 2021; Deng et al., 2021; Garg et al., 2022). However, these methods are typically calculated based on the output predictive distribution of a model, which can become poorly calibrated in out-of-domain (OOD) settings (Morteza & Li, 2022). Complexity measures are one potential set of tools to better fundamentally understand model generalization. Typically, these measures aim to theoretically bound generalization capacity (Vapnik & Chervonenkis, 1971; Bartlett & Mendelson, 2003; McAllester, 1999; Neyshabur et al., 2017a; Dziugaite & Roy, 2017; Neyshabur et al., 2015b) or directly predict generalization (Keskar et al., 2016; Liang et al., 2019; Neyshabur et al., 2015a; Schiff et al., 2021; Jiang et al., 2019), and are useful in reasoning about a model beyond its performance on a specific known test set. However, these methods often rely on a strong set of assumptions such as matching train/test distributions, labelled test data (Schiff et al., 2021), access to model weights (Neyshabur et al., 2015b; Bartlett et al., 2017; Neyshabur et al., 2017b), model gradients (Jiang et al., 2019), and training data (Keskar et al., 2016). In real world settings we are often given access only to a black box model and unlabeled data sampled from an unknown distribution. In these cases where it is potentially *more* important to have robust measures of model complexity, existing complexity measures cannot be used.

In this paper we propose a *local manifold smoothness* complexity measure that provides a generalization metric when existing complexity measures cannot be calculated. Our smoothness measure is defined as the sensitivity of a classifier's output to local perturbations along the data manifold. We expect a smoother classifier that is less sensitive in this manifold neighborhood to generalize better than a less smooth classifier that is more sensitive in the same neighborhood. Empirically, we generate samples in the local manifold neighborhood using data augmentation, and define sensitivity as the fraction classified into the most commonly predicted class among these samples. Since our method depends only on the existence of a data augmentation method, it can be used in a wide range of experimental settings.

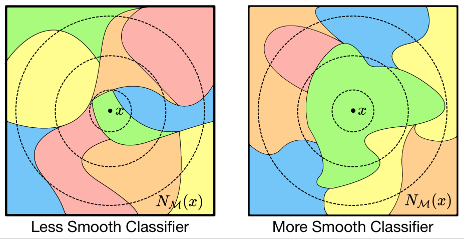


Figure 1: As the manifold neighborhood $N_{\mathcal{M}}(x)$ around a test point x grows, the decision space is partitioned into a distinct set of decision regions. Smoother classifiers (right) have decision spaces that are less sensitive in this neighborhood and should generalize better compared to less smooth classifiers (left).

We investigate the correlation of a model's local manifold smoothness with its capacity to generalize, defined as the model's performance on unseen data. We focus on experimental settings with OOD dataset shifts (Taori et al., 2020), where test data is sampled from a distribution different from the training distribution. We select common OOD benchmarks in image classification (Krizhevsky, 2009; Lu et al., 2020; Recht et al., 2018; Deng, 2012; Darlow et al., 2018; Netzer et al., 2011; Arjovsky et al., 2019), sentiment analysis (Ni et al., 2019), and natural language inference (Williams et al., 2018) and train a large pool of over 4,000 models with varying architectures and generalization properties on these datasets. Generating samples using well-established data augmentation methods (Ng et al., 2020; Cubuk et al., 2020; Wei & Zou, 2019; Xie et al., 2019) for each task and data domain, we evaluate the quality of our measure on over 100 pairs of training/test domains.

Specifically, given a pool of models trained with data from a single training domain, we evaluate the correlation of our measure with generalization on in-domain data (ID τ) and out-of-domain data (Micro/Macro τ). In settings where we have access to additional labeled test data from other domains and can make direct predictions of OOD generalization, we evaluate the Pearson R^2 and mean absolute error of these predictions. Finally, we evaluate the correlation of models trained with different architectures (Arch τ) as well as on extreme OOD data from specialized domains. We find that our smoothness measure outperforms baselines in all combinations of experimental setting and evaluation metrics, save one. To our knowledge, our measure is the only complexity measure that is applicable in common machine learning robustness evaluation settings, and provides the first analysis of a complexity measure in natural language tasks as well as OOD settings.

2 Related Work

Measures of Complexity Traditional methods of analyzing the generalization bounds of neural networks use theoretical measures of complexity. VC dimension (Vapnik & Chervonenkis, 1971) and Rademacher complexity (Bartlett & Mendelson, 2003) can be used to bound the generalization of particular function classes, although they are often vacuous at the scale of deep neural networks (Dziugaite & Roy, 2017). The PAC-Bayes framework (McAllester, 1999; Neyshabur et al., 2017a; Dziugaite & Roy, 2017; Garg et al., 2021) can be used to build tighter generalization bounds by considering the "sharpness" of the local minima. Norm-based measures (Neyshabur et al., 2015b; Bartlett et al., 2017; Neyshabur et al., 2017b) bound generalization by considering different norms of the weights of learned networks. More recent analyses have focused on empirically motivated measures such as the sharpness of minima in parameter space Keskar et al. (2016), Fisher-Rao norm Liang et al. (2019), distance from initialization (Nagarajan & Kolter, 2019), path norm (Neyshabur et al., 2015a), layer margin distributions (Jiang et al., 2019), and perturbation response curves Schiff et al. (2021). Most similar to our method is Aithal K et al. (2021), who use robustness to augmentations as a complexity measure. However, our method considers all neighborhood examples equally and does not require manual weight tuning, while their method utilizes the unaugmented example as a ground truth and requires tuning of augmentation penalties.

Predicting Generalization Recent work has focused on correlating complexity measures with and directly predicting in-domain generalization (Jiang et al., 2020). However, in real world settings, test and training distributions often differ (Koh et al., 2021). Although some generalization bounds have been derived for these OOD settings (Garg et al., 2021; Ben-David et al., 2007; Zhang et al., 2019), they rely on access to the test data distribution. A separate line of work aims to directly predict OOD generalization from unlabelled test data. These methods either predict the correctness on individual examples (Deng & Zheng, 2021; Jiang et al., 2021; Deng et al., 2021) or directly estimate the total error (Garg et al., 2022; Guillory et al., 2021; Chen* et al., 2021; Chuang et al., 2020; Vedantam et al., 2021). Although these methods work well in practice, they do not provide any insight into the underlying mechanism of generalization since they act only on the output layer of the network.

Measuring Function Smoothness The simplest way to measure the smoothness of a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ is through its differentiability class. However, this is not a very useful inductive bias for a model since differentiability is local and metric dependent. One alternative measure is Lipschitz continuity. Enforcing this property on neural networks is commonly done by restricting the learned function class with spectral regularization (Yoshida & Miyato, 2017), spectral normalization (Miyato et al., 2018), and other norm based regularization methods on layer weights (Bartlett et al., 2017; Neyshabur et al., 2017a; Foret et al., 2020). Other methods regularize the the norm of gradients with respect to the the inputs of the network (Elsayed et al., 2018; Arbel et al., 2018; Gulrajani et al., 2017). A key component of smoothness regularization is *where* and *how* it is enforced. Enforcing global properties such as Lipschitz continuity in regions where there is no data can restrict the learned function unnecessarily (Rosca et al., 2020). In addition, if the data lies on a low-dimensional manifold, then smoothness should only be regularized with respect to this manifold, rather than the entire space.

Intrinsic Dimensionality and Manifold Regularization Recent work has proven estimation error bounds for deep ReLU networks in Hölder(Schmidt-Hieber, 2019; Nakada & Imaizumi, 2020; Chen et al., 2019), Besov, mixed smooth Besov(Suzuki, 2018), and anisotropic Besov (Suzuki & Nitanda, 2021) function spaces based on their intrinsic dimensionality and smoothness. However, it is still unclear how to measure the intrinsic dimensionality or smoothness of functions on complex manifold spaces. Separate work aims to implicitly learn the manifold to improve models by encouraging invariance to local changes (Rifai et al., 2011), augmentation graphs (HaoChen et al., 2021), similar neighbors (Luo et al., 2018), or interpolation between points (Verma et al., 2019) on the manifold.

3 Manifold Smoothness Measure

In this section we introduce our manifold smoothness measure. We start by motivating our local smoothness definition, then describe how our measure is estimated in practice using samples from a neighborhood distribution. Finally we describe how we evaluate the quality of our measure.

3.1 Motivation

Consider a classification task from an input space $\mathcal{M} \in \mathcal{X}$ where $\mathcal{M}$ is a manifold of lower dimensionality that lies in an input space $\mathcal{X}$ with higher dimensionality (Chapelle et al., 2006), to an output space $\mathcal{Y}$ with k classes. We are given a model $f : \mathcal{X} \rightarrow \mathcal{Y}$ trained on an in-domain training dataset $\mathcal{D}_i = \{(x_i^1, y_i^1), \ldots, (x_i^n, y_i^n)\}$ sampled from a distribution $P_i(\mathcal{X}, \mathcal{Y})$, and an out-of-domain test dataset $\mathcal{D}_o = \{(x_o^1, y_o^1), \ldots, (x_o^m, y_o^m)\}$ sampled from a distribution $P_o(\mathcal{X}, \mathcal{Y})$. We assume further that domains are covariate shifted such that $P(\mathcal{Y}|\mathcal{X})$ does not change between domains and that $P_i(\mathcal{X})$ and $P_o(\mathcal{X})$ are defined only on the manifold $\mathcal{M}$. Consider a point $x \in \mathcal{D}_o$ that belongs to class j. We define the manifold neighborhood $N_{\mathcal{M}}(x, r)$ as the set of points in $\mathcal{M}$ that are at most a distance r away from x as measured in the original space $\mathcal{X}$. As r increases, the decision space within $N_{\mathcal{M}}$ becomes partitioned into a set of distinct decision regions for each class. For a given r, we define the neighborhood decision distribution as

$$p_j(x) = \frac{|\{x' \in N_{\mathcal{M}} : f(x') = j\}|}{|N_{\mathcal{M}}|}, \quad (1)$$

and define our smoothness measure as

$$\mu(f, x) = \max_{j \in \mathcal{Y}} p_j(x). \quad (2)$$

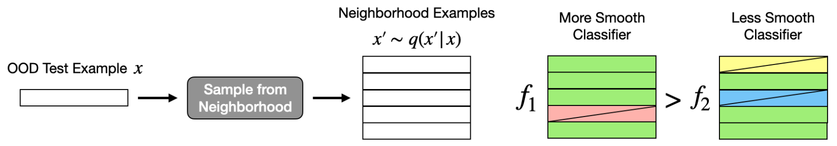


Figure 2: To estimate our smoothness measure we generate a set of nearby examples for every test example using a neighborhood distribution q. This set of examples is then evaluated using each classifier. We expect classifiers with more sampled points classified in the most common class to generalize better to the given test set.

Intuitively, this measure captures the sensitivity of a classifier's output within the manifold neighborhood of x. As shown in Figure 1, a smoother classifier will be less sensitive and will predict the same class for most of the points in the neighborhood. In contrast, a less smooth clasifier will be more sensitive and will predict different classes for different points in the neighborhood. We assume that test domains are sufficiently close to training domains so that our smoothness can be calibrated to make non-constant prediction. In practice, we will perform OOD evaluations to probe this limitation. Although dataset distance is difficult to quantify in high dimensional input spaces, in practice we find that even on extreme OOD settings our method still produces strong results.

3.2 Estimating Neighborhood Smoothness

Calculating our smoothness measure exactly is intractable since we do not have access to the true manifold $\mathcal{M}$. Instead, we assume access to a conditional distribution $q(x'|x)$ from which we can sample points from the manifold neighborhood $N_{\mathcal{M}}(x) = \{x' \sim q(x'|x)\}$. In this paper we consider distributions $q(x'|x)$ defined by data augmentation transformations. We modify the size of the neighborhood by making $q(x'|x)$ more or less noisy. We estimate our smoothness measure for a given point x as the fraction of points sampled from $q(x'|x)$ classified as the dominant label

$$\mu(f, x) = \mathbb{E}_{x' \sim q(x'|x)} \mathbb{1}\left(f(x') = \hat{y}(f, x)\right), \tag{3}$$

where

$$\hat{y}(f, x) = \arg\max_{j \in \mathcal{Y}} \mathbb{E}_{x'' \sim q(x''|x)} \mathbb{1}\left(f(x'') = j\right) \tag{4}$$

is the dominant label. The average smoothness across the entire dataset $\mathcal{D}_o$ is then $\frac{1}{m}\sum_{j=1}^{m} \mu(f, x_o^j)$. Since exactly calculating the expectation in Eqns. 3 and 4 is intractable, we estimate it as shown in figure 2. For a given test example x, we first generate n samples from the neighborhood distribution $q(x'|x)$. Next, we evaluate the classifier at each sampled point x' and calculate $\hat{y}$ as the most commonly predicted class. μ is then the fraction of sampled points that our classifier labels as $\hat{y}$.

Our smoothness measure is simple to calculate and makes no assumptions about the model or the distribution from which test data was sampled. In addition, since our measure relies only on the augmentation method $q(x'|x)$ chosen and does not use labels from the test data, it is applicable in many settings where existing complexity measures cannot be calculated, including common robustness evaluation settings.

3.3 Evaluation Metrics

Given a set of domains defined by distributions $\{P_1, P_2, \ldots P_n\}$ and a set of datasets $\{\mathcal{D}_i \sim P_i\}_{i=1}^{n}$ sampled from these domains, we train a set of models $F_i = \{f_i^1, f_i^2, \ldots, f_i^m\}$ on each dataset $\mathcal{D}_i$. We evaluate all models $f_i^k \in F_i$ on all OOD test datasets $\mathcal{D}_o : o \neq i$, generating a set of smoothness and generalization values (μ_{io}^k, g_{io}^k). We define generalization as the top-1 accuracy of f_i^k on $\mathcal{D}_o$.

