Machine intelligence

Biolab | Cesena Campus | University of Bologna

MEMO: Test Set Robustness via Adaptation and Augmentation

Biolab research meeting
Gabriele Graffieti
29/10/2021

MEMO: Test Time Robustness via Adaptation and Augmentation

Marvin Zhang[1], **Sergey Levine**[1], **Chelsea Finn**[2]
[1] UC Berkeley, [2] Stanford University

Abstract

While deep neural networks can attain good accuracy on in-distribution test points, many applications require robustness even in the face of unexpected perturbations in the input, changes in the domain, or other sources of distribution shift. We study the problem of *test time robustification*, i.e., using the test input to improve model robustness. Recent prior works have proposed methods for test time adaptation, however, they each introduce additional assumptions, such as access to multiple test points, that prevent widespread adoption. In this work, we aim to study and devise methods that make no assumptions about the model training process and are broadly applicable at test time. We propose a simple approach that can be used in any test setting where the model is probabilistic and adaptable: when presented with a test example, perform different data augmentations on the data point, and then adapt (all of) the model parameters by minimizing the entropy of the model's average, or *marginal*, output distribution across the augmentations. Intuitively, this objective encourages the model to make the same prediction across different augmentations, thus enforcing the invariances encoded in these augmentations, while also maintaining confidence in its predictions. In our experiments, we demonstrate that this approach consistently improves robust ResNet and vision transformer models, achieving accuracy gains of 1-8% over standard model evaluation and also generally outperforming prior augmentation and adaptation strategies. We achieve state-of-the-art results for test shifts caused by image corruptions (ImageNet-C), renditions of common objects (ImageNet-R), and, among ResNet-50 models, adversarially chosen natural examples (ImageNet-A).

1 Introduction

Deep neural network models have achieved excellent performance on many machine learning

MEMO: Test Set Robustness via Adaptation and Augmentation

- Authors: Marvin Zhang[1], Sergey Levine[1], Chelsea Finn[2]
  - [1] UC Berkeley, [2] Stanford University

- https://arxiv.org/abs/2110.09506
- Probably submitted to ICLR 2022
- A lot of interest in the social networks (Twitter)

Original problem: distribution shift

Deep neural network *models have achieved excellent performance on many machine learning problems, such as image classification, but* ***are often brittle and susceptible to issues stemming from distribution shift****. For example,* ***deep image classifiers may degrade precipitously in accuracy when encountering input perturbations, such as noise or changes in lighting*** *(Hendrycks & Dietterich, 2019)* ***or domain shifts*** *which occur naturally in real world applications (Koh et al., 2021). Therefore,* ***robustification of deep models against these test shifts is an important and active area of study****.*

How to address distribution shift

- Training time robustification
  - Use larger model or datasets
  - Adversarial training
  - Aggressive data augmentation
- Using these techniques requires modify the training procedure
  - Not always feasible
  - Heavy computation on non-public data (?)
  - No information about the test points that the model has to predict
    - E.g. I don’t know how the test set is different from the training
    - I would want to use such information if I can

Test time robustness

- Several works have proposed methods for improving accuracy via adaptation after seeing the test data
  - Updating a subset of model's parameters
  - Updating statistics (e.g. batch norm)
- Proposal: specific test input may be leveraged in order to improve the model's prediction on that point.

They explicitly use test data to improve the performance on the same data!

MEMO: Marginal Entropy Minimization with One test point

- Makes direct use of pre-trained model (no further training on the train data or custom training procedures).
- No assumption on model's training procedure or architecture.
- Only a single test input at a time is used to optimize the model.
- Can work alongside other test time adaptation methods

MEMO: Marginal Entropy Minimization with One test point

- Take a set *A* of augmentation functions.
- Sample *B* augmentation functions from *A* (using a uniform distribution).
- Augment the test input *x* with the augmentation functions.
- Compute the marginal distribution of the predictions of the network:

$$\bar{p}_\theta(y|\mathbf{x}) \triangleq \mathbb{E}_{\mathcal{U}(\mathcal{A})}\left[p_\theta(y|a(\mathbf{x}))\right] \approx \frac{1}{B}\sum_{i=1}^{B} p_\theta(y|\tilde{\mathbf{x}}_i)\,,$$

