A Method for Accurate Driver Status Monitoring using Domain Adaptation after Deployment

Jaeyoon Lee, Ki-Seok Chung
Hanyang University, Seoul, Korea
zxc1421@hanyang.ac.kr, kchung@hanyang.ac.kr

Abstract

With rapid advances in AI technology, autonomous driving is close to becoming a reality. Nevertheless, most car accidents are still caused by the driver's forward-looking negligence, and driver's intervention is still required. Therefore, monitoring the driver status has become an essential task for preventing car accidents. Many studies have attempted to solve the concern by applying a pre-trained neural network. However, the performance of a pre-trained neural network is deteriorated due to the distributional shift between training data and field data. In this paper, we propose a method to retain the performance of the pre-trained neural network by mitigating the detrimental effect of the distributional shift. In addition, we will show that the proposed method can be implemented on an embedded platform where memory size and computing power are limited.

Keywords: Domain Adaptation, Driver Status Monitoring, Action Recognition, Unbalanced Dataset

1. Introduction

With rapid advances in AI technology, autonomous driving is close to becoming a reality. Nevertheless, drivers are still required to look ahead while driving; otherwise, the forward-looking negligence may lead to a severe accident. Therefore, monitoring the driver status has become crucial and various deep learning methods have been proposed with legitimate datasets. However, a neural network trained with data collected in a specific environment often fails to perform correctly with respect to the data collected on-site. Presumably, it is mainly owing to differences between the on-site data and the data used for training before deployment. This difference is often referred to as distributional shift that may pose various challenges to deal with differences in subject types, camera angles, camera types, etc.

In this paper, a domain-adaptation method is proposed to mitigate the distributional shift between the two datasets. In many domain-adaptation studies, the dataset used for training before deployment is commonly referred to as the source dataset, and the dataset collected from the field is referred to as the target dataset. The domain-adaptation method typically regains neural network's performance on the target dataset by mitigating the distributional difference between the two different datasets. To prove the effectiveness of the proposed method under a realistic circumstance, domain adaptation on a set of unbalanced and unlabeled target data is attempted.

In addition, the driver status monitoring engine should be implemented with the consideration of computing power and memory size of the deployment platform since many embedded platforms in automobiles have limited computing power and memory size. In particular, owing to the limited memory size, it is almost impossible to access the source data after deployment. To keep these in mind, we model a 2D convolutional network (CNN) that judges each frame to meet limited computing power in the proposed method.

2. Related Work

2.1 Driver Status Monitoring Dataset

As the driver-monitoring task gets lots of attention lately, various datasets have been created. The most well-known ones are addressed as follows.

Driver Monitoring Dataset (DMD) DMD [2] consists of a total of 41 hours of video data for a total of 39 subjects. The dataset is multi-labeled for performing tasks such as fatigue and drowsiness, gaze collection, head-pose estimation, hand position, and interaction with the inside object, etc. For driver-distraction tasks, the dataset is labeled into 14 classes of {change gear, drinking, hair and makeup, phonecall left, phonecall right, radio, reach backseat, reach side, safe driving, standstill or waiting, talking to passenger, texting left, texting right, unclassified}.

State Farm Distracted Driver Detection (StateFarm) StateFarm was distributed on the Kaggle website in 2021 as a competition dataset. This dataset was created for the driver-distraction detection task. The dataset consists of 10 driver's actions of {c0 : safe driving, c1 : texting-right, c2 : talking on the phone, c3 : texting-left, c4 : talking on the phone, c5 : operating the radio, c6 : drinking, c7 : reaching behind, c8 : hair and makeup, c9 : talking to passenger}. Data were collected for 26 subjects and consisted of 22424 frames of the front view only.

Drive & Act Drive & Act [3] consists of 29 long sequence data with 12 hours of video data for 15 subjects. The dataset consists of images taken by a multi-view camera of 6 views divided into RGB, IR, and Depth video, respectively. Labels are annotated with a total of 83 hierarchical activity labels.

Among these datasets, our domain-adaptation method uses the DMD dataset and the StateFarm dataset to perform driver action recognition.