We evaluate our measure first by predicting the generalization of a given model to an OOD test set. Specifically, we select an OOD test set $\mathcal{D}_o$ and an in-domain training set $\mathcal{D}_i : i \neq o$ and predict the OOD generalization g_{io}^k of a model f_i^k trained on $\mathcal{D}_i$ and evaluated on $\mathcal{D}_o$ from its smoothness value μ_{io}^k. To generate these predictions we use a linear model $\hat{g} = a\mu + b$ where $a, b \in \mathcal{R}$ are the parameters of our model. To estimate our parameters a and b, we select a pool of models $\{f_j^k \in F_j : j \neq i, o\}$ that are trained on all remaining datasets. Each model f_j^k in this pool is evaluated on the OOD dataset

$\mathcal{D}_o$ to give us a set of pairs $\{(\mu_{jo}^k, g_{jo}^k)\}$. We then find a, b by minimizing the mean squared error $(a^*, b^*) = \arg\min_{a,b} \sum_{j,k}(a\mu_{jo}^k + b - g_{jo}^k)^2$ on all models in the pool.

We use the learned parameters to make generalization predictions $\hat{g}_{io}^k = a\mu_{io}^k + b$ for every model $f_i^k \in F_i$ on $\mathcal{D}_o$ and measure the coefficient of determination $\mathbf{R^2}$ (Glantz et al., 1990). We also measure the residuals of our linear model by calculating the mean absolute error (**MAE**) between our predictions and the actual generalization. For every pair of training domain i and OOD test domain o, we evaluate R^2 and MAE then average each metric across all pairs.

We also consider the rank correlation between our smoothness measure and actual generalization. Specifically, for a pair of models f_i, f_j with measure and generalization pairs (μ_i, g_i) and (μ_j, g_j), we want $g_i > g_j$ if $\mu_i > \mu_j$. We use Kendall's rank τ coefficient (Kendall, 1938) to measure how consistent these sets of rankings are. We measure four different τ values:

ID τ This metric evaluates the correlation of our measure with in-domain generalization. We select a training dataset $\mathcal{D}_i$ and consider pairs $\{(\mu_{ii}^k, g_{ii}^k)\}$ generated from the set of models F_i trained on $\mathcal{D}_i$. τ values are averaged across all training domains.

Macro τ This metric evaluates the correlation of our measure individually on each training/OOD test domain pair. We select a training dataset $\mathcal{D}_i$ and a OOD test dataset $\mathcal{D}_o$ and consider pairs $\{(\mu_{io}^k, g_{io}^k)\}$ generated from the set of models F_i trained on $\mathcal{D}_i$. τ values are averaged across all pairs of training and OOD test domains.

Micro τ This metric evaluates the correlation of our measure on a given OOD test domain across models trained on all other domains. We select a single OOD test domain $\mathcal{D}_o$ and consider pairs $\{(\mu_{io}^k, g_{io}^k)\}$ generated from the set of models $\{f_i^k \in F_i : i \neq o\}$ trained on all other datasets $\{D_i : i \neq o\}$. τ values are averaged across all test domains.

Arch τ This metric evaluates the correlation of our measure on models trained with different architectures. Arch τ is calculated similar to Micro τ, except F_i now includes models from all architectures. τ values are averaged across all test domains.

4 Experimental Setup

Empirically evaluating the quality of a complexity measure is difficult and requires careful experimental design. Typically, evaluation is done by generating a large pool of models with sufficiently varied generalization properties, but if we generate these models by varying only a few hyperparameters, our observed correlation may be an artifact of these factors affecting both generalization and our measure. To this end, we follow a similar experimental setup to Jiang et al. (2019).

4.1 Data

For our experiments we focus on three tasks: image classification as well as sentiment analysis on single sentences and natural language inference on sentence pairs.

Image Classification For image classification we construct two sets of domains, one comprised of CIFAR10 (Krizhevsky, 2009), CINIC10 (Darlow et al., 2018), CIFAR10.1(Recht et al., 2018), and CIFAR10.2(Lu et al., 2020), as well as one comprised of SVHN (Netzer et al., 2011), MNIST (Deng, 2012), and Colored MNIST (Arjovsky et al., 2019). Domains in each set share the same set of output classes.

Sentiment Analysis (SA) We use the Amazon reviews dataset (Ni et al., 2019) which contains product reviews from Amazon. Following Hendrycks et al. (2020b) and Ng et al. (2020), we split the dataset into 10 different domains based on review category. For all domains and datasets, models are trained to predict a review's star rating from 1 to 5.

Natural Language Inference (NLI) We use the MNLI (Williams et al., 2018) dataset, a corpus of NLI data from 10 distinct genres of written and spoken English. We train on the 5 genres with training data and evaluate on all 10 genres. Models are given two sentences, a premise and hypothesis, and predict whether the hypothesis is entailed by, is neutral to, or contradicts the premise.

4.2 Defining the Neighborhood

There are many different ways to define the neighborhood distribution $q(x'|x)$ from which we sample. Each choice implicitly also defines the underlying manifold with respect to which we meaure smoothness. One natural choice is to utilize data augmentations, since these are transformations over which we want our classifiers to be smooth. We compare two methods for data augmentation for image classification: RandAugment (Cubuk et al., 2020) which combines various augmentation methods at random, and random patch erasing (Zhong et al., 2020) which removes randomly sized patches from the image. We call smoothness measures based on these two augmentations **MS-RandAug** and **MS-Erase** respectively. We also consider three methods for data augmentation on natural language: SSMBA (Ng et al., 2020) which generates examples in a manifold neighborhood using a denoising autoencoder, EDA (Wei & Zou, 2019) which applies random word level operations, and backtranslation (BT) (Rico Sennrich, 2016; Xie et al., 2019) which translates back and forth from a pivot language. We call smoothness measures based on these three augmentations **MS-SSMBA**, **MS-EDA**, and **MS-BT** respectively. For all experiments we generate $n = 10$ samples for each example, although ablations in section 5.3 show that our method is relatively robust to the specific number of samples generated. We provide further details on implementations and noise levels for all methods in the Appendix A.5.

4.3 Model and Hyperparameter Space

For image classification, we use models trained for the tasks 1, 2, 4, 5, and 9 from the Predicting Generalization in Deep Learning competition (PGDL) (Jiang et al., 2020) as well as models from Jiang et al. (2019), which covers Network in Network (NiN) (Lin et al., 2013), VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2015), and CNN models trained on CIFAR10, CINIC10, and SVHN. On natural language tasks we consider CNN (Kim, 2014; Mou et al., 2016) and RoBERTa (Liu et al., 2019) based models. On natural language models we apply label noise by randomly replacing a fraction of training labels with uniform samples from the label space. We argue that label noise is not an artificial training setting as stated in Jiang et al. (2019) but rather a method of entropy regularization (Pereyra et al., 2017; Xie et al., 2016) which prevents models from becoming overconfident and decreases smoothness.

In order to control for the varying convergence rates and learning capacities of our different models, we follow Jiang et al. (2019) and early stop the training of models when they reach a given training cross entropy loss (usually around 99% training accuracy), or if they reach the max number of training epochs. We discard all models which do not converge within this time. The total number of models trained and converged in each pool as well as details on hyperparameter variations for each task and model provided in Table 3. We include further details on model training, the hyperparameter space, and specific choices in hyperparameters in Appendix A.4, A.2, and A.3.

4.4 Baselines

As a baseline comparison against our selected neighborhood distributions, we consider horizontal flips and crops for image classification and random replacement on a percentage of tokens for natural language tasks. We call these baselines **MS-FC** and **MS-Random**. Since our experimental setting makes so few assumptions, there are very few complexity measures that we can compare against. We consider complexity measures that require only model weights, specifically the **Spectral** (Yoshida & Miyato, 2017; Neyshabur et al., 2017b) and **Frobenius** (Neyshabur et al., 2015b) norms. However, in our experiments we find close to 0 or negative correlation for these measures, so we do not report their performance.

We also compare our method against output based methods that directly predict OOD generalization. We use **ATC-MC** and **ATC-NE** Garg et al. (2022) as our two baselines, which calculate a threshold on validation data based on max confidence and negative entropy scores respectively. To calculate metrics on these methods we treat the generated accuracy predictions as a score. To calculate ID τ values we select a threshold value based on validation data then calculate predicted accuracy values on test data from the same domain.

Measure	R^2	MAE	Macro τ	ID τ	Arch τ
MS-RandAug	**0.880**	**3.43**	**0.750**	**0.784**	**0.837**
MS-Erase	0.466	5.23	0.333	0.394	0.299
MS-FC	0.387	5.32	0.273	0.290	0.683
ATC-NE	0.649	4.06	0.561	0.717	0.689
ATC-MC	0.647	4.01	0.564	0.705	0.685

(a) Results on image classification. Results are averaged across all model architectures and datasets.

	CNN					RoBERTa					
Measure	R^2	MAE	Macro τ	Micro τ	ID τ	R^2	MAE	Macro τ	Micro τ	ID τ	Arch τ
MS-SSMBA	0.662	**1.93**	**0.677**	**0.689**	**0.629**	**0.972**	**1.29**	**0.832**	**0.829**	**0.838**	0.588
MS-EDA	0.641	2.04	0.664	0.649	0.611	0.968	1.45	0.830	0.810	0.830	0.512
MS-BT	0.550	2.99	0.592	0.501	0.538	0.961	1.47	0.813	0.801	0.801	0.523
MS-Random	0.409	2.64	0.544	0.554	0.439	0.967	1.27	0.821	0.816	0.822	0.537
ATC-NE	0.760	2.47	0.514	0.633	0.467	0.852	2.38	0.707	0.691	0.749	0.660
ATC-MC	**0.761**	2.46	0.517	0.634	0.467	0.869	2.26	0.722	0.705	0.749	**0.663**

(b) Results on sentiment analysis (SA) datasets.

	CNN					RoBERTa					
Measure	R^2	MAE	Macro τ	Micro τ	ID τ	R^2	MAE	Macro τ	Micro τ	ID τ	Arch τ
MS-SSMBA	0.575	2.09	0.570	**0.534**	0.704	0.933	1.19	0.750	0.730	0.771	0.301
MS-EDA	**0.577**	**2.04**	**0.581**	0.511	**0.709**	0.941	1.26	**0.789**	**0.757**	**0.799**	0.572
MS-BT	0.509	2.11	0.470	0.449	0.584	**0.944**	**1.07**	0.759	0.740	0.778	0.563
MS-Random	0.451	2.20	0.452	0.428	0.570	0.890	1.70	0.688	0.647	0.710	0.401
ATC-NE	0.576	2.52	0.568	0.446	0.705	0.737	2.22	0.557	0.541	0.739	0.635
ATC-MC	0.576	2.52	0.568	0.446	0.706	0.769	2.10	0.581	0.567	0.748	**0.636**

(c) Results on natural language inference (NLI) datasets

Table 1: Evaluation metrics measuring the correlation of our smoothness measure with OOD generalization. The best performing measures for each metric are bolded. On all tasks, our smoothness measure achieves strong generalization and beats baseline methods in almost all metrics. Full tables for R^2, Macro τ, Micro τ, and ID τ on individual train/test domains are in Appendix C

5 RESULTS

5.1 CORRELATION WITH OOD GENERALIZATION

We first present results analyzing the correlation of our proposed manifold smoothness measure with OOD generalization in Table 1. We report R^2, MAE, Macro τ, Micro τ, ID τ, and Arch τ as detailed in Section 3.3. We omit results on Spectral and Frobenius norm measures as they are close to 0 or negative for all metrics. We do not report Micro τ values for image classification models since each model type is trained on only one domain.

Image Classification Results on image classification tasks are presented in Table 1a and are averaged across all dataset domains and architectures. Our MS-RandAug significantly outperforms ATC based methods and all other measures on all metrics. MS-RandAug also exhibits only a small decrease in correlation when moving from in-domain (ID τ) to OOD datasets (Macro τ), compared to ATC methods which suffer a much larger drop. Interestingly, patch erasing and flip and crops fail to achieve strong correlations, indicating that they may poorly reflect the true image manifold.

Sentiment Analysis (SA) Results on Sentiment Analysis tasks are presented in Table 1b. In experiments on both architectures, our smoothness measures achieves strong correlation with OOD generalization on almost all metrics and beats all baselines. Of the neighborhoods tested, MS-SSMBA performs the best across both architectures. On cross architecture analysis, we observe strong Arch τ for our neighborhood measures, although they are outperformed by both ATC methods. We observe

	CNN		RoBERTa	
Measure	R^2	Micro τ	R^2	Micro τ
MS-SSMBA	**0.584**	**0.566**	**0.941**	**0.816**
MS-EDA	0.575	**0.567**	0.884	0.715
MS-BT	0.538	0.470	0.906	0.766
MS-Random	0.277	0.373	0.918	0.776
ATC-NE	0.271	0.495	0.329	0.356
ATC-MC	0.295	0.506	0.437	0.436

(a) Results on the `Drugs.com` dataset.

	CNN		RoBERTa	
Measure	R^2	Micro τ	R^2	Micro τ
MS-SSMBA	0.083	0.080	0.691	0.463
MS-EDA	0.202	0.110	**0.739**	**0.540**
MS-BT	**0.213**	**0.247**	0.730	0.527
MS-Random	0.096	0.102	0.030	0.012
ATC-NE	0.077	-0.107	0.719	0.345
ATC-MC	0.076	-0.106	0.734	0.354

(b) Results on the MedNLI dataset.

Table 2: Evaluation metrics measuring the correlation of our smoothness measure with generalization on extreme OOD datasets. MS-* methods beat baselines on both tasks, with RoBERTa models exhibiting only a slight degradation in correlation compared to more typical OOD settings.

particularly strong correlation on RoBERTa models, with a nearly perfectly linear R^2 value of 0.972 and large Micro τ of 0.829. We hypothesize that this is due to the pretrained initialization of RoBERTa models, which gives a strong inductive bias towards learning a space aligned with the one in which we measure smoothness. In contrast, CNN models are trained from random initializations and may not learn as closely aligned a space.

