Dynamic Adaptation on Non-stationary Visual Domains

Sindi Shkodrani[1,2(✉)], Michael Hofmann[2(✉)], and Efstratios Gavves[1(✉)]

[1] University of Amsterdam, Amsterdam, The Netherlands
e.gavves@uva.nl
[2] TomTom, Amsterdam, The Netherlands
{sindi.shkodrani,michael.hofmann}@tomtom.com

Abstract. Domain adaptation aims to learn models on a supervised source domain that perform well on an unsupervised target. Prior work has examined domain adaptation in the context of stationary domain shifts, i.e. static data sets. However, with large-scale or dynamic data sources, data from a defined domain is not usually available all at once. For instance, in a streaming data scenario, dataset statistics effectively become a function of time. We introduce a framework for adaptation over non-stationary distribution shifts applicable to large-scale and streaming data scenarios. The model is adapted sequentially over incoming unsupervised streaming data batches. This enables improvements over several batches without the need for any additionally annotated data. To demonstrate the effectiveness of our proposed framework, we modify associative domain adaptation to work well on source and target data batches with unequal class distributions. We apply our method to several adaptation benchmark datasets for classification and show improved classifier accuracy not only for the currently adapted batch, but also when applied on future stream batches. Furthermore, we show the applicability of our associative learning modifications to semantic segmentation, where we achieve competitive results.

1 Introduction

Domain adaptation aims to adapt classifiers trained on source domains to novel unlabeled target domains, where a domain shift, namely a difference in distribution statistics, is expected [8,34]. Typically, the domain shift is considered within the context of "closed", static domains, implicitly assuming datasets available in their entirety at adaptation time [8,36]. However, in realistic applications data collection is not static nor closed but "open", giving rise to non-stationary domain shifts [11].

Consider e.g. social media feeds, or urban imagery taken from inside a car. These images often arrive in "bundles" with different distribution statistics, due to, for instance, being collected in different cities or with different weather conditions (see Fig. 1). If we were to consider these bundles as isolated domains, we wouldn't be exploiting the available unlabeled data entirely. In addition, if the

L. Leal-Taixé and S. Roth (Eds.): ECCV 2018 Workshops, LNCS 11130, pp. 158–171, 2019.
https://doi.org/10.1007/978-3-030-11012-3_12

Fig. 1. Distribution shift across stream batches in GTA5

distribution changes gradually over time in a streaming-like fashion, being able to adapt over bundles sequentially may benefit real-time predictions on future incoming data bundles. In streaming data this distribution shift over time is called concept drift, and the incoming stream is usually too large to be held in memory, therefore it is processed in data bundles which are later discarded.

As an adaptation method, we look into associative learning proposed by Haeusser *et al.* [15,16], which uses association of embeddings in latent space and has been shown to work well for domain adaptation and semi-supervised learning. However, associative domain adaptation makes the implicit assumption that the class probability distributions between the source and the target domains are similar at adaptation time. This assumption cannot be guaranteed to hold when the target dataset is not well known in advance, such as in "open" datasets the class probability statistics may change dynamically or in tasks where class statistics across domains may vary a lot. An example of such a task is semantic segmentation. To this end, the associations between source and target embeddings need to be performed while taking into account the non-stationary changes of the class probability statistics.

This work makes three contributions. First, we argue that domain adaptation is important beyond static domain datasets, including continuously collected datasets whose statistics are non-stationary. For dynamic datasets domain adaptation should be able to adapt to the evolving statistics. Second, starting from associative domain adaptation [15] we show that the similar class distribution assumption between domains hurts adaptation. We therefore reformulate the approach to make the adaptation loss invariant to the inevitable non-stationary changes on the class distribution statistics. Third, we present two applications of our proposed approach, one on adapting streaming image classification, where the streaming data distribution changes over time, and one on domain adaption for semantic segmentation (see Fig. 2), where the source and target datasets have inherently different class statistics.