2.2 Action Recognition

Action-recognition task classifies the behavior of a given person. The data for this task is commonly composed of various types. Usually, the data is video data in successive frames. Each frame could be a data of skeleton data, IR, and depth data not just a RGB data.

2D convolution 2D CNN processes a single image. A neural network based on 2D convolution requires a relatively small amount of computation and memory. The authors of [1] experimented with a single image based on a 2D CNN.

3D convolution 3D CNN is the most common data processing method on the action recognition task. Temporal features that cannot be extracted by a 2D CNN can be extracted by a 3D CNN. MoviNets [4] and SlowFast [5] improved the performance for action recognition based on a 3D CNN. However, compared to 2D CNN, it has the disadvantage of requiring a large amount of memory and computation.

Transformer Methods for recording SOTA performance on action recognition are based on method called Transformer, such as Swin Transformer [6]. Considering the huge amount of computation and memory of Transformer, it is not practically useful for embedded platforms where memory and computing power are limited.

Multi-Modality Several methods to improve performance by fusing various types of data have been proposed. Different features from multiple datasets are fused to get better representation. For example, SGM-Net [7] extracts some features of skeleton data through a graph convolutional network and the RGB data through a CNN. PoseC3D [8] converts the skeleton data into a 2D image that is to be fused with the RGB image. However, to collect multi-modal data, multiple high-end cameras are required, which is not practical.

2.3 Domain Adaptation

Domain Adaptation (DA) is a type of transfer learning. DA aims to improve neural network's performance on the target data when the network is trained with related, yet different source data. Scenarios of DA are categorized based on two criteria. The first criterion is whether the target data is labeled. Based on this, the scenario is categorized as supervised, semi-supervised, or unsupervised learning. The second criterion is the relation between the label space of the source data and the label space of the target data. If two label spaces are identical, it is called a closed set scenario. A partial set is when the source label space is a superset of the target label space. An open set represents a case where the source label space and the target label space have labels that do not share.

MCD [9] measures the maximum distance between two domains using the neural network's output and minimizes it under the unsupervised scenario. MMD [10] and JMMD [11] compute a metric called maximum mean discrepancy (MMD) between the two domains and try to minimize the value. However, MCD, MMD, and JMMD assume that accesses to the source data are available. However, domain adaptation may suffer from practically inefficient data transmission between devices. Even though SHOT [12] shares the unsupervised scenario with the methods mentioned above, in contrast, it does not access the source data after deployment.

3. Proposed Method

In the proposed method, we apply domain adaptation to the driver-action recognition task to minimize accuracy drop after deployment. Unlike the conventional domain adaptation and action recognition situation, the proposed method assumes that the driver-action recognition engine is deployed in an embedded device that has limited computing power and memory size. First, considering the lack of computing power, a 2D-CNN is used instead of Transformer or a 3D-CNN. Second, a single-modal dataset is used because collecting multi-modal data using multiple high-end cameras inside a vehicle is not practical. Finally, in consideration of the limited memory capacity of the embedded environment, a source-free domain adaptation method without inefficient data transmission is applied.

The training is divided into two stages. The first stage is to create a trained model (F_S) before deployment. This step assumes that the source data ($\{x_s, y_s\}$) is accessed. The training is processed by trying to minimize a cross-entropy loss. In the second stage, after deployment, an additional training is performed with the pre-trained model and the unlabeled target data ($\{x_t\}$) without access to the source data.

3.1 Source Model Training

First, a model that classifies the source data well is generated. The loss to minimize is computed as follows:

$$\mathcal{L}_{Xent}(F_S; \mathcal{X}_s) = -\mathbb{E}_{x_s \in \mathcal{X}_S} \sum y_s \log(F_S(x_s)) \quad (1)$$

where $\mathcal{L}_{Xent}$ denotes the cross-entropy loss.

