Adversarial Domain Adaptation with K-dimensional Discriminator head

Prasanna B [1]

Abstract

In unsupervised Domain Adaptation labeled source data and unlabeled target domain data are provided and it is assumed that both have same set of classes. Adversarial Discriminative Domain Adaptation (ADDA) is a work which efficiently solves the above problem. In this work, I implement ADDA and modify 2-head Discriminator used in ADDA to a K-head Discriminator in order to leverage the availability of source classes during Adaptation and examine its impact over performance.

1. Technical Details

1.1. Dataset

In this work, MNIST has been used as the source domain dataset and USPS as the target domain dataset. MNSIT consists of 60000 train samples and 10000 test samples, of which each sample is a 28x28 grayscales image. USPS dataset used here consists of 7291 train samples and 2007 test samples of which each sample is a 28x28 grayscale image.

1.2. Network Architecture

The Neural network has 2 parts:

- Feature Encoder part consisting of a Convolutional layer with 20 filters followed by Maxpooling and ReLU activation and followed by another Convolutional layer with 50 filters followed by Dropout, Maxpooling and ReLU activations. Finally, a Fully Connected Layer with 500 neurons is added.
- Feature Classifier part consists of a Fully connected layer with 10 neurons, each for a label in digits datasets.
- Discriminator consists of 3 Fully Connected Layers with 500 neurons in the first 2 layers. The last classifier layer has 2 neurons in the first method, as it was proposed in ADDA, each for source and target classes. In the second method I have tried, the last layer of the Discriminator contains K+1 neurons, where K is the no. of classes in the data set, where K are for source labels and 1 is for target.
- Although Discriminator is trained with 11 classes, the target encoder is still trained with only 2 classes since target encoder's aim is to just fool the discriminator into thinking the features it gives belong to source data set, no matter which label. To achieve this, the first 10 probabilities, which belong to the source classes, are added into a single value, which will act as the probability of source class, while the 11th term would be the probability of the target class and these will be used in target enocder's loss function.

[1]Department of Computational and Data Sciences, Indian Institute of Science, Bangalore, India.. Correspondence to: Prasanna B<bprasanna@iisc.ac.in>.

2. Results

The accuracy after training on MNIST data before/after unsupervised adapting to USPS for different methods is shown in table below.(Values are in %)

Model	Source-only	Domain-Adapted
2-head Discriminator	85.81	95.54
11-head Discriminator	85.69	96.08

From the above results, we can see that the models are successfully adapting to the target domain, since the accuracy on target data after domain adaptation is significantly higher than the accuracy obtained on the model trained just on the source data. Also, training the Discriminator on 11 labels (10 for source classes and 1 for target) instead of just 2 (1 for source and 1 for target) improves the performance slightly.

3. Novel Contributions

Discriminator with (K+1) output classes has been tried and implemented successfully. Modified several parts of code in [3], written in PyTorch=0.2.0 to work on PyTorch=1.10.0. Modified Image transformation parameters and Normalization parameters as with the original parameters accuracy after adaptation was just 7.68%.

4. Tools used

We have used the following tools:

- PyTorch
- Other standard Python Libraries

5. References

[1] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. *Adversarial Discriminative Domain Adaptation.* In CVPR, 2016

[2] Aaron Chadha, Yiannis Andreopoulos. *Improved Techniques for Adversarial Discriminative Domain Adaptation.* In IEEE Transactions on Image Processing, 2019.

[3] https://github.com/corenel/pytorch-adda