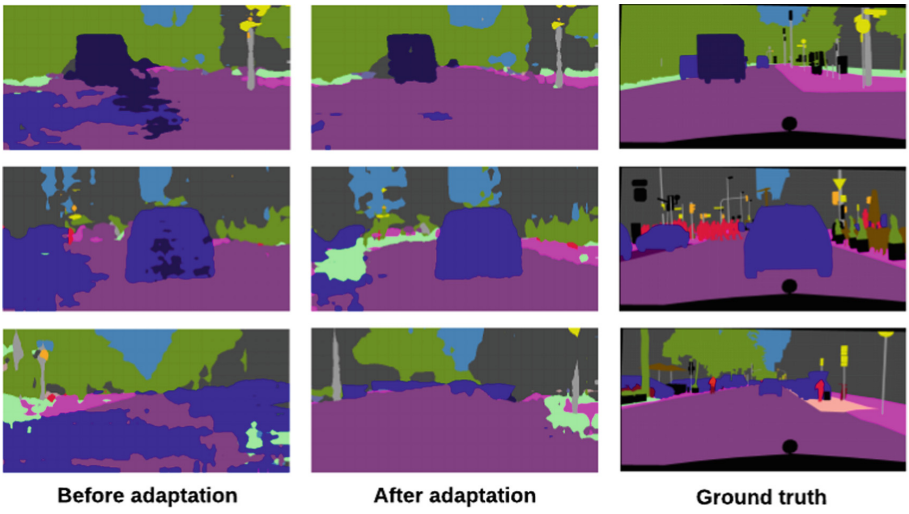

Fig. 2. Adaptation results for semantic segmentation on Cityscapes

2 Related Work

Domain Adaptation. A handful of domain adaptation methods revolve around discrepancy-based adaptation [23,24,31], for instance, [22] use a multi-kernel maximum mean discrepancy (MMD) minimization approach. Other methods are data reconstruction-based and often use reconstruction with e.g. autoencoders, as an auxiliary task to learn invariant features [5,12,13].

Another category is adversarial approaches. Adversarial discriminative methods use a classifier to discriminate between domains during training and ensure feature invariance for source and target [10,35]. Adversarial generative methods use a generative adversarial network (GAN) [14] to learn a mapping between source and target images by interleaving the task loss, mapping generator and discriminator loss [3,18].

Domain adaptation for semantic segmentation was recently pioneered by [19] with an adversarial discriminative based approach. Similarly [6] use discriminators for feature invariance, but for different parts of an image grid. [28] use a standard GAN approach to have a generator network learn the mapping while a discriminator network distinguishes between real and fake images. [38] split the original segmentation network into three output branches where the first two generate pseudo-labels for the third branch. [39] adopt a curriculum learning approach for by solving easy to difficult tasks to achieve adaptation.

Associative Learning. Introduced by [16], learning by association was initially applied to semi-supervised learning. [15] use associations between source and target to close the domain gap for classification achieve competitive results across different domain adaptation benchmarks for classification. The advantage of this

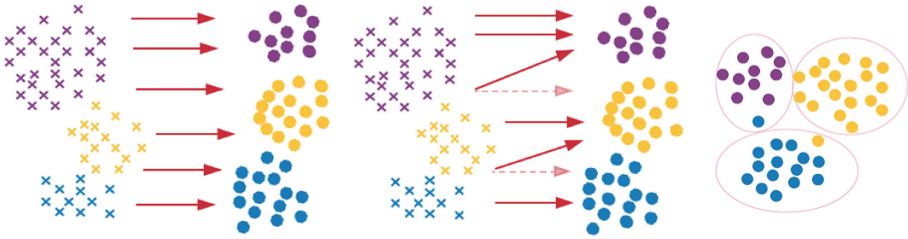

Fig. 3. Associative domain adaptation for unequal class distributions. Crosses represent the source domain and circles represent the target. Arrows represent source to target probabilities. (a) Uniformly distributed visit loss. (b) Intuition of correcting wrong associations by balancing the visit loss according to class distributions. (c) Cluster estimates to approximate class distribution in target.

method compared to discrepancy-based approaches is that it does not require a choice of kernel and extra hyper-parameters that come with them.