3.2 Adaptation to Target

With the source-trained model, adaptation to the target data proceeds without further accesses to the source data. The authors of [13] claimed the transferred neural network does not deviate much from the source-trained neural network if the source domain and the target domain are related. Likewise, in the proposed method, the classifier of the pre-trained neural network is frozen and transferred, and only the encoder is fine-tuned according to the target data. The reason for fine-tuning only the encoder is because it has been reported in [14] and [15] that the encoder tailored to the target data yields better performance.

As the source data is not accessible after the deployment, we cannot calculate the distributional shift between two datasets directly. Instead, we assume that the output vector of the target data would be similar to one-hot vector if the distributional shift is alleviated. This is because the output of the neural network trained with source data also derives one-hot vector-like output. Therefore, to alleviate the domain shift between the two datasets, the entropy is maximized by minimizing the entropy loss ($\mathcal{L}_{ent}$), as shown below.

$$\mathcal{L}_{ent}(F_S; \mathcal{X}_T) \\ = -\mathbb{E}_{x_t \in \mathcal{X}_T} \sum_{k=1}^{11} \delta_k(F_S(x_t)) \log \delta_k(F_S(x_t)) \quad (2)$$

where $\delta_k(a)$ denotes the k^{th} element of vector a. This makes the output vector look like a one-hot vector. However, if only the entropy loss term is minimized, the target data tends to be classified into a single class. Therefore, we minimize the Kullback-Leibler (KL) divergence ($\mathcal{L}_{KL}$) between the output of the target data and target class distribution D_T:

$$\mathcal{L}_{KL}(F_S; \mathcal{X}_T) = D_{KL}(\hat{p}, D_T) \\ = \sum_{k=1}^{11} \hat{p}_k log \hat{p}_k - \hat{p}_k \log(D_T) \quad (3)$$

where $\hat{p}_k = \mathbb{E}_{x_t \in \mathcal{X}_T}[\delta_k(F_S(x_t)]$.

$\mathcal{L}_{KL}$ prevents the output value from being overwhelmingly classified as one class.

3.3 Label Correction

Even if the two terms ($\mathcal{L}_{ent}$ and $\mathcal{L}_{KL}$) are minimized, some data may be assigned to a wrong label. To prevent this incorrect assignment, we train the model with a pseudo labeling. Since the target data does not have labels, data is initially labeled with the model's output. In the subsequent training, the data with a high-loss value due to a high confidence level is identified and the mislabeling of such data is corrected by assigning it to the index of the logit that produces the largest value among all the output logits. With the corrected labels, the cross-entropy loss of the target data and the pseudo label is minimized. In the experiment, the pseudo-labeling is performed once every ten epochs.

4. Experiments

4.1 Data preparation

In our experiments, the DMD dataset is chosen as the source dataset and the StateFarm data as the target dataset because such selection is thought to imitate a real situation well. The label space of DMD and StateFarm overlaps significantly. Therefore, if a proper data preprocessing is applied, the StateFarm label space becomes a subset of the DMD label space. Accordingly, it becomes an experimental environment suitable for domain adaptation. Specifically, the labels of StateFarm are properly reassigned to match the labels of DMD. Also, the labels that are divided according to the direction are integrated. Eleven labels created through the reassignment and the integration are: {change gear, drinking, hair and makeup, phonecall, radio, reach backseat, reach side, safe drive, standstill or waiting, talking to passenger, texting}.

Domain adaptation is performed for one of the 26 subjects of StateFarm, and the entire set is divided into three datasets; train, validation, and test sets with a ratio of 3:1:1. The final accuracy is the accuracy on the test set when the best performance is achieved on the validation set.

4.2 Implementation Details

We conducted experiments on the ResNet-50 network, which is composed of an encoder and a classifier. The classifier comprises three fully connected layers, two ReLU activation functions, and a dropout layer with dropout rates of 0.5 and 0.2. The network is first loaded from an ImageNet pre-trained checkpoint. After loading from the checkpoint, the network is trained with the DMD dataset before deployment. The network is trained by the Adam optimizer with a learning rate of 0.001 for 300 epochs, a batch size of 64, a weight decay of 0.0001, and a cosine learning rate scheduler.

