Domain adaptation evaluation with neuron coverage

Chiori Azuma, Hiroki Yaemori, Yuki Shindo, Kan Hishiyama, ComputerMind Corp., Japan (ai.sales@compmind.co.jp)

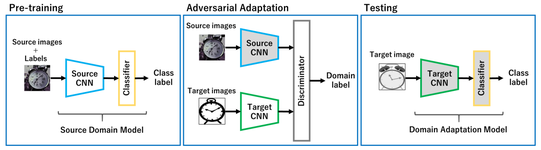
Abstract

Neuron coverage test was introduced to evaluate the test data quality. It observes the firing of neurons.
In this study, we used it to measure the similarity of different trained models.
Domain adaptation is a technique which is used to adapt a model trained on a dataset in a given domain (source domain) to a dataset in a new unseen domain (target domain). We attempted to apply neuron coverage to compare domain adaptation model with their original models.
We trained a domain adaptation model with a standard open dataset for domain adaptation and evaluated the its effectiveness to the target domain data.
In order to visualize the firing of neurons, we used a function of the inference library presented in our poster session at GTC2020.

Related Work

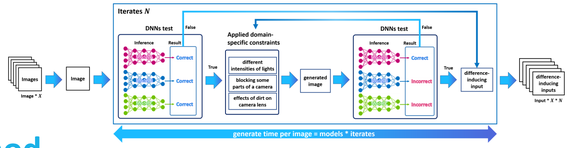
- **Domain Adaptation**

  Domain adaptation refers to applying an already trained model to a different domain. Various domain adaptation studies have been done so far, but in this study,
  we will use Adversarial Discriminative Domain Adaptation (ADDA) [1], published in 2017.



- **Neuron Coverage Test**

  Neuron coverage test is a way to measure the test data sufficiency for evaluating trained models of deep learning.
  In this study, we used DeepXplore [2], published in 2017.



Method

Domain adaptation is performed using a public dataset, the Office-Home dataset [3].
It includes four domain datasets (Art, Clipart, Product and Real World).

We trained the following 2 types of models:

1. Source domain model:
   Train with each of the four domain datasets.
2. Domain adaptation model:
   Adapt each of the four source domain models to the other three target domains.

For classification tasks in domains different from trained domains, we compared the domain adaptation models with the source domain models using accuracy and neuron coverage.

Environment

We used Nvidia GeForce RTX 3080Ti graphics card and the inference library "DeepEye Predictor" [4][5] to perform the calculations.

Results

We calculated the accuracy and neuron coverage by classifying data in each domain using each model. The results are as follows.
In domain adaptation model tables, green value indicates the improve from source domain model, red value indicates the worsen from it.

- **Accuracy** (%)

  Source domain model

Source Domain \ Target Domain	Art	Clipart	Product	Real World
Art	65.28	31.82	63.89	62.50
Clipart	64.58	96.59	90.28	83.75
Product	58.33	51.14	93.52	87.50
Real World	75.00	95.45	84.72	89.17

  Domain adaptation model

Source Domain \ Target Domain	Art	Clipart	Product	Real World
Art		31.82	62.50	62.50
Clipart	64.58		88.89	86.25
Product	58.33	53.41		96.25
Real World	75.00	93.18	83.33	

- **Neuron coverage** (%)

  Source domain model

Source Domain \ Target Domain	Art	Clipart	Product	Real World
Art		35.74	74.94	69.79
Clipart	80.76		92.14	87.29
Product	87.47	71.64		85.45
Real World	75.55	86.67	71.36	

  Domain adaptation model

Source Domain \ Target Domain	Art	Clipart	Product	Real World
Art		35.74	74.94	69.87
Clipart	80.76		91.86	89.40
Product	87.57	72.21		86.24
Real World	75.64	85.75	71.67	

The accuracy and neuron coverage results show similar tendency with improvements when Product is the source domain or when Real World is the target domain.
On the other hand, accuracy tends to decrease when using Real World as source domain or using Product as target domain.

Conclusion & Future Work

Conclusion

- When using ADDA, the source domain is more likely to get the effect of domain adaptation if the data does not contain extraneous information, such as a white background. In the Office-Home dataset, Product domain is more suitable for domain adaptation.
- Since models with higher coverage rates tend to have improved accuracy, models that attempted to adapt to the domain by firing unused nodes are more likely to have improved accuracy.
- As a result, we may be able to say that neuron coverage allows us to evaluate domain adaptation.

Future Work

- In this study, the coverage rate was calculated for the entire architecture, and we didn't evaluate in detail which coverage rate is improved, the domain-adapted feature extractor or the classifier of the source domain model. In the future, we would like to calculate the coverage for the models with successful domain adaptation by separating the feature extractor and classifier, then confirm that the feature extractor improves the coverage rate and the classifier's coverage rate does not change.
- We want to input images of the target and source domains of the same class and check the IoU of firing in each layer and see that the IoU increases as we approach the output layer.
- In addition to the coverage evaluation function of our DeepEye Predictor used for verification this time, we would like to implement another evaluation function in the future and expand it as a library that can perform inference + evaluation at high speed. TensorRT is one of the most prominent library used to speedup inference time, but we would like to not just have fast inference functionality but also the ability to visualization.

Open Source (coming soon)

- **DeepEye Predictor**

  ComputerMind's "DeepEye Predictor" is a C++ inference library that has been in development since 2019. DeepEye (a deep learning software and hardware package sold by our company) also uses DeepEye Predictor as one of its deployment target.
  The main feature of our library is that there are very few dependent libraries, and CUDA tuning reduces GPU idle time which results in fast inference, also, since the library is developed in C ++ it can easily be incorporated into embedding computing devices.
  GitHub: https://github.com/ComputermindCorp

Reference

[1] Eric Tzeng, Adversarial Discriminative Domain Adaptation, 2017
[2] Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17). ACM, New York, NY, USA, 1-18. GitHub: https://github.com/peikexin9/deepxplore
[3] Office-Home Dataset, http://hemanthdv.org/OfficeHome-Dataset/
[4] Yuki Shindo, Hiroki Yaemori, and Chiori Azuma, Fast deep learning library implemented in C++, GTC2019, Poster Session, 2019.
[5] Yuki Shindo, Hiroki Yaemori, and Chiori Azuma, Acceleration of test data quality assurance technology using neuron coverage, GTC2020, Poster Session, 2020.

Company HP:
https://www.compmind.co.jp/

Computermind