Streaming Data Classification. Approaches that deal with streaming data are either passive approaches that use a single classifier or an ensemble [30,37] or active ones where an extra decision is made on whether to update the classifier. Most often classification algorithms such as Decision Trees, Rule-Based and Nearest Neighbor are used, whereas adjustments in neural network architectures to account for streaming have been proposed [1]. [2] use a complex sampling and filtering mechanism for active training and a random forest based classifier. [33] use a micro-cluster nearest neighbor which makes use of statistical summaries for data streams. Not many works look into exploiting unsupervised data for improving data stream classifiers. [32] use semi-supervised feature learning to adjust k-nearest neighbor weights. To our best knowledge, we are the first to explore this direction for image classification with modern deep architectures.

3 Method

3.1 Associative Domain Adaptation

We start from two datasets, source and target. The source dataset, $D^S = \{x_i^S, y_i^S\}, i = 1, ..., N^S$, comprises of N^S image samples with embeddings x_i^S, annotated by one-hot vectors $y_i^S = [y_{ic}^S], c = 1, ..., C$, which equals to 1 if the image x_{ic}^S belongs to class c, and 0 otherwise. The target dataset, $D^T = \{x_j^T\}, j = 1, ..., N^T$, comprises only of image embeddings which belong to the same set of classes, $c = 1, ..., C$; however, no class annotations are available for retraining or fine-tuning. Between the source and target datasets there is a domain shift in the distribution of their respective embeddings, thus $p(x^S) \neq p(x^T)$. The goal, therefore, is to adapt a classifier trained on the source dataset to work well for the target.

Associative domain adaptation [15] adapts by considering an additional adaptation loss during training on top of the standard task-specific loss,

$\mathcal{L} = \mathcal{L}_{task} + \mathcal{L}_{assoc}$. Specifically, the associative domain adaptation is decomposed into a walker and a visit loss,

$$\mathcal{L}_{assoc} = \mathcal{L}_{walker} + \beta \mathcal{L}_{visit}, \tag{1}$$

where β is a weighting coefficient. Central to the associative domain adaptation is the affinity matrix $A \in \mathbb{R}^{N_S \times N_T}$, which contains elements a_{ij} proportional to how likely the i-th embedding in the source domain, x_i^S, is to be associated with the j-th embedding in the target domain, namely $a_{ij} \propto p(x_j^T | x_i^S)$. Learning embeddings that yield an affinity matrix that minimizes the loss in Eq. (1) is the goal of associative domain adaptation.

The walker and the visit losses have complementary objectives. The objective of the walker loss is to encourage the source embeddings to lie close after adaptation to source embeddings of the same class. As no class labels are available in the target dataset, however, this objective is reformulated. Specifically, after double transition from the source to the target and back to the source, the starting and finishing source class labels should minimize the cross-entropy loss with respect to a normalized equality matrix $E = \{e_{ik}\}$, namely

$$\mathcal{L}_{walker} = \sum_{i,k} e_{ik} \log \left[p(x_k^S | x_j^T) \cdot p(x_j^T | x_i^S) \right], \tag{2}$$

where x_j^T is the closest embedding in the target set and $e_{jk} = \frac{y_i^S \cdot y_k^S}{N_S}$.

The walker loss alone, however, can lead to degenerate solutions, where the transition probabilities are learned to associate source embeddings only with a few relevant yet "easy" target embeddings. To mitigate this, the *visit loss* encourages that all target embeddings are equally visited. This is achieved by a minimizing cross-entropy objective

$$\mathcal{L}_{visit} = \sum_{j} v_j \log p(x_j^T | x_i^S), v_j = 1/N_T \tag{3}$$

where $v_j = 1/N_T$.

3.2 Associative Domain Adaptation for Unequal Class Distributions

Associative domain adaptation implicitly assumes that the source and target distributions are similar on batch level during adaptation. The reason is that for the visit loss to be minimized in Eq. (3) it is assumed that the ideal target is the average over the size of the target dataset, $v_j = 1/N_T$. [15] consider a smaller β for the visit loss, if the class distributions between the source and target datasets are unequal. However, this solution implicitly expects access to the adaptation set in order to tune β. In addition, simply receiving a weaker signal from the visit loss does not exploit the full adaptation capacity and might enforce wrong associations, as we illustrate in Fig. 3(a).