Natural Language Inference (NLI) Results on Natural Language Inference tasks are presented in Table 1c. Similar to our sentiment analysis results, our manifold smoothness measures achieve strong correlation with OOD generalization on both architectures and beat all baselines. Correlations in general on NLI are lower than those of sentiment analysis, perhaps because the sentence pair data manifold is more complex and difficult to learn compared to the single sentence data manifold. This also explains why we observe no single best performing neighborhood, as defining the distribution $q(x'|x)$ for sentence pairs is much more difficult. As in sentiment analysis experiments we observe exceptionally high correlation on RoBERTa models, for which we offer a similar hypothesis as above. On cross architecture analysis, we again observe strong correlation for our smoothness measures that outperform both ATC methods.

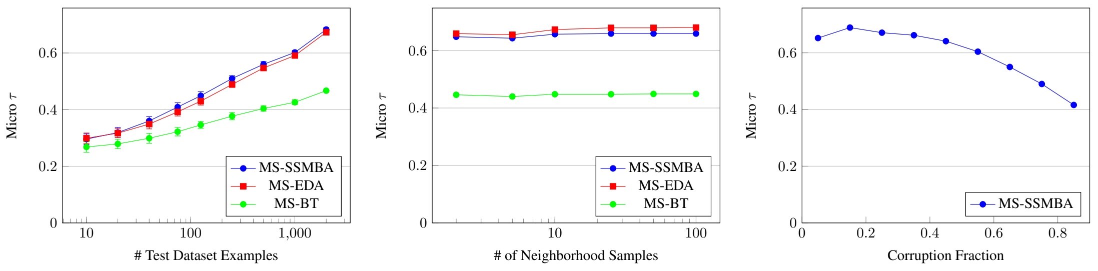
5.2 Extreme OOD Generalization

We now consider more extreme generalization to data domains with specialized and knowledge intensive data. We consider only natural language tasks as it is difficult to find a sufficiently specialized image classification dataset that maintains the same output classes. For sentiment analysis we use the `Drugs.com` review dataset (Gräßer et al., 2018), and for natural language inference we use MedNLI (Romanov & Shivade), an NLI dataset generated from clinical notes and patient history. Both these datasets contain sentences with highly specific medical terms and phrases not seen in any of our training domains. All models from all original training domains are evaluated on each of these extreme OOD domains, and we report R^2 and Micro τ. Results are shown in Table 2

On the `Drugs.com` dataset, we observe only a small decrease in correlation for smoothness methods. However, ATC methods begin to fail here, with Micro τ on RoBERTa models dropping significantly from 0.706 to 0.356. This suggests that models become poorly calibrated in extreme OOD settings, making output based OOD prediction methods fragile. On MedNLI we observe a much larger drop in correlation for both model types. For CNN models, most of our measures fail to correlate at all, and ATC methods degrade so much they became anti correlated with generalization. For RoBERTa models we observe only minor drops in correlation for all measures. Surprisingly, ATC R^2 values actually increase compared to our original experiments, although Micro τ decreases. For both tasks MS-Random exhibits almost no correlation with OOD generalization. This suggests that the choice of neighborhood becomes much more important as we move farther from our training domain.

5.3 Smoothness Estimation Ablations

In this section we examine factors that may affect the quality of the smoothness estimation generated from our measure. Since rerunning all of our experiments is too costly, we focus on evaluating CNN models trained on the sentiment analysis task. Specifically, we evaluate only on Amazon reviews of `toys` using a pool of CNN models trained on all other domains.



(a) Correlation improves as the size of the test dataset that we evaluate on increases.

(b) Correlation is constant as the number of samples decreases, even when we generate only a single example.

(c) Correlation increases until an optimal neighborhood size, then decreases as our neighborhood grows larger.

Figure 3: Micro τ for our smoothness measure calculated with varying ablations on CNN models evaluated on Amazon `toy` reviews.

Test Dataset Size: Does our manifold smoothness measure still correlate well when the test dataset is small? We randomly and iteratively subsample our test dataset of 2000 examples to reduce our dataset size down to 10 examples. We then measure our models' smoothness on each subsampled dataset and calculate the Micro τ on all models. Results are shown in Figure 3a. We find that for all neighborhoods, smaller datasets lead to noisier smoothness estimates and lower correlation. As dataset size increases, correlation increases as well.

Number of Samples Generated: How many examples do we need to sample from our neighborhood in order to generate a reliable smoothness estimate? We generate a varying number of samples for each test example, from a minimum of a single example to a maximum of 100 examples, then estimate our smoothness measure with each set of samples and calculate the micro τ. Results are shown in Figure 3b. We find that our measure is surprisingly robust to the number of samples, with only a small difference between 100 samples and 1 sample. For all measures, correlation slightly increases as the number of samples increases and we achieve a better estimation of the true smoothness value.

Neighborhood Size: Does the quality of our measure depend on how large of a neighborhood we sample from? We define the neighborhood size as the level of noise in our neighborhood distribution $q(x'|x)$, since a noisier distribution will generate points farther from x. For our experiments we use MS-SSMBA and vary the corruption percentage from a minimum of 5% to a maximum of 85%. After generating sets of neighborhood examples from each distribution, we estimate our models' smoothness on each one and calculate the micro τ on all models. Results are shown in Figure 3c. We find that as we begin to increase the neighborhood size, correlation begins to increase as well. Correlation quickly reaches a maximum, then decreases as we continue to increase our neighborhood size. However, even at a 85% corruption, our method is quite robust and achieves a micro τ of 0.416.

6 Discussion

In this paper, motivated by the limited settings in which existing complexity measures can be applied, we develop a *local manifold smoothness* measure that can be applied even when test distributions are unknown and model training data, weights, and gradients are unavailable. We evaluate our method on image classification, sentiment analysis, and natural language inference datasets, calculating a variety of correlation metrics on both in-domain and out-of-domain test sets. Across all tasks and experimental settings, we find that our smoothness measure outperforms baseline methods in all metrics except one. However, our method has several limitations. We cannot compare against other complexity measures since they can't be applied in our OOD experimental settings. Local manifold smoothness also depends on access to a neighborhood distribution $q(x'|x)$ from which we can sample. In settings where $q(x'|x)$ is difficult to define, our method may not be applicable or provide inappropriately high estimates, so practitioners must be careful to verify their estimates on a labelled test set. In addition, our smoothness measure may fail in sufficiently OOD settings where a model may become poorly calibrated or constant, although we find in practice even extreme OOD settings are similar enough for our measure to perform well. The relationship between smoothness and generalization is also not perfectly linear, as a maximally smooth constant classifier cannot achieve maximal generalization. In future work we plan to explore applying smoothness as a method for OOD detection.

REFERENCES

Sumukh Aithal K, Dhruva Kashyap, and Natarajan Subramanyam. Robustness to Augmentations as a Generalization metric. *arXiv e-prints*, art. arXiv:2101.06459, January 2021.

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and Generalization in Overparameterized Neural Networks, Going Beyond Two Layers. *arXiv e-prints*, art. arXiv:1811.04918, November 2018.

Michael Arbel, Danica J Sutherland, Mikoł aj Bińkowski, and Arthur Gretton. On gradient regularizers for mmd gans. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper/2018/file/07f75d9144912970de5a09f5a305e10c-Paper.pdf`.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv*, 2019.

Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *J. Mach. Learn. Res.*, 3(null):463–482, mar 2003. ISSN 1532-4435.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/b22b257ad0519d4500539da3c8bcf4dd-Paper.pdf`.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In B. Schölkopf, J. Platt, and T. Hoffman (eds.), *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2007. URL `https://proceedings.neurips.cc/paper/2006/file/b1b0432ceafb0ce714426e9114852ac7-Paper.pdf`.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. *Semi-Supervised Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006. ISBN 0262033585.

Mayee Chen*, Karan Goel*, Nimit Sohoni*, Fait Poms, Kayvon Fatahalian, and Christopher Re. Mandoline: Model evaluation under distribution shift. *International Conference of Machine Learning (ICML)*, 2021.

Minshuo Chen, Haoming Jiang, Wenjing Liao, and Tuo Zhao. Nonparametric Regression on Low-Dimensional Manifolds using Deep ReLU Networks : Function Approximation and Statistical Recovery. *arXiv e-prints*, art. arXiv:1908.01842, August 2019.

Ching-Yao Chuang, Antonio Torralba, and Stefanie Jegelka. Estimating generalization under distribution shifts via domain-invariant representations. *International conference on machine learning*, 2020.

Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18613–18624. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/d85b63ef0ccb114d0a3bb7b7d808028f-Paper.pdf`.

Luke N. Darlow, Elliot J. Crowley, Antreas Antoniou, and Amos J. Storkey. CINIC-10 is not ImageNet or CIFAR-10. *arXiv e-prints*, art. arXiv:1810.03505, October 2018.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Weijian Deng and Liang Zheng. Are labels always necessary for classifier accuracy evaluation? In *Proc. CVPR*, 2021.

Weijian Deng, Stephen Gould, and Liang Zheng. What does rotation prediction tell us about classifier accuracy under varying testing environments? In *ICML*, 2021.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the 33rd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017.

Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/42998cf32d552343bc8e460416382dca-Paper.pdf.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-Aware Minimization for Efficiently Improving Generalization. art. arXiv:2010.01412, 2020.

Saurabh Garg, Sivaraman Balakrishnan, Zachary C. Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging Unlabeled Data to Predict Out-of-Distribution Performance. *arXiv e-prints*, art. arXiv:2201.04234, January 2022.

Vikas Garg, Adam Tauman Kalai, Katrina Ligett, and Steven Wu. Learn to expect the unexpected: Probably approximately correct domain generalization. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 3574–3582. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/garg21a.html.

Stanton A Glantz, Bryan K Slinker, and Torsten B Neilands. *Primer of Applied Regression and Analysis of Variance*. Health Professions Division, McGraw-Hill, New York, 1990.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv 1412.6572*, 12 2014.

Felix Gräßer, Surya Kallumadi, Hagen Malberg, and Sebastian Zaunseder. Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. In *Proceedings of the 2018 International Conference on Digital Health*, DH '18, pp. 121–125, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450364935. doi: 10.1145/3194658.3194677. URL https://doi.org/10.1145/3194658.3194677.

Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. Predicting with Confidence on Unseen Distributions. In *International Conference on Computer Vision*, 2021.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 5769–5779, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Abhishek Gupta, Alagan Anpalagan, Ling Guan, and Ahmed Shaharyar Khwaja. Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues. *Array*, 10:100057, 2021. ISSN 2590-0056. doi: https://doi.org/10.1016/j.array.2021.100057. URL https://www.sciencedirect.com/science/article/pii/S2590005621000059.

Jeff Z. HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable Guarantees for Self-Supervised Deep Learning with Spectral Contrastive Loss. *arXiv e-prints*, art. arXiv:2106.04156, June 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv e-prints*, art. arXiv:1512.03385, December 2015.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2744–2751, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.244. URL https://aclanthology.org/2020.acl-main.244.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. In *Association for Computational Linguistics*, 2020b.

Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJlQfnCqKX.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic Generalization Measures and Where to Find Them. *arXiv e-prints*, art. arXiv:1912.02178, December 2019.

Yiding Jiang, Pierre Foret, Scott Yak, Daniel M. Roy, Hossein Mobahi, Gintare Karolina Dziugaite, Samy Bengio, Suriya Gunasekar, Isabelle Guyon, and Behnam Neyshabur. NeurIPS 2020 Competition: Predicting Generalization in Deep Learning. *arXiv e-prints*, art. arXiv:2012.07976, December 2020.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J. Zico Kolter. Assessing Generalization of SGD via Disagreement. *arXiv e-prints*, art. arXiv:2106.13799, June 2021.

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938. ISSN 00063444. URL http://www.jstor.org/stable/2332226.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL http://arxiv.org/abs/1412.6980. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton A. Earnshaw, Imran S. Haque, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning (ICML)*, 2021.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 888–896. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/liang19a.html.

Weixin Liang and James Zou. Metashift: A dataset of datasets for evaluating contextual distribution shifts and training conflicts. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=MTex8qKavoS.

Min Lin, Qiang Chen, and Shuicheng Yan. Network In Network. *arXiv e-prints*, art. arXiv:1312.4400, December 2013.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Shangyun Lu, Bradley Nott, Aaron Olson, Alberto Todeschini, Hossein Vahabi, Yair Carmon, and Ludwig Schmidt. Harder or different? a closer look at distribution shift in dataset reproduction. In *ICML Workshop on Uncertainty and Robustness in Deep Learning*, 2020.

Yucen Luo, Jun Zhu, Mengxi Li, Yong Ren, and Bo Zhang. Smooth neighbors on teacher graphs for semi-supervised learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8896–8905, 2018. doi: 10.1109/CVPR.2018.00927.

David A. McAllester. Pac-bayesian model averaging. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, COLT '99, pp. 164–170, New York, NY, USA, 1999. Association for Computing Machinery. ISBN 1581131674. doi: 10.1145/307400.307435. URL `https://doi.org/10.1145/307400.307435`.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International COnference on Learning Representations*, 2018.

Peyman Morteza and Yixuan Li. Provable guarantees for understanding out-of-distribution detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7831–7840, Jun. 2022. doi: 10.1609/aaai.v36i7.20752. URL `https://ojs.aaai.org/index.php/AAAI/article/view/20752`.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 130–136, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-2022. URL `https://aclanthology.org/P16-2022`.

Vaishnavh Nagarajan and J Zico Kolter. Generalization in deep networks: The role of distance from initialization. In *NeurIPS Workshop on Deep Learning: Bridging Theory and Practice*, 2019.

Ryumei Nakada and Masaaki Imaizumi. Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *Journal of Machine Learning Research*, 21(174):1–38, 2020. URL `http://jmlr.org/papers/v21/20-002.html`.