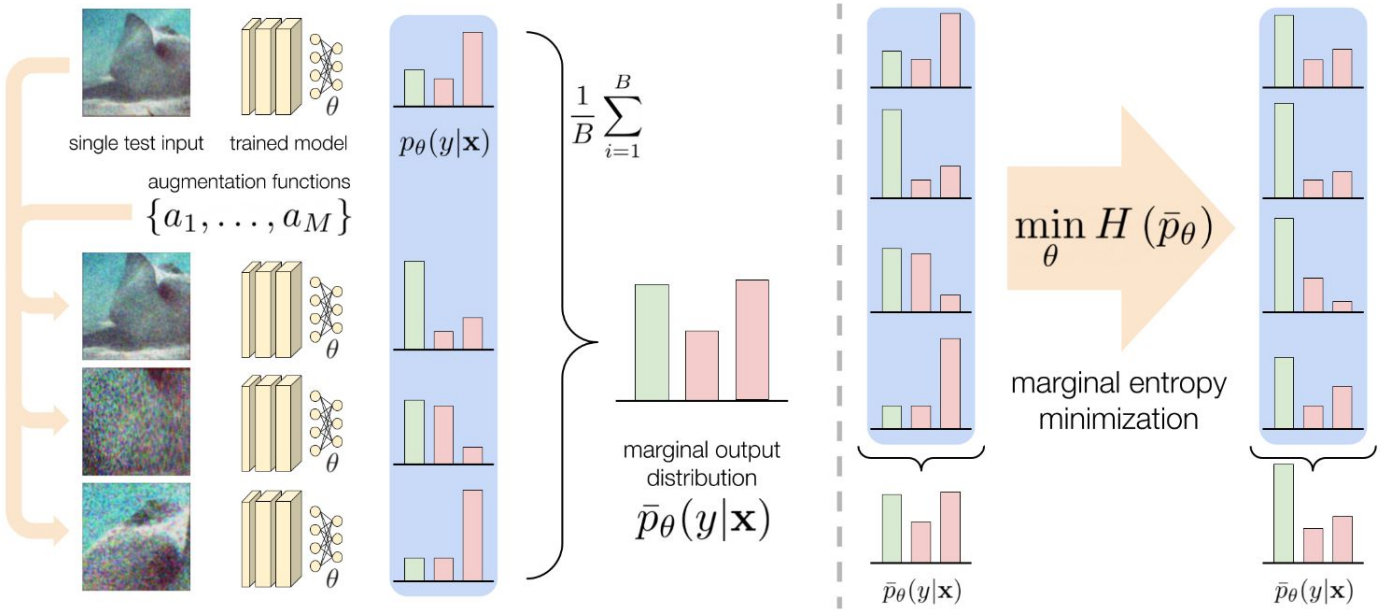
MEMO: Marginal Entropy Minimization with One test point

- Calculate the entropy of the marginal

$$\ell(\theta; \mathbf{x}) \triangleq H\left(\bar{p}_\theta(\cdot|\mathbf{x})\right) = -\sum_{y \in \mathcal{Y}} \bar{p}_\theta(y|\mathbf{x}) \log \bar{p}_\theta(y|\mathbf{x}) \,.$$

- Entropy is:
  - High if the marginal probability is spread-out -> different augmentation changes the prediction of the network.
  - Low if the marginal is peaked -> for many augmentation the network remains confident on the prediction.

MEMO: Marginal Entropy Minimization with One test point



MEMO: Marginal Entropy Minimization with One test point

- Once calculated, use the entropy as a loss and minimize it.
- Optimize all the model parameters (one step) to minimize the entropy.
- Predict the original test point *x*

Algorithm 1 Test time robustness via MEMO

Require: trained model f_θ, test point $\mathbf{x}$, # augmentations B, learning rate η, update rule G

1: Sample $a_1, \ldots, a_B \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\mathcal{A})$ and produce augmented points $\tilde{\mathbf{x}}_i = a_i(\mathbf{x})$ for $i \in \{1, \ldots, B\}$

2: Compute Monte Carlo estimate $\tilde{p} = \frac{1}{B} \sum_{i=1}^{B} p_\theta(y|\tilde{\mathbf{x}}_i) \approx \bar{p}_\theta(y|\mathbf{x})$ and $\tilde{\ell} = H(\tilde{p}) \approx \ell(\theta; \mathbf{x})$

3: Adapt model parameters via update rule $\theta' \leftarrow G(\theta, \eta, \tilde{\ell})$