Domain adaptation is done with the StateFarm dataset with the Adam optimizer, a learning rate of 0.0001, a batch size of 64, a weight decay of 0.0005, a cosine learning rate scheduler for 40 epochs. Both datasets are transformed by scaling to (3, 224,224), rotating up to 25 degrees with a probability of 0.6, a random cropping up to 20 percent of an image, and a skewing probability of 0.3 with a magnitude of 0.3.

4.3 Results of Domain Adaptation

The results of domain adaptation for the 26 subjects were averaged and compared with those of well-known existing methods.

Methods	Dataset
	DMD→StateFarm
No Training	12.84
Cross-Entropy	14.59
SHOT	28.22
Ours	**34.64**

Table 1 Comparison of Test Accuracy(%) on StateFarm

In Table 1, 'No Training' denotes a method where the StateFarm data is simply provided as input into the DMD-trained model. The accuracy, which is the probability of getting the correct answer, is 12.84%, which is almost random. 'Cross-Entropy' denotes a method that minimized the cross-entropy loss of the target data and its pseudo label which is decided as the output of the first inference. 'Cross-Entropy' is better than 'No Training', but the accuracy is still quite low. 'SHOT', one of the state-of-the-art source-free unsupervised domain adaptation methods, shows an accuracy of 28%. Among all the compared methods, our method shows the highest accuracy of 34.64%.

5. Conclusion

Although autonomous driving techniques are rapidly advanced, driver intervention is still compulsory, and the driver's negligence in looking ahead accounts for the most significant proportion of the causes of car accidents. Accordingly, the driver action recognition task has received much attention, and there have been lots of efforts to apply deep learning. However, the accuracy drops when the pre-trained network is deployed mainly owning to the difference between the pre-trained data and the on-site data. This paper proposed a method to regain the performance by applying domain adaptation. The proposed method is simple yet effective so that it can be run on an embedded platform with limited computing power and memory size.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2020-0-01304, Development of Self-learnable Mobile Recursive Neural Network Processor Technology)

References

[1] Cañas, Paola, et al. "Detection of Distraction-related Actions on DMD: An Image and a Video-based Approach Comparison." VISIGRAPP (5: VISAPP). 2021.

[2] Ortega, Juan Diego, et al. "Dmd: A large-scale multi-modal driver monitoring dataset for attention and alertness analysis." European Conference on Computer Vision. Springer, Cham, 2020.

[3] Martin, Manuel, et al. "Drive&act: A multi-modal dataset for fine-grained driver behavior recognition in autonomous vehicles." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019.

[4] Kondratyuk, Dan, et al. "Movinets: Mobile video networks for efficient video recognition." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021.

[5] Feichtenhofer, Christoph, et al. "Slowfast networks for video recognition." Proceedings of the IEEE/CVF international conference on computer vision. 2019.

[6] Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." arXiv preprint arXiv:2103.14030 (2021).

[7] Li, Jianan, et al. "SGM-Net: Skeleton-guided multimodal network for action recognition." Pattern Recognition 104 (2020): 107356.

[8] Duan, Haodong, et al. "Revisiting Skeleton-based Action Recognition." arXiv preprint arXiv:2104.13586 (2021).

[9] Saito, Kuniaki, et al. "Maximum classifier discrepancy for unsupervised domain adaptation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[10] Long, Mingsheng, et al. "Learning transferable features with deep adaptation networks." International conference on machine learning. PMLR, 2015.

[11] Long, Mingsheng, et al. "Deep transfer learning with joint adaptation networks." International conference on machine learning. PMLR, 2017.

[12] Liang, Jian, Dapeng Hu, and Jiashi Feng. "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation." International Conference on Machine Learning. PMLR, 2020.

[13] Kuzborskij, Ilja, and Francesco Orabona. "Stability and hypothesis transfer learning." International Conference on Machine Learning. PMLR, 2013.

[14] Tzeng, Eric, et al. "Adversarial discriminative domain adaptation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[15] Shu, Rui, et al. "A dirt-t approach to unsupervised domain adaptation." arXiv preprint arXiv:1802.08735 (2018).