Yuval Netzer, Tao Wang, Adam Coates, A. Bissacco, Bo Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Behnam Neyshabur, Russ R Salakhutdinov, and Nati Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015a. URL `https://proceedings.neurips.cc/paper/2015/file/eaa32c96f620053cf442ad32258076b9-Paper.pdf`.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In Peter Grünwald, Elad Hazan, and Satyen Kale (eds.), *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 1376–1401, Paris, France, 03–06 Jul 2015b. PMLR. URL `https://proceedings.mlr.press/v40/Neyshabur15.html`.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. In *Proceeding of NeurIPS*, 2017a.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations*, 2017b.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pp. 314–319, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5333. URL `https://aclanthology.org/W19-5333`.

Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. Ssmba: Self-supervised manifold based data augmentation for improving out-of-domain robustness. In *Proc. of EMNLP*, 2020.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fined-grained aspects. In *Proceedings of EMNLP*, 2019.

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

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pp. 506–519, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349444. doi: 10.1145/3052973.3053009. URL https://doi.org/10.1145/3052973.3053009.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. In *ICLR*, 2017.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? 2018. https://arxiv.org/abs/1806.00451.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5389–5400. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/recht19a.html.

Alexandra Birch Rico Sennrich, Barry Haddow. Improving neural machine translation models with monolingual data. In *Proc. of ACL*, 2016.

Salah Rifai, Yann N Dauphin, Pascal Vincent, Yoshua Bengio, and Xavier Muller. The manifold tangent classifier. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper/2011/file/d1f44e2f09dc172978a4d3151d11d63e-Paper.pdf.

Alexey Romanov and Chaitanya Shivade. Lessons from natural language inference in the clinical domain. URL http://arxiv.org/abs/1808.06752.

Mihaela Rosca, Theophane Weber, Arthur Gretton, and Shakir Mohamed. A case for new neural network smoothness constraints. In *NeurIPS ICBINB Workshop*, 2020.

Yair Schiff, Brian Quanz, Payel Das, and Pin-Yu Chen. Predicting deep neural network generalization with perturbation response curves. In *Neural Information Processing Systems*, 2021.

Johannes Schmidt-Hieber. Deep ReLU network approximation of functions on a manifold. *arXiv e-prints*, art. arXiv:1908.00695, August 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations*, 2015.

Taiji Suzuki. Adaptivity of deep relu network for learning in besov and mixed smooth besov spaces: optimal rate and curse of dimensionality. In *International Conference on Learning Representations*, 2018.

Taiji Suzuki and Atsushi Nitanda. Deep learning is adaptive to intrinsic dimensionality of model smoothness in anisotropic besov space. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 3609–3621. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/1dacb10f0623c67cb7dbb37587d8b38a-Paper.pdf.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18583–18599. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/d8330f857a17c53d217014ee776bfd50-Paper.pdf.

V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971. doi: 10.1137/1116025. URL https://doi.org/10.1137/1116025.

Ramakrishna Vedantam, David Lopez-Paz, and David J. Schwab. An empirical investigation of domain generalization with empirical risk minimizers. In *Neural Information Processing Systems*, 2021.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6438–6447. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/verma19a.html`.

Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 6383–6389, Hong Kong, China, November 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D19-1670`.

J. Wiens, S. Saria, M. Sendak, M. Ghassemi, V. Liu, F. Doshi-Velez, K. Jung, K. Heller, D. Kale, M. Saeed, P. Ossorio, S. Thadaney-Israni, and A. Goldenberg. Do no harm: A roadmap for responsible machine learning for healthcare. *Nature Medicine*, 25(10):1337–1340, 2019.

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122. Association for Computational Linguistics, 2018. URL `http://aclweb.org/anthology/N18-1101`.

Lingxi Xie, Jingdong Wang, Zhen Wei, Meng Wang, and Qi Tian. DisturbLabel: Regularizing CNN on the Loss Layer. *arXiv e-prints*, art. arXiv:1605.00055, April 2016.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

Yuichi Yoshida and Takeru Miyato. Spectral Norm Regularization for Improving the Generalizability of Deep Learning. *arXiv e-prints*, art. arXiv:1705.10941, May 2017.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv e-prints*, art. arXiv:1611.03530, November 2016.

Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7404–7413. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/zhang19i.html`.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):13001–13008, Apr. 2020. doi: 10.1609/aaai.v34i07.7000. URL `https://ojs.aaai.org/index.php/AAAI/article/view/7000`.

Model	Dataset	Training Domain	Batch Size	Depth	Width	Dropout	Weight Decay	Label Noise	Learning Rate	Batch Norm	Seed	Data Aug	# Trained	# Converged
CNN	Amazon	10	3	3	3	3	3	—	—	—	—	—	2430	2418
	MNLI	5	3	3	3	2	—	3	—	—	—	—	910	796
RoBERTa	Amazon	10	3	—	—	2	3	3	—	—	—	—	540	332
	MNLI	5	3	—	—	2	3	3	—	—	—	—	270	213
NiN	SVHN	—	3	3	—	3	2	—	—	—	—	—	54	54
	CIFAR10	—	2	2	2	2	2	—	—	—	—	—	32	32
VGG	CIFAR10	—	3	3	—	3	2	—	—	—	—	—	54	54
ResNet	CIFAR10	—	—	—	3	—	3	—	2	2	3	2	216	216
CNN	CINIC10	—	2	2	4	—	2	—	2	2	—	—	128	128

Table 3: Number of possible hyperparameter values for each architecture and task. Fields denoted with a — indicate that this hyperparameter is fixed or not applicable. We also list the total number of models trained and converged in each model pool. In total we evaluate 4,027 models.

A Experimental Setup

In this section we present full details of our experimental setup, including data preprocessing and specifics on model architecture and hyperparameter space. All models are trained on a single RTX6000 GPU.

A.1 Data Preprocessing

Image classification datasets are preprocessed by normalizing pixel values and resizing to 32×32 if necessary. We use the same preprocessing steps for sentiment analysis and NLI experiments. All data is first tokenized using a GPT-2 style tokenizer and BPE vocabulary provided by `fairseq` (Ott et al., 2019). This BPE vocabulary consists of 50263 types. Corresponding labels are encoded using a label dictionary consisting of as many types as there are classes. Input text and labels are then binarized for model training.

A.2 Model Architecture

Our image classification models are Network in Network (NiN) (Lin et al., 2013), VGG (Simonyan & Zisserman, 2015), and CNN models. Training and hyperparemeter details for these models are provided in Jiang et al. (2020).

Our CNN models are based on the architecture in Kim (2014). Our input embeddings are 512 dimensional, which we treat as our channel dimension. Our base model applies a set of three one dimensional convolutions of kernel size 3, 4, and 5 with 256 output channels. We modulate the number of stacked convolutions (depth) as well as the channel size (width). Each convolution generates a separate representation that is max pooled across the sequence and concatenated together. We feed this representation into a MLP classifier with a single hidden layer of 512 dimensions. We apply dropout of 0.2 to our inputs and MLP classifier.

Our RoBERTa models use a pre-trained RoBERTa$_{\text{BASE}}$ model provided by `fairseq`. Classification token embeddings are fed into an MLP classifier with a single hidden layer of 512 dimensions. All models are written within the `fairseq` framework (Ott et al., 2019) and trained on a single RTX6000 or T4 GPU.

A.3 Model Hyperparameters

Hyperparameter values for image classification models are provided in Jiang et al. (2020). For natural language models we vary the following hyperparameters: training domain, batch size, depth, width, dropout, weight decay, and label noise. For training domains, on sentiment analysis we choose between `books, clothing, home, kindle, movies, pets, sports, tech, tools, toys`. For training domains on NLI, we choose between `slate, government, fiction, telephone, travel`. NLI datasets include additional test sets `oup, nineeleven, facetoface, verbatim, letters`. Possible values for all other hyperparameters are provided in Table 4

	CNN		RoBERTa	
Hyperparameter	SA	NLI	SA	NLI
Batch Size	{32, 64, 128}	{32, 64, 128 }	{8, 16, 32}	{8, 16, 32}
Depth	{1, 2, 3 }	{1, 2, 3 }	1	1
Width	{128, 256, 512}	{128, 256, 512 }	768	768
Dropout	{0.0, 0.25, 0.5}	{0.0, 0.25}	{0.0, 0.1}	{0.0, 0.1}
Weight Decay	{0.0, 0.0001, 0.0005}	0.0	{0.0, 0.0001, 0.0005}	{0.0, 0.0001, 0.0005}
Label Noise	0.0	{0.0, 0.2, 0.4}	{0.0, 0.2, 0.4}	{0.0, 0.2, 0.4}

Table 4: Possible hyperparameter values for each architecture and task.

A.4 Model Training

All models are trained with the Adam optimizer (Kingma & Ba, 2014) with $\beta = (0.9, 0.98)$ and $\epsilon = 1 \times 10^{-6}$. CNN models are trained with learning rate 1×10^{-3} and RoBERTa models are trained with learning rate 1×10^{-5}. We use a inverse square root learning rate scheduler to anneal learning rate over training. We early stop CNN models on sentiment analysis at 0.04 cross entropy and on NLI at 0.03 cross entropy. We early stop RoBERTa models on sentiment analysis at 0.05 cross entropy and on NLI at 0.3 cross entropy. Training details for image classification models are provided in Jiang et al. (2020).

A.5 Neighborhood Hyperparameters

We select hyperparameters for our neighborhood distributions based on best practices provided in their respective papers. For RandAugment we use a magnitude parameter of 15 and augment only once. For FlipCrop we flip the image horizontally with probability 50% and generate a crop with a lower bound of 8% of total image area and an aspect ratio between 3/4 and 4/3. Images are resized to 32×32 after cropping. For Erase we select a region to erase with a maximum size of 33% of the total image area and an aspect ratio between 1/3 and 10/3.

For SSMBA, we select 15% of tokens to corrupt, leaving 10% of them unmasked, randomly replacing 10% of them, and masking the remaining 80%. We reconstruct these corrutped sentences with a RoBERTa base model (Liu et al., 2019). For EDA, we apply all operations on 10% of tokens. For backtranslation, we use English to German and German to English models provided in (Ng et al., 2019). For our random baseline, we randomly replace 15% of tokens.

B Additional Experiments

B.1 Norm-Based Complexity Measures

Following (Jiang et al., 2019), we calculate our spectral norm measure as $\Pi_{i=1}^{d}||\mathbf{W}_i||_2^2$ and Frobenius norm measure $\Pi_{i=1}^{d}||\mathbf{W}_i||_F^2$. We do not list results on these measures as the correlations are often negative or 0.

B.2 Cross-Domain Correlation

In this set of experiments we measure the correlation between smoothness and generalization values of a single model trained on a single training domain evaluated across different OOD test domains. For image classification experiments we specify both the training domain and model architecture. For natural language experiments we call these the **CNN** τ and **Roberta** τ when averaged respectively across CNN and RoBERTa models and training domains. Results are presented in Table 5b.

For image classification tasks MS-RandAug achieves strong correlation on all models and datasets. However, for natural language tasks, our smoothness measure performs quite poorly on both models, although they still outperforms ATC baselines. We hypothesize different regions of the input space may have different optimal levels of smoothness that achieve the lowest generalization error. Our value of interest is then not the absolute smoothness, but the *relative* smoothness compared to this optimal value. These values are the same when comparing different models evaluated on the same

	Train Domain (Model)			
Measure	SVHN (NiN)	CIFAR10 (NiN)	CIFAR10 (VGG)	CINIC10 (CNN)
MS-RandAug	0.641	**0.491**	**0.784**	**0.747**
MS-Erase	-0.139	0.216	-0.150	0.564
MS-FC	-0.304	0.011	0.557	0.382
ATC-NE	0.635	0.478	0.531	0.668
ATC-MC	**0.676**	0.444	0.530	0.660

(a) Results on image classification tasks.

Task	Measure	CNN τ	RoBERTa τ
SA	MS-SSMBA	0.360	0.010
	MS-EDA	0.431	**0.266**
	MS-BT	0.505	0.245
	MS-Random	**0.570**	0.150
	ATC-NE	0.543	0.228
	ATC-MC	0.539	0.224
NLI	MS-SSMBA	0.022	0.260
	MS-EDA	0.102	0.335
	MS-BT	0.089	0.333
	MS-Random	**0.226**	**0.440**
	ATC-NE	0.219	0.231
	ATC-MC	0.223	0.239

(b) Results on natural language tasks.

Table 5: Correlation metrics evaluating the ability of our smoothness measure to predict OOD generalization across test datasets. Our smoothness measures achieves strong correlation in image classification tasks but fails in natural language tasks.

domain, but are not the same for the same model evaluated on different domains, making correlating across domains difficult. On the natural image manifold where domains are more well behaved and uniform compared to the natural language manifold, the relative smoothness may not differ much between domains allowing us to correlate our measure across domains.

B.3 Negative Entropy Results

A full table of results including metrics calculated on smoothness measured with negative entropy is provided in Table 6. We refer to measures calculated with entropy as **MSE-SSMBA**, **MSE-EDA**, **MSE-BT**, and **MSE-Random**. For most metrics, MSE-* methods perform similarly or slightly worse.

