

[DL project]Sufficient Invariant Learning Benchmark

Wonsang Yun

June 10, 2024

Abstract

It is possible that an invariant features are only visible in the training set but not in the test set. Current SOTA(State-Of-The-Art) models also do not perform well in such environments. Furthermore, there are no metrics to evaluate these environments. Therefore, this paper aims to provide such an evaluation metric.

1 Introduction

Sufficient Invariant Learning(Taero Kim et al., 2022) [1] is machine learning in an environment where there are many invariant features in the training set and only a few features in the test set. We want to make model to focus on all features, not just one. However, even SOTA models focus on only one feature and perform very poorly when that feature is removed. In this paper, we set up an environment where multiple invariant features are present in the training set and only some features are present in the test set and provide it as a benchmark(like [2]). We also show which models are robust in a sufficient environment.

2 Datasets

2.1 Handwriting Datasets

Handwriting Dataset which can be downloaded in [Kaggle](#) is a dataset with multiple handwritten alphabets in a single image. This dataset has handwritten words as labels. In training set, we transformed this dataset following,

- label 1: Images containing T and U, (6474 images)
- label 2: Images containing S and I, (17074 images)
- label 3: Images containing O and M, (8256 images)

Note that Each label does not contain features from the other labels.

In the test set, we transformed the dataset so that only one of the two features was present (e.g., in label 1, we collected data containing only T but not U). See details in Figure 1, Figure 2. And split validation and train set by a ratio of 1:9.

You can transform dataset when download handwriting dataset and execute *file_lo.ipynb*. In my file, since I already execute *file_lo.ipynb*, you don't have to execute *file_lo.ipynb*.

2.2 pretreatment

Since Handwriting Dataset has images of different size, we resize images as (72, 388). Then, we perform normalization(using `transforms.Normalize()`) as $\mu = 0, \sigma^2 = 1$. You can see details in *dataLoader.py*.

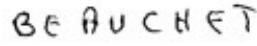


Figure 1: Example of label 1 in training set. You can see there is T and U in handwriting.

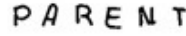


Figure 2: Example of label 1 in test set. You can see there is T but not U in handwriting.

3 Algorithm

3.1 baseline model

We use ResNet18 to perform this benchmark.[3] We build ResNet18 at *resnet.py*

3.2 method

We evaluate method following,

- Adam: Just evaluate naive resnet18 model
- SAM: Sharpness-Aware Minimization method[4] with Adam. (We build it at *SAM.py*) SAM aims to minimize neighborhood of loss. Then, parameter space of loss function can be less-sharpness. It helps increase robustness (for domain shift).
- Blur: Blur the images and train them. It can help increase robustness.[5] We sampled iid by the amount of pixels in the image from the normal distribution $\mathcal{N}(0, 1)$. And some $\epsilon(0 \leq \epsilon \leq 1)$, we performed $\text{blurred_img} = (1 - \epsilon) * \text{image} + \epsilon * \text{sample}$. ϵ is linearly increasing or decreasing as epochs change according to initial and end ϵ . (We build it at *Blur.py*)
- Adversarial Example Generation(FGSM): Adversarial Example Generation[6] image for model and train it. This method also have ϵ and linearly increasing or decreasing as epochs change according to initial and end ϵ . (We build it at *Adversarial.py*)

3.3 How to Execute

(pandas, numpy, torch, torchvision should be required.) If you want to execute learning rate = $1e-4$ with method Adam, execute 'python run.py --epochs 100 --batch_size 32 --lr $1e-4$ --method 'Adam' in terminal. If you want to execute learning rate = $1e-3$ with method SAM($\rho=0.05$), execute 'python run.py --epochs 100 --batch_size 32 --lr $1e-3$ --method 'SAM' --rho 0.05' in terminal. (You can see details in *run.py* and *train.py*)

3.4 Model Selection

Since our assumption is models can not access test domain, we selected model which has best validation accuracy (at last epochs). Batch size and epochs are set to fixed values which are 32, 100 respectively. Here is sweep of each method.

- Adam: We performed for learning rate in $1e-5$ $5e-5$ $1e-4$ $5e-4$. The result of model selection is lr: $1e-4$.
- SAM: We performed for learning rate in $1e-5$ $5e-5$ $1e-4$ $5e-4$ for ρ in 0.001 0.005 0.01 0.05 0.1. The result of model selection is lr: $1e-4$, ρ : 0.005.
- Blur: We performed for learning rate in $1e-5$ $5e-5$ $1e-4$ $5e-4$ for initial ϵ for 1.0 0.5.(End of ϵ is fixed as 0) The result of model selection is lr: $1e-4$, initial ϵ 1.0.
- Adversarial Example Generation(FGSM): We performed for learning rate in $1e-5$ $5e-5$ $1e-4$ $5e-4$ for initial ϵ for 0.5.(End of ϵ is fixed as 0) The result of model selection is lr: $5e-5$, initial ϵ 0.5. (Because of the limitation of computation cost, we just performed one ϵ .)

Method	val	T	U	S	I	O	M	avg	worst
Adam	99.120	42.950	15.109	93.380	85.831	25.036	22.835	47.524	15.109
SAM	99.465	56.043	27.787	94.174	84.540	40.333	34.215	56.182	27.787
Blur	99.057	46.862	30.971	93.191	85.310	51.034	43.498	58.478	30.971
FGSM	96.730	35.038	22.816	76.560	75.182	39.878	37.841	47.886	22.816

Table 1: Result of Experiment.

4 Result

You can see result of Table here Table 1. *val* is Validation accuracy which is same distribution with train set. T, U, S, I, O, M are test accuracy that contains only one letter of the word. *avg* is average test accuracy of T, U, S, I, O, M. *worst* is worst test accuracy of T, U, S, I, O, M.

All results of sweep are available in *result* folder. In *result* folder, we report Validation loss and accuracy once every epochs and accuracy and loss of test sets once every 5 epochs.

5 Analysis and Conclusion

The high performance of S and I is due to the largest number of data in label 2. It implies that in an OOD(Out-Of-Distribution) situation, the train set is more likely to predict with labels that have a lot of data in them.

All methods have low performance in test *avg* accuracy. It implies that even robust models cannot perform well in sufficient learning environment. Nevertheless, it can be seen that the robust model(SAM, Blur, FGSM) performs better than the Naive model in both *avg* and *worst*. Especially, it can be seen that Blur method has the best performance in both *avg* and *worst*. Our hypothesis is one of features of images is obscured by blurring images in training set. This may act as a data augmentation.

This analysis above shows that new methods for sufficient invariant learning must be required to achieve model robustness. As we saw in the performance of blur method, if we can remove the features that the model focuses on from the training set, we expect it to perform well in sufficient invariant learning environment.

References

- [1] Taero Kim, Sungjun Lim, and Kyungwoo Song. Sufficient invariant learning for distribution shift. *arXiv preprint arXiv:2210.13533*, 2022.
- [2] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.
- [3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [4] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.
- [5] Lukas Vogelsang, Sharon Gilad-Gutnick, Evan Ehrenberg, Albert Yonas, Sidney Diamond, Richard Held, and Pawan Sinha. Potential downside of high initial visual acuity. *Proceedings of the National Academy of Sciences*, 115(44):11333–11338, 2018.
- [6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.