As we want target embeddings to be visited by the same-class source embeddings, intuitively they should be visited at a rate proportional to the difference

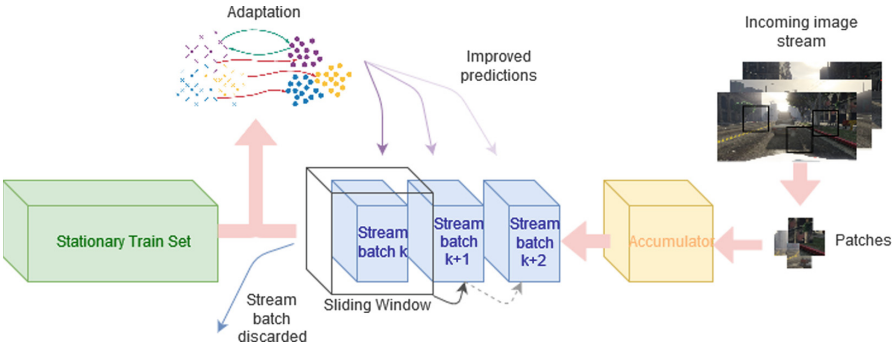

Fig. 4. The streaming setup uses a pre-trained model from a stationary supervised set. The model is then sequentially adapted to the incoming stream batches.

between the source and target class distributions, as shown in Fig. 3(b). We can formalize the intuition by adding a weighting coefficient in front of v_j and reformulating Eq. (3) as:

$$\mathcal{L}_{visit} = \sum_j \gamma_j v_j \log p(x_j^T | x_i^S), \gamma_j = \frac{p(Y^S = y_j^T)}{p(Y^T = y_j^T)} \tag{4}$$

namely weighted by the ratio of class probabilities at the source and target for the correct class of the target embedding. Clearly, we cannot directly compute the ratio $p(Y^S = y_j^T)/p(Y^T = y_j^T)$, as we would need to know the true class of the target embedding y_j^T. However, we propose a way to estimate them.

Although we have no control on the target dataset, we do have control over the source dataset for which the labels are available, thus when constructing the mini-batch based on which we will perform the adaptation, we can first sample the source uniformly such that all class probabilities are equal in the source dataset, i.e. $p(Y^S = y_j^T) = const$. Consequently, from a probabilistic perspective it is not important which particular class the j-th target embedding belongs to, alleviating the necessity to make a soft prediction for the class label of the j-th embedding.

What remains to compute the weighting coefficient γ_j is computing the class probability $p(Y^T = y_j^T)$ for the j-th embedding. It is logical to expect that same-class embeddings cluster together for a modern classifier to be able to discriminate between classes. We can retrieve the class cluster around an embedding sample in an unsupervised manner and compute the probability based on cluster size. We rely on unsupervised clustering to estimate class probabilities in the batch. The approximation holds true under the assumption that the clusters are well aligned to the means of the respective, optimal classifiers. In practice, we consider hierarchical agglomerative clustering, which experimentally appears to allow for good alignment between the obtained clusters and works well when clusters have very different sizes. We illustrate the process in Fig. 3(c).

3.3 Dynamic Domain Shift in Streaming Data

Let us consider a pre-collected annotated set $D^S = \{x_i^S, y_i^S\}, i = 1, ..., N^S$ with embeddings x_i^S, with one-hot labels y_i^S and an incoming stream of image data that needs to be classified. At every time step $\tau = 1, ., K$, the stream is accumulated in a streaming data batch D_τ^T. Due to the *concept drift*, i.e. distribution shift over time, a classifier $f_0(\theta)$ trained to minimize $\mathcal{L}_{task}(\theta, D^S)$ will perform worse on the streaming batches. Being able to produce accurate predictions as soon as a stream batch comes in is crucial. Changing the models over time aims to account for the concept drift. A second problem to account for in streaming is the size of the incoming data. Usually only a small part of this data can be stored in memory. One way to deal with this is to have a mechanism in place that selects the data to be stored; another way is to be able to use and then discard all the data coming into the stream.