		CNN				RoBERTa			
Task	Measure	R^2	MAE	Macro τ	Micro τ	R^2	MAE	Macro τ	Micro τ
SA	MS-SSMBA	**0.662**	1.93	**0.677**	**0.689**	**0.972**	1.29	**0.832**	**0.829**
	MS-EDA	0.641	2.04	0.664	0.649	0.968	1.45	0.830	0.810
	MS-BT	0.550	2.99	0.592	0.501	0.961	1.47	0.813	0.801
	MS-Random	0.409	2.64	0.544	0.554	0.967	**1.27**	0.821	0.816
	MSE-SSMBA	0.595	2.53	0.708	0.713	0.971	1.32	0.830	0.824
	MSE-EDA	0.534	2.71	0.698	0.674	0.965	1.55	0.825	0.801
	MSE-BT	0.471	3.59	0.618	0.541	0.961	1.46	0.813	0.799
	MSE-Random	0.283	3.37	0.570	0.552	0.964	1.34	0.818	0.809
	ATC-NE	0.530	3.80	0.506	0.642	0.849	3.59	0.684	0.706
	ATC-MC	0.528	3.76	0.507	0.642	0.863	3.54	0.698	0.716
NLI	MS-SSMBA	0.575	2.09	0.570	**0.534**	0.933	1.19	0.750	0.730
	MS-EDA	0.577	**2.04**	0.581	0.511	0.941	1.26	**0.789**	**0.757**
	MS-BT	0.509	2.11	0.470	0.449	0.944	1.07	0.759	0.740
	MS-Random	0.451	2.20	0.452	0.428	0.890	1.70	0.688	0.647
	MSE-SSMBA	0.588	2.20	0.579	0.520	0.941	1.39	0.738	0.711
	MSE-EDA	**0.606**	2.11	**0.597**	0.512	0.937	1.48	0.767	0.732
	MSE-BT	0.536	2.27	0.480	0.422	**0.954**	**1.12**	0.764	0.750
	MSE-Random	0.457	2.36	0.451	0.397	0.904	1.79	0.665	0.591
	ATC-NE	0.378	3.57	0.430	0.294	0.673	2.35	0.536	0.52
	ATC-MC	0.382	3.57	0.433	0.297	0.718	2.21	0.570	0.556

Table 6: Correlation metrics evaluating the quality of our smoothness measure on two tasks, sentiment analysis and natural language inference, and two architectures, CNN and RoBERTa. Details on metric calculations and baselines are provided in sections 3.3 and 4.4. This full table of results includes metrics calculated with our negative entropy smoothness measure as well.

C Full Results

We provide a breakdown of results on the correlation metrics R^2 (Tables 7, 8, 9, 10, 11, 12), macro τ (Tables 13, 14, 15, 16, 17, 18, micro τ (Tables 19, 20), and ID τ (Tables 21, 22, 23) for each set of datasets and models.

			Test Domain		
Model	Train Domain	Measure	SVHN	Colored MNIST	MNIST
NiN	SVHN	MS-RandAug	—	0.785	0.744
		MS-Erase	—	0.088	0.218
		MS-FC	—	0.134	0.283
		ATC-NE	—	0.506	0.725
		ATC-MC	—	0.615	0.769

Table 7: Full R^2 metrics for all test domains for image classification models on SVHN, Colored MNIST, and MNIST.

			Test Domain			
Model	Train Domain	Measure	CIFAR10	CINIC10	CIFAR10.1	CIFAR10.2
NiN	CIFAR10	MS-RandAug	—	0.898	0.927	0.876
		MS-Erase	—	0.404	0.526	0.547
		MS-FC	—	0.109	0.028	0.000
		ATC-NE	—	0.237	0.575	0.435
		ATC-MC	—	0.252	0.538	0.390
ResNet	CIFAR10	MS-RandAug	—	0.816	0.844	0.853
		MS-Erase	—	0.807	0.812	0.783
		MS-FC	—	0.726	0.628	0.660
		ATC-NE	—	0.736	0.782	0.693
		ATC-MC	—	0.702	0.760	0.680
VGG	CIFAR10	MS-RandAug	—	0.969	0.950	0.929
		MS-Erase	—	0.096	0.100	0.104
		MS-FC	—	0.637	0.524	0.487
		ATC-NE	—	0.557	0.774	0.724
		ATC-MC	—	0.559	0.764	0.709
CNN	CINIC10	MS-RandAug	0.922	—	0.876	0.865
		MS-Erase	0.603	—	0.504	0.548
		MS-FC	0.416	—	0.397	0.395
		ATC-NE	0.869	—	0.750	0.724
		ATC-MC	0.868	—	0.741	0.716

Table 8: Full R^2 metrics for all test domains for image classification models on CIFAR10, CINIC10, CIFAR10.1, and CIFAR10.2.

Train Domain	Measure	Test Domain books	clothing	home	kindle	movies	pets	sports	tech	tools	toys
books	MS-SSMBA	—	0.688	0.771	0.752	0.683	0.823	0.762	0.697	0.711	0.729
	MS-EDA	—	0.720	0.725	0.699	0.689	0.810	0.693	0.663	0.727	0.805
	MS-BT	—	0.560	0.663	0.639	0.593	0.675	0.602	0.591	0.592	0.658
	MS-Random	—	0.413	0.418	0.331	0.343	0.479	0.373	0.304	0.401	0.496
	ATC-NE	—	0.765	0.859	0.828	0.777	0.834	0.834	0.791	0.814	0.872
	ATC-MC	—	0.759	0.854	0.821	0.771	0.828	0.830	0.790	0.808	0.871
clothing	MS-SSMBA	0.517	—	0.601	0.531	0.525	0.535	0.566	0.484	0.549	0.631
	MS-EDA	0.409	—	0.419	0.452	0.408	0.395	0.457	0.388	0.486	0.534
	MS-BT	0.234	—	0.164	0.267	0.273	0.121	0.191	0.164	0.193	0.355
	MS-Random	0.236	—	0.237	0.223	0.250	0.212	0.213	0.156	0.246	0.335
	ATC-NE	0.480	—	0.673	0.478	0.467	0.466	0.739	0.299	0.772	0.847
	ATC-MC	0.477	—	0.675	0.477	0.466	0.474	0.742	0.300	0.772	0.848
home	MS-SSMBA	0.533	0.545	—	0.524	0.511	0.665	0.648	0.576	0.604	0.592
	MS-EDA	0.412	0.554	—	0.451	0.433	0.586	0.496	0.417	0.525	0.657
	MS-BT	0.313	0.260	—	0.327	0.316	0.346	0.366	0.304	0.385	0.408
	MS-Random	0.273	0.260	—	0.236	0.325	0.286	0.299	0.270	0.282	0.370
	ATC-NE	0.608	0.861	—	0.675	0.618	0.838	0.838	0.720	0.863	0.894
	ATC-MC	0.612	0.861	—	0.679	0.626	0.838	0.839	0.719	0.863	0.895
kindle	MS-SSMBA	0.645	0.680	0.650	—	0.626	0.765	0.659	0.551	0.556	0.634
	MS-EDA	0.722	0.688	0.695	—	0.662	0.785	0.690	0.658	0.623	0.783
	MS-BT	0.655	0.674	0.684	—	0.625	0.745	0.704	0.695	0.649	0.735
	MS-Random	0.325	0.364	0.253	—	0.269	0.442	0.292	0.244	0.274	0.445
	ATC-NE	0.747	0.659	0.701	—	0.690	0.792	0.717	0.505	0.610	0.776
	ATC-MC	0.759	0.642	0.699	—	0.687	0.784	0.708	0.507	0.594	0.765
movies	MS-SSMBA	0.541	0.640	0.658	0.615	—	0.633	0.656	0.583	0.653	0.741
	MS-EDA	0.542	0.662	0.571	0.676	—	0.629	0.594	0.574	0.636	0.789
	MS-BT	0.602	0.679	0.677	0.653	—	0.723	0.739	0.700	0.709	0.754
	MS-Random	0.194	0.276	0.281	0.279	—	0.228	0.316	0.216	0.316	0.369
	ATC-NE	0.698	0.719	0.708	0.668	—	0.718	0.707	0.611	0.752	0.818
	ATC-MC	0.707	0.725	0.713	0.664	—	0.732	0.711	0.624	0.755	0.820
pets	MS-SSMBA	0.458	0.543	0.578	0.529	0.548	—	0.606	0.528	0.590	0.721
	MS-EDA	0.426	0.552	0.554	0.467	0.513	—	0.546	0.463	0.549	0.677
	MS-BT	0.485	0.523	0.477	0.549	0.563	—	0.581	0.550	0.580	0.587
	MS-Random	0.260	0.333	0.284	0.302	0.320	—	0.304	0.271	0.345	0.432
	ATC-NE	0.641	0.851	0.849	0.651	0.680	—	0.825	0.591	0.812	0.883
	ATC-MC	0.644	0.846	0.848	0.648	0.683	—	0.824	0.589	0.810	0.883
sports	MS-SSMBA	0.499	0.530	0.569	0.524	0.463	0.573	—	0.517	0.590	0.721
	MS-EDA	0.464	0.514	0.460	0.489	0.381	0.508	—	0.448	0.541	0.594
	MS-BT	0.406	0.295	0.334	0.383	0.381	0.325	—	0.331	0.393	0.411
	MS-Random	0.283	0.173	0.212	0.256	0.219	0.239	—	0.204	0.231	0.331
	ATC-NE	0.712	0.900	0.926	0.744	0.656	0.891	—	0.828	0.931	0.953
	ATC-MC	0.723	0.897	0.926	0.754	0.665	0.896	—	0.833	0.933	0.952
tech	MS-SSMBA	0.610	0.508	0.635	0.620	0.596	0.604	0.636	—	0.545	0.848
	MS-EDA	0.550	0.475	0.519	0.603	0.608	0.581	0.587	—	0.532	0.649
	MS-BT	0.588	0.447	0.572	0.632	0.597	0.558	0.603	—	0.523	0.648
	MS-Random	0.340	0.241	0.262	0.307	0.337	0.285	0.291	—	0.200	0.432
	ATC-NE	0.766	0.820	0.904	0.791	0.817	0.866	0.874	—	0.882	0.893
	ATC-MC	0.771	0.823	0.904	0.794	0.817	0.864	0.875	—	0.882	0.893
tools	MS-SSMBA	0.554	0.505	0.548	0.556	0.605	0.651	0.606	0.577	—	0.660
	MS-EDA	0.466	0.413	0.460	0.459	0.589	0.587	0.512	0.465	—	0.593
	MS-BT	0.451	0.385	0.447	0.476	0.482	0.501	0.483	0.484	—	0.462
	MS-Random	0.323	0.241	0.204	0.306	0.399	0.316	0.253	0.227	—	0.297
	ATC-NE	0.661	0.875	0.901	0.679	0.719	0.867	0.908	0.795	—	0.916
	ATC-MC	0.670	0.874	0.903	0.684	0.729	0.866	0.909	0.801	—	0.916
toys	MS-SSMBA	0.625	0.693	0.656	0.620	0.643	0.661	0.708	0.618	0.650	—
	MS-EDA	0.473	0.582	0.487	0.481	0.498	0.552	0.535	0.429	0.528	—
	MS-BT	0.311	0.408	0.331	0.351	0.324	0.355	0.420	0.334	0.357	—
	MS-Random	0.272	0.307	0.234	0.254	0.286	0.257	0.301	0.215	0.248	—
	ATC-NE	0.686	0.936	0.855	0.742	0.712	0.838	0.885	0.561	0.866	—
	ATC-MC	0.691	0.935	0.858	0.742	0.718	0.843	0.886	0.574	0.869	—

Table 9: Full R^2 metrics for all pairs of training and test domains for CNN models trained on AWS.