4: Predict $\hat{y} \triangleq \arg\max_y p_{\theta'}(y|\mathbf{x})$

Results

	ImageNet-C mCE ↓	ImageNet-R Error (%)	ImageNet-A Error (%)
Baseline ResNet-50 (He et al., 2016)	76.7	63.9	100.0
+ TTA	77.9 (−1.2)	61.3 (+2.6)	98.4 (+1.6)
+ Single point BN	71.4 (+5.3)	61.1 (+2.8)	99.4 (+0.6)
+ MEMO (ours)	69.9 (+6.8)	58.8 (+5.1)	99.1 (+0.9)
+ BN ($N = 256, n = 256$)	61.6 (+15.1)	59.7 (+4.2)	99.8 (+0.2)
+ Tent (Wang et al., 2021)	54.4 (+22.3)	57.7 (+6.2)	99.8 (+0.2)
+ DeepAugment+AugMix (Hendrycks et al., 2021a)	53.6	53.2	96.1
+ TTA	55.2 (−1.6)	51.0 (+2.2)	93.5 (+2.6)
+ Single point BN	51.3 (+2.3)	51.2 (+2.0)	95.4 (+0.7)
+ MEMO (ours)	**49.8** (+3.8)	**49.2** (+4.0)	94.8 (+1.3)
+ BN ($N = 256, n = 256$)	45.4 (+8.2)	48.8 (+4.4)	96.8 (−0.7)
+ Tent (Wang et al., 2021)	**43.5** (+10.1)	**46.9** (+6.3)	96.7 (−0.6)
+ MoEx+CutMix (Li et al., 2021)	74.8	64.5	91.9
+ TTA	75.7 (−0.9)	62.7 (+1.8)	89.5 (+2.4)
+ Single point BN	71.0 (+3.8)	62.6 (+1.9)	91.1 (+0.8)
+ MEMO (ours)	69.1 (+5.7)	59.4 (+3.3)	**89.0** (+2.9)
+ BN ($N = 256, n = 256$)	60.9 (+13.9)	61.6 (+2.9)	93.9 (−2.0)
+ Tent (Wang et al., 2021)	54.0 (+20.8)	58.7 (+5.8)	94.4 (−2.5)
RVT*-small (Mao et al., 2021)	49.4	52.3	73.9
+ TTA	53.0 (−3.6)	49.0 (+3.3)	**68.9** (+5.0)
+ Single point BN	48.0 (+1.4)	51.1 (+1.2)	74.4 (−0.5)
+ MEMO (ours)	**40.6** (+8.8)	**43.8** (+8.5)	69.8 (+4.1)
+ BN ($N = 256, n = 256$)	44.3 (+5.1)	51.0 (+1.3)	78.3 (−4.4)
+ Tent (Wang et al., 2021)	46.8 (+2.6)	50.7 (+1.6)	82.1 (−8.2)

Discussion

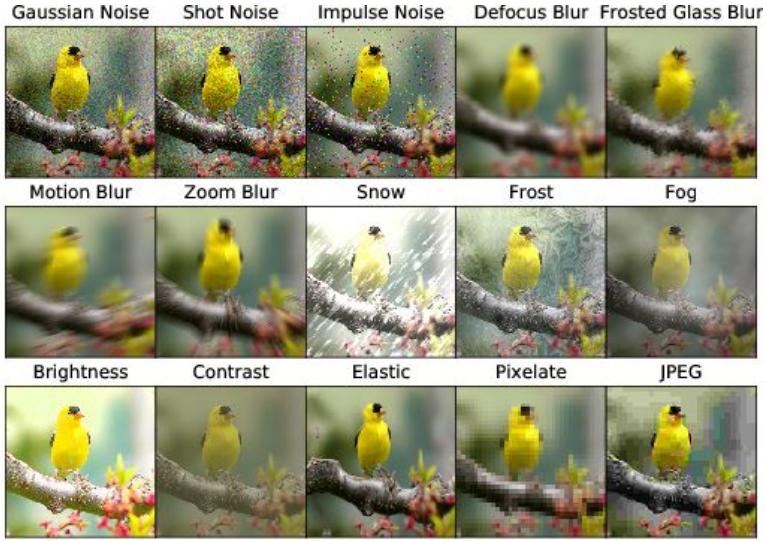
- Inference computationally expensive
  - An optimization step have to be performed for every test point.
  - Impossible to use mini-batches during test.
- If test set changes or the model needs to be adapted to a changing environment the methods do not work.
  - Degenerate solutions far away from the pretrained model.
  - Model always predict a constant label with max probability (entropy minimized).

Questions

- Can this procedure can be considered **training on the test set**?
  - The model is adapted using the same test point to classify. It is natural that the model performs better.
  - All the parameters are optimized.
  - Possible to make more than one optimization step (re-sampling augmentation functions).
- How fast the model degrades?
  - Since we are optimizing the model on the entropy, it is easy to minimize it by always yielding the same label.
  - What about an incremental test set?
- Can this method can be considered an **hybrid and strange form of CL** without having the labels?
  - We can **take some ideas here for unsupervised or weak-supervised learning**.

Appendix: ImageNet-C

15 types of algorithmically generated corruptions from noise, blur, weather, and digital categories. Each type of corruption has five levels of severity, resulting in 75 distinct corruptions.



Appendix: ImageNet-R

ImageNet-R(endition) contains art, cartoons, deviantart, graffiti, embroidery, graphics, origami, paintings, patterns, plastic objects, plush objects, sculptures, sketches, tattoos, toys, and video game renditions of ImageNet classes.

Appendix: ImageNet-A

Natural adversarial examples -- real-world, unmodified, and naturally occurring examples that cause machine learning model performance to significantly degrade.