		Test Domain									
Train Domain	Measure	`books`	`clothing`	`home`	`kindle`	`movies`	`pets`	`sports`	`tech`	`tools`	`toys`
`books`	MS-SSMBA	—	0.973	0.983	0.987	0.963	0.973	0.985	0.985	0.977	0.968
	MS-EDA	—	0.961	0.970	0.982	0.966	0.966	0.970	0.957	0.978	0.981
	MS-BT	—	0.946	0.979	0.985	0.961	0.972	0.972	0.975	0.970	0.955
	MS-Random	—	0.956	0.970	0.981	0.958	0.967	0.970	0.973	0.952	0.961
	ATC-NE	—	0.845	0.861	0.957	0.927	0.940	0.852	0.925	0.853	0.876
	ATC-MC	—	0.838	0.842	0.947	0.924	0.929	0.840	0.927	0.845	0.876
`clothing`	MS-SSMBA	0.944	—	0.974	0.942	0.980	0.981	0.984	0.973	0.969	0.978
	MS-EDA	0.974	—	0.970	0.922	0.955	0.978	0.961	0.971	0.953	0.972
	MS-BT	0.944	—	0.977	0.921	0.971	0.983	0.982	0.980	0.970	0.993
	MS-Random	0.950	—	0.965	0.933	0.982	0.977	0.985	0.981	0.974	0.973
	ATC-NE	0.792	—	0.934	0.764	0.913	0.939	0.933	0.846	0.914	0.973
	ATC-MC	0.801	—	0.945	0.789	0.917	0.948	0.933	0.887	0.926	0.971
`home`	MS-SSMBA	0.973	0.972	—	0.976	0.972	0.979	0.976	0.980	0.972	0.984
	MS-EDA	0.961	0.975	—	0.967	0.959	0.960	0.949	0.917	0.963	0.979
	MS-BT	0.947	0.958	—	0.944	0.947	0.948	0.977	0.972	0.971	0.974
	MS-Random	0.978	0.976	—	0.970	0.967	0.975	0.973	0.964	0.975	0.983
	ATC-NE	0.878	0.959	—	0.958	0.908	0.930	0.962	0.860	0.928	0.951
	ATC-MC	0.885	0.960	—	0.963	0.913	0.945	0.964	0.872	0.938	0.952
`kindle`	MS-SSMBA	0.992	0.963	0.966	—	0.961	0.957	0.963	0.971	0.943	0.984
	MS-EDA	0.983	0.968	0.951	—	0.966	0.962	0.953	0.974	0.963	0.969
	MS-BT	0.973	0.947	0.975	—	0.954	0.949	0.954	0.973	0.923	0.973
	MS-Random	0.979	0.946	0.956	—	0.967	0.961	0.946	0.970	0.934	0.975
	ATC-NE	0.931	0.380	0.510	—	0.861	0.162	0.260	0.424	0.179	0.608
	ATC-MC	0.930	0.534	0.623	—	0.871	0.294	0.392	0.551	0.316	0.672
`movies`	MS-SSMBA	0.984	0.976	0.969	0.981	—	0.962	0.983	0.958	0.961	0.971
	MS-EDA	0.974	0.975	0.947	0.985	—	0.928	0.977	0.950	0.972	0.969
	MS-BT	0.971	0.952	0.965	0.965	—	0.951	0.970	0.957	0.950	0.977
	MS-Random	0.981	0.969	0.954	0.971	—	0.958	0.969	0.957	0.958	0.962
	ATC-NE	0.948	0.864	0.902	0.949	—	0.746	0.908	0.813	0.817	0.917
	ATC-MC	0.949	0.893	0.928	0.948	—	0.802	0.929	0.861	0.859	0.926
`pets`	MS-SSMBA	0.943	0.974	0.981	0.945	0.942	—	0.979	0.982	0.978	0.969
	MS-EDA	0.965	0.980	0.985	0.958	0.952	—	0.983	0.975	0.976	0.983
	MS-BT	0.936	0.971	0.974	0.931	0.926	—	0.980	0.963	0.979	0.977
	MS-Random	0.941	0.976	0.971	0.925	0.931	—	0.976	0.968	0.976	0.979
	ATC-NE	0.594	0.967	0.900	0.577	0.862	—	0.947	0.959	0.929	0.882
	ATC-MC	0.611	0.968	0.909	0.589	0.851	—	0.955	0.959	0.939	0.894
`sports`	MS-SSMBA	0.979	0.982	0.994	0.980	0.979	0.970	—	0.982	0.984	0.978
	MS-EDA	0.979	0.987	0.986	0.960	0.984	0.985	—	0.985	0.986	0.987
	MS-BT	0.966	0.977	0.989	0.949	0.969	0.979	—	0.986	0.984	0.988
	MS-Random	0.981	0.979	0.994	0.978	0.980	0.978	—	0.985	0.985	0.984
	ATC-NE	0.655	0.945	0.963	0.688	0.850	0.890	—	0.938	0.969	0.957
	ATC-MC	0.705	0.951	0.965	0.722	0.882	0.905	—	0.944	0.972	0.953
`tech`	MS-SSMBA	0.944	0.979	0.977	0.947	0.967	0.958	0.975	—	0.990	0.981
	MS-EDA	0.917	0.969	0.971	0.925	0.953	0.937	0.967	—	0.963	0.972
	MS-BT	0.894	0.982	0.987	0.924	0.937	0.933	0.978	—	0.987	0.972
	MS-Random	0.929	0.959	0.977	0.918	0.952	0.944	0.963	—	0.984	0.971
	ATC-NE	0.937	0.939	0.983	0.933	0.946	0.961	0.961	—	0.974	0.981
	ATC-MC	0.941	0.941	0.980	0.928	0.955	0.958	0.963	—	0.971	0.982
`tools`	MS-SSMBA	0.977	0.988	0.975	0.967	0.971	0.986	0.985	0.978	—	0.983
	MS-EDA	0.975	0.985	0.978	0.969	0.980	0.986	0.988	0.975	—	0.982
	MS-BT	0.940	0.976	0.970	0.941	0.944	0.969	0.978	0.948	—	0.975
	MS-Random	0.963	0.985	0.978	0.954	0.963	0.988	0.984	0.977	—	0.987
	ATC-NE	0.857	0.945	0.964	0.883	0.865	0.925	0.964	0.963	—	0.922
	ATC-MC	0.868	0.941	0.965	0.887	0.876	0.936	0.967	0.971	—	0.919
`toys`	MS-SSMBA	0.955	0.971	0.985	0.980	0.961	0.966	0.988	0.978	0.971	—
	MS-EDA	0.965	0.968	0.984	0.970	0.985	0.973	0.981	0.975	0.983	—
	MS-BT	0.890	0.946	0.959	0.944	0.925	0.935	0.965	0.943	0.934	—
	MS-Random	0.959	0.970	0.979	0.960	0.965	0.967	0.977	0.978	0.974	—
	ATC-NE	0.883	0.878	0.885	0.763	0.906	0.798	0.938	0.815	0.899	—
	ATC-MC	0.889	0.897	0.895	0.754	0.920	0.818	0.947	0.833	0.914	—

Table 10: Full R^2 metrics for all pairs of training and test domains for BERT models trained on AWS.

		Test Domain									
Train Domain	Measure	`slate`	`verbatim`	`facetoface`	`oup`	`nineeleven`	`fiction`	`telephone`	`travel`	`letters`	`government`
`slate`	MS-SSMBA	—	0.457	0.760	0.544	0.642	0.706	0.727	0.585	0.640	0.714
	MS-EDA	—	0.574	0.723	0.565	0.620	0.708	0.731	0.549	0.595	0.706
	MS-BT	—	0.862	0.920	0.942	0.930	0.913	0.878	0.936	0.949	0.940
	MS-Random	—	0.242	0.336	0.501	0.415	0.121	0.170	0.301	0.652	0.470
	ATC-NE	—	0.681	0.677	0.594	0.599	0.741	0.669	0.664	0.661	0.773
	ATC-MC	—	0.682	0.675	0.590	0.600	0.737	0.670	0.665	0.661	0.771
`fiction`	MS-SSMBA	0.581	0.492	0.765	0.451	0.530	—	0.683	0.499	0.502	0.614
	MS-EDA	0.657	0.601	0.728	0.620	0.530	—	0.676	0.527	0.508	0.642
	MS-BT	0.889	0.887	0.917	0.950	0.921	—	0.863	0.928	0.935	0.941
	MS-Random	0.360	0.414	0.459	0.666	0.531	—	0.188	0.464	0.680	0.631
	ATC-NE	0.530	0.425	0.718	0.323	0.536	—	0.556	0.425	0.597	0.522
	ATC-MC	0.525	0.418	0.719	0.326	0.534	—	0.555	0.422	0.592	0.521
`telephone`	MS-SSMBA	0.539	0.453	0.826	0.429	0.480	0.647	—	0.436	0.559	0.439
	MS-EDA	0.677	0.636	0.804	0.492	0.639	0.819	—	0.557	0.653	0.568
	MS-BT	0.882	0.901	0.928	0.936	0.927	0.912	—	0.937	0.916	0.919
	MS-Random	0.332	0.498	0.488	0.584	0.538	0.310	—	0.586	0.677	0.576
	ATC-NE	0.557	0.518	0.762	0.527	0.467	0.695	—	0.368	0.554	0.468
	ATC-MC	0.561	0.518	0.762	0.524	0.467	0.698	—	0.372	0.560	0.470
`travel`	MS-SSMBA	0.503	0.506	0.588	0.419	0.501	0.543	0.597	—	0.670	0.507
	MS-EDA	0.444	0.429	0.502	0.388	0.342	0.532	0.478	—	0.438	0.377
	MS-BT	0.806	0.799	0.861	0.916	0.889	0.828	0.802	—	0.923	0.907
	MS-Random	0.315	0.333	0.385	0.643	0.516	0.182	0.224	—	0.690	0.626
	ATC-NE	0.561	0.516	0.498	0.686	0.546	0.581	0.606	—	0.585	0.598
	ATC-MC	0.563	0.517	0.503	0.690	0.544	0.588	0.611	—	0.582	0.607
`government`	MS-SSMBA	0.592	0.497	0.618	0.572	0.596	0.648	0.593	0.539	0.676	—
	MS-EDA	0.577	0.574	0.550	0.476	0.618	0.600	0.491	0.518	0.526	—
	MS-BT	0.818	0.812	0.859	0.909	0.893	0.813	0.809	0.909	0.917	—
	MS-Random	0.010	0.083	0.207	0.306	0.279	0.011	0.027	0.123	0.360	—
	ATC-NE	0.663	0.405	0.564	0.722	0.337	0.634	0.630	0.509	0.653	—
	ATC-MC	0.666	0.405	0.574	0.721	0.341	0.633	0.636	0.509	0.654	—

Table 11: Full R^2 metrics for all pairs of training and test domains for CNN models trained on MNLI.

		Test Domain									
Train Domain	Measure	`slate`	`verbatim`	`facetoface`	`oup`	`nineeleven`	`fiction`	`telephone`	`travel`	`letters`	`government`
`slate`	MS-SSMBA	—	0.920	0.905	0.943	0.916	0.929	0.930	0.957	0.931	0.970
	MS-EDA	—	0.701	0.754	0.816	0.784	0.790	0.708	0.787	0.820	0.837
	MS-BT	—	0.862	0.920	0.942	0.930	0.913	0.878	0.936	0.949	0.940
	MS-Random	—	0.242	0.336	0.501	0.415	0.121	0.170	0.301	0.652	0.470
	ATC-NE	—	0.878	0.899	0.954	0.911	0.925	0.911	0.904	0.890	0.951
	ATC-MC	—	0.871	0.884	0.953	0.908	0.932	0.914	0.911	0.887	0.949
`fiction`	MS-SSMBA	0.954	0.928	0.895	0.952	0.946	—	0.952	0.975	0.946	0.954
	MS-EDA	0.661	0.687	0.699	0.840	0.686	—	0.683	0.764	0.799	0.797
	MS-BT	0.889	0.887	0.917	0.950	0.921	—	0.863	0.928	0.935	0.941
	MS-Random	0.360	0.414	0.459	0.666	0.531	—	0.188	0.464	0.680	0.631
	ATC-NE	0.808	0.585	0.881	0.726	0.939	—	0.836	0.826	0.850	0.890
	ATC-MC	0.817	0.652	0.889	0.732	0.936	—	0.841	0.830	0.861	0.892
`telephone`	MS-SSMBA	0.943	0.962	0.948	0.953	0.917	0.954	—	0.921	0.934	0.956
	MS-EDA	0.776	0.787	0.790	0.852	0.823	0.847	—	0.853	0.877	0.852
	MS-BT	0.882	0.901	0.928	0.936	0.927	0.912	—	0.937	0.916	0.919
	MS-Random	0.332	0.498	0.488	0.584	0.538	0.310	—	0.586	0.677	0.576
	ATC-NE	0.662	0.576	0.884	0.904	0.762	0.665	—	0.771	0.856	0.889
	ATC-MC	0.714	0.668	0.902	0.924	0.807	0.732	—	0.795	0.857	0.899
`travel`	MS-SSMBA	0.938	0.935	0.949	0.924	0.948	0.946	0.958	—	0.962	0.953
	MS-EDA	0.538	0.553	0.471	0.711	0.602	0.638	0.533	—	0.755	0.699
	MS-BT	0.806	0.799	0.861	0.916	0.889	0.828	0.802	—	0.923	0.907
	MS-Random	0.315	0.333	0.385	0.643	0.516	0.182	0.224	—	0.690	0.626
	ATC-NE	0.319	0.417	0.500	0.889	0.768	0.396	0.494	—	0.928	0.911
	ATC-MC	0.412	0.506	0.591	0.906	0.789	0.525	0.572	—	0.921	0.917
`government`	MS-SSMBA	0.916	0.960	0.781	0.919	0.816	0.882	0.914	0.950	0.920	—
	MS-EDA	0.484	0.572	0.544	0.728	0.715	0.655	0.566	0.644	0.681	—
	MS-BT	0.818	0.812	0.859	0.909	0.893	0.813	0.809	0.909	0.917	—
	MS-Random	0.010	0.083	0.207	0.306	0.279	0.011	0.027	0.123	0.360	—
	ATC-NE	0.544	0.385	0.439	0.914	0.731	0.157	0.449	0.538	0.842	—
	ATC-MC	0.624	0.481	0.478	0.919	0.797	0.243	0.505	0.576	0.856	—

Table 12: Full R^2 metrics for all pairs of training and test domains for BERT models trained on MNLI.

			Test Domain		
Model	Train Domain	Measure	SVHN	Colored MNIST	MNIST
NiN	SVHN	MS-RandAug	—	0.668	0.616
		MS-Erase	—	-0.102	-0.168
		MS-FC	—	-0.233	-0.398
		ATC-NE	—	0.577	0.695
		ATC-MC	—	0.646	0.716

Table 13: Full macro τ metrics for all test domains for image classification models on SVHN, Colored MNIST, and MNIST.

			Test Domain			
Model	Train Domain	Measure	CIFAR10	CINIC10	CIFAR10.1	CIFAR10.2
NiN	CIFAR10	MS-RandAug	—	0.785	0.830	0.783
		MS-Erase	—	0.385	0.485	0.499
		MS-FC	—	0.194	0.077	-0.037
		ATC-NE	—	0.250	0.543	0.484
		ATC-MC	—	0.319	0.514	0.446
ResNet	CIFAR10	MS-RandAug	—	0.706	0.722	0.742
		MS-Erase	—	0.723	0.727	0.699
		MS-FC	—	0.647	0.585	0.607
		ATC-NE	—	0.644	0.687	0.630
		ATC-MC	—	0.631	0.672	0.624
VGG	CIFAR10	MS-RandAug	—	0.868	0.831	0.772
		MS-Erase	—	-0.153	-0.196	-0.215
		MS-FC	—	0.597	0.569	0.512
		ATC-NE	—	0.531	0.656	0.599
		ATC-MC	—	0.533	0.648	0.586
CNN	CINIC10	MS-RandAug	0.756	—	0.698	0.695
		MS-Erase	0.619	—	0.539	0.577
		MS-FC	0.249	—	0.286	0.252
		ATC-NE	0.607	—	0.510	0.447
		ATC-MC	0.611	—	0.500	0.440

Table 14: Full macro τ metrics for all test domains for image classification models on CIFAR10, CINIC10, CIFAR10.1, and CIFAR10.2.

		Test Domain									
Train Domain	Measure	`books`	`clothing`	`home`	`kindle`	`movies`	`pets`	`sports`	`tech`	`tools`	`toys`
`books`	MS-SSMBA	—	0.708	0.741	0.687	0.678	0.754	0.738	0.738	0.721	0.669
	MS-EDA	—	0.722	0.738	0.640	0.685	0.756	0.688	0.709	0.719	0.725
	MS-BT	—	0.615	0.631	0.549	0.596	0.656	0.602	0.644	0.584	0.621
	MS-Random	—	0.591	0.586	0.499	0.587	0.587	0.530	0.530	0.546	0.589
	ATC-NE	—	0.605	0.642	0.534	0.496	0.629	0.605	0.538	0.618	0.673
	ATC-MC	—	0.604	0.639	0.534	0.489	0.630	0.606	0.544	0.616	0.670
`clothing`	MS-SSMBA	0.769	—	0.774	0.773	0.793	0.726	0.653	0.716	0.692	0.709
	MS-EDA	0.681	—	0.670	0.787	0.721	0.696	0.675	0.694	0.690	0.784
	MS-BT	0.574	—	0.542	0.654	0.595	0.481	0.506	0.501	0.503	0.679
	MS-Random	0.623	—	0.591	0.603	0.612	0.557	0.496	0.507	0.549	0.640
	ATC-NE	0.420	—	0.366	0.408	0.450	0.308	0.501	0.242	0.560	0.563
	ATC-MC	0.421	—	0.363	0.409	0.451	0.315	0.498	0.239	0.558	0.563
`home`	MS-SSMBA	0.786	0.552	—	0.731	0.731	0.769	0.790	0.789	0.775	0.614
	MS-EDA	0.734	0.659	—	0.767	0.649	0.757	0.707	0.723	0.696	0.748
	MS-BT	0.670	0.498	—	0.641	0.639	0.634	0.706	0.688	0.662	0.630
	MS-Random	0.700	0.549	—	0.665	0.705	0.686	0.663	0.664	0.632	0.651
	ATC-NE	0.450	0.480	—	0.459	0.447	0.455	0.559	0.497	0.558	0.445
	ATC-MC	0.459	0.480	—	0.460	0.452	0.458	0.557	0.495	0.557	0.448
`kindle`	MS-SSMBA	0.546	0.679	0.648	—	0.557	0.699	0.656	0.608	0.607	0.630
	MS-EDA	0.663	0.644	0.717	—	0.637	0.743	0.699	0.676	0.643	0.696
	MS-BT	0.556	0.648	0.652	—	0.562	0.686	0.683	0.668	0.643	0.674
	MS-Random	0.454	0.519	0.406	—	0.428	0.555	0.434	0.415	0.415	0.535
	ATC-NE	0.370	0.517	0.468	—	0.338	0.583	0.540	0.383	0.472	0.545
	ATC-MC	0.388	0.512	0.469	—	0.342	0.578	0.538	0.384	0.467	0.541
`movies`	MS-SSMBA	0.572	0.689	0.717	0.660	—	0.732	0.741	0.675	0.714	0.749
	MS-EDA	0.500	0.717	0.711	0.629	—	0.733	0.725	0.660	0.719	0.748
	MS-BT	0.619	0.681	0.687	0.595	—	0.738	0.729	0.717	0.717	0.717
	MS-Random	0.435	0.491	0.523	0.554	—	0.460	0.528	0.436	0.517	0.570
	ATC-NE	0.436	0.605	0.623	0.398	—	0.571	0.622	0.502	0.659	0.701
	ATC-MC	0.449	0.612	0.631	0.403	—	0.586	0.628	0.520	0.661	0.704
`pets`	MS-SSMBA	0.747	0.560	0.653	0.766	0.769	—	0.719	0.696	0.697	0.731
	MS-EDA	0.762	0.684	0.682	0.794	0.773	—	0.709	0.666	0.689	0.745
	MS-BT	0.741	0.550	0.625	0.751	0.776	—	0.704	0.676	0.702	0.654
	MS-Random	0.619	0.575	0.572	0.696	0.662	—	0.580	0.532	0.590	0.710
	ATC-NE	0.570	0.545	0.462	0.533	0.543	—	0.511	0.333	0.503	0.634
	ATC-MC	0.575	0.541	0.461	0.535	0.546	—	0.510	0.332	0.502	0.632
`sports`	MS-SSMBA	0.774	0.561	0.657	0.812	0.764	0.683	—	0.689	0.746	0.651
	MS-EDA	0.770	0.633	0.659	0.791	0.710	0.712	—	0.701	0.732	0.692
	MS-BT	0.670	0.484	0.598	0.669	0.672	0.545	—	0.534	0.704	0.538
	MS-Random	0.684	0.472	0.562	0.704	0.638	0.589	—	0.614	0.622	0.693
	ATC-NE	0.502	0.456	0.454	0.506	0.443	0.374	—	0.397	0.565	0.577
	ATC-MC	0.513	0.459	0.459	0.521	0.453	0.393	—	0.410	0.574	0.574
`tech`	MS-SSMBA	0.784	0.674	0.768	0.793	0.779	0.755	0.785	—	0.690	0.746
	MS-EDA	0.718	0.698	0.760	0.743	0.752	0.817	0.772	—	0.726	0.781
	MS-BT	0.715	0.576	0.719	0.732	0.746	0.716	0.726	—	0.687	0.683
	MS-Random	0.680	0.647	0.652	0.686	0.668	0.677	0.664	—	0.553	0.742
	ATC-NE	0.547	0.688	0.681	0.560	0.640	0.653	0.666	—	0.637	0.723
	ATC-MC	0.553	0.690	0.678	0.567	0.639	0.650	0.664	—	0.633	0.721
`tools`	MS-SSMBA	0.783	0.541	0.604	0.777	0.782	0.757	0.758	0.730	—	0.682
	MS-EDA	0.716	0.544	0.628	0.769	0.735	0.764	0.722	0.696	—	0.745
	MS-BT	0.691	0.342	0.454	0.681	0.677	0.567	0.575	0.636	—	0.513
	MS-Random	0.661	0.550	0.581	0.675	0.685	0.668	0.623	0.584	—	0.687
	ATC-NE	0.621	0.472	0.480	0.647	0.648	0.536	0.598	0.448	—	0.619
	ATC-MC	0.630	0.465	0.491	0.649	0.655	0.538	0.597	0.456	—	0.622
`toys`	MS-SSMBA	0.760	0.702	0.758	0.763	0.731	0.775	0.782	0.750	0.753	—
	MS-EDA	0.689	0.678	0.661	0.757	0.646	0.736	0.711	0.661	0.700	—
	MS-BT	0.582	0.604	0.683	0.626	0.610	0.695	0.723	0.660	0.658	—
	MS-Random	0.629	0.660	0.575	0.661	0.609	0.605	0.599	0.543	0.582	—
	ATC-NE	0.401	0.617	0.309	0.479	0.370	0.323	0.485	0.143	0.526	—
	ATC-MC	0.408	0.614	0.315	0.476	0.368	0.329	0.488	0.146	0.535	—

Table 15: Full macro τ metrics for all pairs of training and test domains for CNN models trained on AWS.

		Test Domain									
Train Domain	Measure	`books`	`clothing`	`home`	`kindle`	`movies`	`pets`	`sports`	`tech`	`tools`	`toys`
`books`	MS-SSMBA	—	0.886	0.906	0.893	0.901	0.900	0.925	0.918	0.931	0.857
	MS-EDA	—	0.883	0.851	0.890	0.852	0.863	0.854	0.855	0.895	0.844
	MS-BT	—	0.830	0.887	0.893	0.883	0.884	0.871	0.878	0.869	0.824
	MS-Random	—	0.882	0.871	0.908	0.906	0.897	0.894	0.900	0.913	0.846
	ATC-NE	—	0.803	0.814	0.851	0.821	0.885	0.756	0.843	0.774	0.832
	ATC-MC	—	0.789	0.806	0.850	0.794	0.874	0.760	0.845	0.768	0.824
`clothing`	MS-SSMBA	0.797	—	0.882	0.714	0.773	0.826	0.881	0.826	0.883	0.824
	MS-EDA	0.828	—	0.848	0.813	0.787	0.862	0.741	0.839	0.848	0.777
	MS-BT	0.817	—	0.903	0.686	0.837	0.832	0.849	0.835	0.823	0.778
	MS-Random	0.762	—	0.858	0.691	0.826	0.832	0.872	0.822	0.868	0.786
	ATC-NE	0.729	—	0.805	0.672	0.681	0.817	0.728	0.698	0.792	0.791
	ATC-MC	0.735	—	0.794	0.683	0.701	0.858	0.744	0.736	0.808	0.791
`home`	MS-SSMBA	0.736	0.800	—	0.786	0.780	0.868	0.896	0.876	0.888	0.885
	MS-EDA	0.680	0.885	—	0.768	0.830	0.880	0.879	0.843	0.877	0.930
	MS-BT	0.749	0.788	—	0.756	0.778	0.819	0.871	0.835	0.906	0.866
	MS-Random	0.729	0.811	—	0.701	0.759	0.853	0.931	0.831	0.876	0.874
	ATC-NE	0.691	0.788	—	0.750	0.802	0.791	0.797	0.739	0.824	0.824
	ATC-MC	0.684	0.782	—	0.738	0.793	0.806	0.772	0.751	0.826	0.819
`kindle`	MS-SSMBA	0.735	0.850	0.885	—	0.701	0.913	0.875	0.869	0.763	0.871
	MS-EDA	0.840	0.830	0.824	—	0.709	0.879	0.856	0.865	0.854	0.790
	MS-BT	0.727	0.848	0.892	—	0.763	0.856	0.926	0.871	0.879	0.861
	MS-Random	0.773	0.848	0.816	—	0.735	0.875	0.857	0.843	0.794	0.847
	ATC-NE	0.695	0.335	0.486	—	0.644	0.211	0.274	0.426	0.238	0.589
	ATC-MC	0.729	0.455	0.558	—	0.671	0.337	0.366	0.509	0.369	0.652
`movies`	MS-SSMBA	0.845	0.897	0.888	0.782	—	0.850	0.931	0.877	0.881	0.896
	MS-EDA	0.764	0.881	0.852	0.863	—	0.859	0.901	0.863	0.882	0.938
	MS-BT	0.810	0.894	0.882	0.719	—	0.837	0.894	0.864	0.872	0.870
	MS-Random	0.851	0.875	0.855	0.742	—	0.850	0.905	0.864	0.861	0.874
	ATC-NE	0.692	0.745	0.807	0.712	—	0.561	0.768	0.580	0.620	0.763
	ATC-MC	0.698	0.770	0.847	0.722	—	0.622	0.813	0.637	0.678	0.784
`pets`	MS-SSMBA	0.728	0.737	0.866	0.737	0.829	—	0.883	0.867	0.859	0.773
	MS-EDA	0.732	0.831	0.880	0.601	0.788	—	0.853	0.834	0.903	0.841
	MS-BT	0.697	0.685	0.816	0.758	0.816	—	0.867	0.814	0.865	0.810
	MS-Random	0.753	0.731	0.813	0.735	0.841	—	0.828	0.871	0.869	0.774
	ATC-NE	0.623	0.740	0.719	0.553	0.737	—	0.815	0.788	0.748	0.781
	ATC-MC	0.662	0.753	0.723	0.566	0.711	—	0.831	0.781	0.752	0.750
`sports`	MS-SSMBA	0.715	0.743	0.879	0.764	0.837	0.863	—	0.879	0.860	0.870
	MS-EDA	0.757	0.731	0.807	0.673	0.798	0.786	—	0.815	0.827	0.833
	MS-BT	0.728	0.753	0.766	0.664	0.826	0.832	—	0.783	0.824	0.865
	MS-Random	0.735	0.702	0.816	0.728	0.808	0.802	—	0.830	0.849	0.836
	ATC-NE	0.489	0.734	0.738	0.567	0.720	0.726	—	0.726	0.725	0.862
	ATC-MC	0.484	0.728	0.766	0.562	0.754	0.727	—	0.725	0.741	0.860
`tech`	MS-SSMBA	0.784	0.763	0.845	0.710	0.769	0.853	0.755	—	0.886	0.877
	MS-EDA	0.771	0.867	0.881	0.763	0.801	0.855	0.886	—	0.904	0.869
	MS-BT	0.753	0.750	0.867	0.719	0.805	0.869	0.787	—	0.842	0.835
	MS-Random	0.819	0.721	0.843	0.711	0.854	0.889	0.787	—	0.854	0.874
	ATC-NE	0.737	0.744	0.787	0.751	0.732	0.795	0.727	—	0.830	0.863
	ATC-MC	0.771	0.730	0.764	0.749	0.704	0.809	0.771	—	0.789	0.843
`tools`	MS-SSMBA	0.792	0.854	0.786	0.707	0.746	0.875	0.888	0.792	—	0.866
	MS-EDA	0.859	0.846	0.834	0.847	0.790	0.837	0.886	0.819	—	0.894
	MS-BT	0.769	0.802	0.763	0.709	0.750	0.832	0.824	0.744	—	0.856
	MS-Random	0.792	0.825	0.816	0.717	0.724	0.885	0.877	0.801	—	0.845
	ATC-NE	0.745	0.789	0.775	0.744	0.632	0.698	0.856	0.737	—	0.799
	ATC-MC	0.749	0.798	0.772	0.745	0.634	0.737	0.876	0.777	—	0.793
`toys`	MS-SSMBA	0.779	0.830	0.813	0.766	0.695	0.808	0.872	0.872	0.891	—
	MS-EDA	0.696	0.790	0.805	0.821	0.790	0.837	0.834	0.860	0.840	—
	MS-BT	0.748	0.762	0.746	0.755	0.707	0.840	0.809	0.773	0.811	—
	MS-Random	0.813	0.788	0.783	0.770	0.660	0.788	0.838	0.873	0.891	—
	ATC-NE	0.656	0.469	0.743	0.550	0.628	0.515	0.792	0.703	0.771	—
	ATC-MC	0.694	0.482	0.726	0.539	0.651	0.571	0.805	0.730	0.777	—

Table 16: Full macro τ metrics for all pairs of training and test domains for BERT models trained on AWS.

		Test Domain									
Train Domain	Measure	`slate`	`verbatim`	`facetoface`	`oup`	`nineeleven`	`fiction`	`telephone`	`travel`	`letters`	`government`
`slate`	MS-SSMBA	—	0.483	0.675	0.553	0.608	0.650	0.661	0.570	0.622	0.641
	MS-EDA	—	0.564	0.675	0.563	0.599	0.658	0.672	0.557	0.599	0.649
	MS-BT	—	0.669	0.742	0.736	0.756	0.727	0.754	0.751	0.738	0.791
	MS-Random	—	0.393	0.526	0.529	0.480	0.398	0.438	0.445	0.640	0.531
	ATC-NE	—	0.650	0.632	0.577	0.594	0.683	0.625	0.602	0.621	0.699
	ATC-MC	—	0.652	0.632	0.578	0.594	0.681	0.625	0.603	0.622	0.697
`fiction`	MS-SSMBA	0.577	0.537	0.692	0.482	0.539	—	0.629	0.520	0.546	0.598
	MS-EDA	0.632	0.584	0.691	0.608	0.557	—	0.631	0.529	0.540	0.608
	MS-BT	0.783	0.704	0.771	0.693	0.671	—	0.776	0.680	0.707	0.693
	MS-Random	0.495	0.446	0.522	0.531	0.437	—	0.434	0.464	0.531	0.493
	ATC-NE	0.538	0.461	0.630	0.428	0.513	—	0.542	0.469	0.581	0.536
	ATC-MC	0.530	0.450	0.631	0.427	0.513	—	0.543	0.463	0.576	0.534
`telephone`	MS-SSMBA	0.567	0.503	0.736	0.493	0.507	0.608	—	0.492	0.572	0.496
	MS-EDA	0.637	0.633	0.726	0.537	0.615	0.735	—	0.579	0.642	0.580
	MS-BT	0.657	0.668	0.722	0.637	0.663	0.639	—	0.621	0.602	0.614
	MS-Random	0.504	0.470	0.573	0.470	0.510	0.480	—	0.482	0.576	0.440
	ATC-NE	0.533	0.507	0.681	0.526	0.500	0.632	—	0.453	0.564	0.491
	ATC-MC	0.534	0.507	0.681	0.522	0.501	0.637	—	0.455	0.564	0.494
`travel`	MS-SSMBA	0.524	0.528	0.585	0.462	0.517	0.537	0.569	—	0.617	0.521
	MS-EDA	0.492	0.484	0.523	0.461	0.418	0.550	0.524	—	0.507	0.458
	MS-BT	0.619	0.634	0.630	0.660	0.681	0.655	0.645	—	0.713	0.699
	MS-Random	0.465	0.476	0.467	0.514	0.492	0.472	0.526	—	0.569	0.497
	ATC-NE	0.579	0.528	0.533	0.643	0.541	0.571	0.591	—	0.572	0.597
	ATC-MC	0.576	0.528	0.534	0.642	0.536	0.573	0.594	—	0.570	0.602
`government`	MS-SSMBA	0.578	0.531	0.600	0.568	0.565	0.617	0.589	0.563	0.616	—
	MS-EDA	0.581	0.582	0.565	0.535	0.594	0.624	0.519	0.552	0.566	—
	MS-BT	0.719	0.693	0.686	0.687	0.755	0.701	0.651	0.712	0.719	—
	MS-Random	0.274	0.268	0.357	0.444	0.464	0.301	0.247	0.395	0.452	—
	ATC-NE	0.624	0.464	0.581	0.666	0.440	0.601	0.616	0.524	0.641	—
	ATC-MC	0.621	0.465	0.583	0.667	0.446	0.602	0.616	0.522	0.642	—

Table 17: Full macro τ metrics for all pairs of training and test domains for CNN models trained on MNLI.

		Test Domain									
Train Domain	Measure	`slate`	`verbatim`	`facetoface`	`oup`	`nineeleven`	`fiction`	`telephone`	`travel`	`letters`	`government`
`slate`	MS-SSMBA	—	0.743	0.746	0.709	0.767	0.742	0.727	0.771	0.782	0.819
	MS-EDA	—	0.580	0.695	0.674	0.696	0.703	0.655	0.651	0.671	0.737
	MS-BT	—	0.669	0.742	0.736	0.756	0.727	0.754	0.751	0.738	0.791
	MS-Random	—	0.393	0.526	0.529	0.480	0.398	0.438	0.445	0.640	0.531
	ATC-NE	—	0.689	0.768	0.786	0.728	0.752	0.761	0.744	0.743	0.766
	ATC-MC	—	0.684	0.761	0.794	0.730	0.761	0.789	0.765	0.744	0.785
`fiction`	MS-SSMBA	0.755	0.673	0.699	0.726	0.729	—	0.769	0.806	0.770	0.727
	MS-EDA	0.609	0.505	0.649	0.659	0.532	—	0.608	0.554	0.624	0.629
	MS-BT	0.783	0.704	0.771	0.693	0.671	—	0.776	0.680	0.707	0.693
	MS-Random	0.495	0.446	0.522	0.531	0.437	—	0.434	0.464	0.531	0.493
	ATC-NE	0.618	0.377	0.707	0.478	0.711	—	0.596	0.558	0.658	0.711
	ATC-MC	0.631	0.422	0.702	0.513	0.707	—	0.603	0.549	0.670	0.706
`telephone`	MS-SSMBA	0.720	0.768	0.793	0.792	0.668	0.720	—	0.813	0.758	0.788
	MS-EDA	0.670	0.667	0.706	0.617	0.718	0.795	—	0.642	0.732	0.661
	MS-BT	0.657	0.668	0.722	0.637	0.663	0.639	—	0.621	0.602	0.614
	MS-Random	0.504	0.470	0.573	0.470	0.510	0.480	—	0.482	0.576	0.440
	ATC-NE	0.462	0.486	0.692	0.575	0.641	0.542	—	0.644	0.697	0.702
	ATC-MC	0.519	0.551	0.717	0.641	0.637	0.597	—	0.665	0.684	0.708
`travel`	MS-SSMBA	0.700	0.806	0.737	0.676	0.744	0.707	0.848	—	0.789	0.783
	MS-EDA	0.483	0.453	0.486	0.493	0.489	0.515	0.483	—	0.522	0.495
	MS-BT	0.619	0.634	0.630	0.660	0.681	0.655	0.645	—	0.713	0.699
	MS-Random	0.465	0.476	0.467	0.514	0.492	0.472	0.526	—	0.569	0.497
	ATC-NE	0.155	0.320	0.234	0.559	0.533	0.146	0.360	—	0.633	0.630
	ATC-MC	0.175	0.370	0.320	0.613	0.576	0.219	0.443	—	0.648	0.646
`government`	MS-SSMBA	0.769	0.777	0.727	0.652	0.640	0.783	0.784	0.805	0.754	—
	MS-EDA	0.521	0.513	0.480	0.578	0.661	0.657	0.494	0.565	0.598	—
	MS-BT	0.719	0.693	0.686	0.687	0.755	0.701	0.651	0.712	0.719	—
	MS-Random	0.274	0.268	0.357	0.444	0.464	0.301	0.247	0.395	0.452	—
	ATC-NE	0.377	0.176	0.472	0.660	0.558	0.239	0.497	0.302	0.638	—
	ATC-MC	0.427	0.231	0.487	0.653	0.600	0.275	0.496	0.323	0.629	—

Table 18: Full macro τ metrics for all pairs of training and test domains for BERT models trained on MNLI.

		Test Domain									
Model	Measure	`books`	`clothing`	`home`	`kindle`	`movies`	`pets`	`sports`	`tech`	`tools`	`toys`
CNN	MS-SSMBA	0.677	0.688	0.758	0.641	0.691	0.786	0.784	0.758	0.722	0.683
	MS-EDA	0.637	0.672	0.704	0.612	0.658	0.746	0.739	0.680	0.703	0.711
	MS-BT	0.601	0.532	0.572	0.509	0.521	0.509	0.587	0.536	0.571	0.529
	MS-Random	0.485	0.542	0.501	0.522	0.440	0.491	0.650	0.477	0.507	0.550
	ATC-NE	0.406	0.727	0.728	0.362	0.499	0.722	0.726	0.679	0.730	0.752
	ATC-MC	0.410	0.726	0.729	0.359	0.500	0.724	0.727	0.681	0.730	0.752
BERT	MS-SSMBA	0.790	0.850	0.868	0.664	0.899	0.864	0.855	0.874	0.880	0.845
	MS-EDA	0.739	0.848	0.825	0.731	0.793	0.836	0.825	0.828	0.837	0.841
	MS-BT	0.702	0.845	0.830	0.668	0.792	0.839	0.843	0.813	0.819	0.858
	MS-Random	0.839	0.841	0.891	0.694	0.864	0.834	0.747	0.825	0.850	0.831
	ATC-NE	0.613	0.721	0.712	0.618	0.695	0.666	0.733	0.693	0.708	0.755
	ATC-MC	0.621	0.735	0.730	0.616	0.711	0.691	0.749	0.710	0.726	0.764

Table 19: Full micro τ metrics for all test domains for CNN and BERT models trained on AWS.

		Test Domain									
Model	Measure	`slate`	`verbatim`	`facetoface`	`oup`	`nineeleven`	`fiction`	`telephone`	`travel`	`letters`	`government`
CNN	MS-SSMBA	0.540	0.493	0.562	0.493	0.527	0.431	0.607	0.544	0.579	0.563
	MS-EDA	0.560	0.555	0.427	0.520	0.488	0.435	0.499	0.552	0.532	0.542
	MS-BT	0.698	0.684	0.611	0.713	0.713	0.595	0.676	0.714	0.697	0.722
	MS-Random	0.448	0.417	0.427	0.503	0.472	0.324	0.395	0.450	0.550	0.509
	ATC-NE	0.461	0.413	0.549	0.427	0.350	0.517	0.531	0.307	0.497	0.404
	ATC-MC	0.460	0.411	0.550	0.429	0.351	0.517	0.532	0.307	0.496	0.404
BERT	MS-SSMBA	0.782	0.710	0.742	0.652	0.724	0.756	0.779	0.691	0.768	0.700
	MS-EDA	0.563	0.556	0.481	0.622	0.586	0.545	0.529	0.607	0.607	0.630
	MS-BT	0.698	0.684	0.611	0.713	0.713	0.595	0.676	0.714	0.697	0.722
	MS-Random	0.448	0.417	0.427	0.503	0.472	0.324	0.395	0.450	0.550	0.509
	ATC-NE	0.375	0.389	0.660	0.606	0.593	0.414	0.523	0.511	0.655	0.687
	ATC-MC	0.405	0.434	0.676	0.632	0.614	0.466	0.554	0.530	0.663	0.693

Table 20: Full micro τ metrics for all test domains for CNN and BERT models trained on MNLI.

	Train Domain (Model)				
Measure	SVHN (NiN)	CIFAR10 (NiN)	CIFAR10 (ResNet)	CIFAR10 (VGG)	CINIC10 (CNN)
MS-RandAug	0.733	0.797	0.746	0.869	0.759
MS-Erase	0.324	0.409	0.708	-0.183	0.606
MS-FC	-0.033	-0.015	0.568	0.628	0.289
ATC-NE	0.859	0.648	0.757	0.773	0.548
ATC-MC	0.843	0.605	0.756	0.767	0.553

Table 21: Full in-domain τ metrics for all test domains for image classification models and datasets.

		Train Domain									
Model	Measure	`books`	`clothing`	`home`	`kindle`	`movies`	`pets`	`sports`	`tech`	`tools`	`toys`
CNN	MS-SSMBA	0.646	0.613	0.643	0.577	0.597	0.591	0.703	0.648	0.643	0.620
	MS-EDA	0.653	0.558	0.615	0.660	0.447	0.666	0.708	0.646	0.663	0.565
	MS-BT	0.569	0.449	0.582	0.537	0.626	0.605	0.568	0.653	0.640	0.652
	MS-Random	0.560	0.557	0.559	0.559	0.593	0.596	0.572	0.545	0.566	0.615
	ATC-NE	0.575	0.413	0.561	0.478	0.294	0.464	0.539	0.561	0.446	0.341
	ATC-MC	0.566	0.405	0.558	0.478	0.300	0.461	0.540	0.569	0.451	0.338
BERT	MS-SSMBA	0.735	0.874	0.868	0.693	0.780	0.871	0.884	0.851	0.851	0.884
	MS-EDA	0.875	0.788	0.814	0.700	0.870	0.894	0.818	0.858	0.882	0.810
	MS-BT	0.801	0.811	0.857	0.638	0.835	0.813	0.860	0.846	0.803	0.750
	MS-Random	0.839	0.841	0.891	0.694	0.864	0.834	0.747	0.825	0.850	0.831
	ATC-NE	0.733	0.724	0.822	0.640	0.721	0.711	0.778	0.783	0.799	0.783
	ATC-MC	0.742	0.746	0.813	0.630	0.730	0.725	0.814	0.780	0.774	0.739

Table 22: Full in-domain τ metrics for all test domains for CNN and BERT models trained on AWS.

Model	Measure	Train Domain				
		`slate`	`fiction`	`telephone`	`travel`	`government`
CNN	MS-SSMBA	0.664	0.655	0.753	0.692	0.758
	MS-EDA	0.629	0.723	0.786	0.652	0.755
	MS-BT	0.751	0.765	0.637	0.786	0.826
	MS-Random	0.437	0.471	0.546	0.570	0.491
	ATC-NE	0.704	0.662	0.719	0.715	0.722
	ATC-MC	0.703	0.660	0.722	0.716	0.728
BERT	MS-SSMBA	0.713	0.754	0.766	0.785	0.839
	MS-EDA	0.608	0.664	0.781	0.575	0.726
	MS-BT	0.751	0.765	0.637	0.786	0.826
	MS-Random	0.437	0.471	0.546	0.570	0.491
	ATC-NE	0.722	0.796	0.740	0.737	0.701
	ATC-MC	0.745	0.791	0.791	0.719	0.696

Table 23: Full in-domain τ metrics for all train domains for CNN and BERT models trained on MNLI.