We simulate a streaming scenario where the stationary training set D^S is pre-collected and first used for off-line training of a predictive model $f_0(\theta)$. Incoming stream data batches D_τ^T are small compared to the stationary set D^S, but the whole stream cannot be stored in memory, so at a time step $\tau = k$ only a set of $D_k^T, D_{k+1}^T, ...D_{k+w}^T$ is available, where w is a storage window size. A classifier $f_{k-1}(\theta)$ trained on the stationary set and adapted to $D_1^T, ...D_{k-1}^T$ sequentially is available. We adapt to D_k^T by minimizing the objective

$$\arg\min_\theta \mathcal{L}_{task}(f_{k-1}(\theta), y^S) + \mathcal{L}_{walker}(\theta, D^S, D_k^T) + \beta\mathcal{L}_{visit}(\theta, D^S, D_k^T)$$

The benefits of this approach are twofold. First, adapting to D_k^T improves prediction results on D_k^T itself in an unsupervised manner without extra annotation. Second, the predictions improve for $D_{k+1}^T, D_{k+2}^T...$ and so on in a cascade fashion, since distribution in incoming sets is more likely to be similar to the previous stream sets nearby than the stationary source, especially if we would use a sliding window over incoming sets. For simplicity we take a window size of 1. An illustration is provided in Fig. 4. In our setting, we extract patches from the GTA5 dataset and do patch-wise classification in order to demonstrate the working of our setup with a simpler task. We expect a similar behavior for more complex tasks such as semantic segmentation and object detection.

3.4 Dynamic Domain Shift in Semantic Segmentation

Having relaxed the distribution assumption, associative domain adaptation can be applied to tasks where source and target class distributions in a batch are not uniform or uniformity cannot be approximated, such as semantic segmentation. Consider a source dataset $D^S = \{x_i^S, y_{i,H\times W}^S\} i = 1, ..., N^S$, where H, W are image dimensions, is annotated at pixel level. The target images $D^T = \{x_j^T\}, j = 1, ..., N^T$ are available without annotations. Using modern segmentation architectures, we can consider embeddings extracted from a mid-network layer which contains downsampled data. Using a DeepLab-V2 [4] architecture, we extract embedding $x_{i'}^S$ at pixel level in decoder layers before bilinear

upsampling which are 8 times downsampled in each spatial dimension. We downsample the label annotations and use $y^S_{i',U\times V}$, where $U = H/8$ and $V = W/8$ together with downsampled embeddings for adaptation.

An important consideration when adapting for dense prediction is the choice of affinity matrix $A \in \mathbb{R}^{N_S \times N_T}$ between embeddings, where $a_{ij} \propto p(x^T_j|x^S_i)$. In [15], $p(x^T_j|x^S_i)$ is computed as softmax over rows of A, i.e. $p(x^T_j|x^S_i) = \exp(a_{ij})/\sum_{j'} \exp(a_{ij'})$, where $a_{ij} = x^S_i \cdot x^T_j$ is the dot product between embedding vectors. The unnormalized dot product as an affinity is unbounded and can cause very small probability values for the softmax, which may lead to exploding gradients. We mitigate this by using an affinity measure based on Euclidean distance. In addition, we observe that the dimensionality of pixel embeddings for semantic segmentation is crucial for convergence. If too large, the gradients propagated are noisy and adaptation not very effective. However, dimensionality has to be large enough to allow for similar embeddings to group together but still preserve discriminable structures in latent space. For this, we add an *embedding layer* in the decoder where dimensionality can be adjusted for the task.

4 Experiments and Results

We validate the performance of the proposed domain adaptation method under different settings for domain class distribution divergence. First, we show the effect of increased class distribution divergence on associative domain adaptation [15] and how we can recover accuracy drops with our formulation. Second, we evaluate on a visual stream classification setting, where data and class distributions change over time. Third, we further validate the proposed method on domain adaptation for semantic segmentation The code, models and datasets will all become available upon publication.

4.1 Classification Under Different Class Distributions Between Domains

We report our results on several image classification adaptation benchmarks. For digit classification we adapt on MNIST [21] → MNISTM [10], SVHN (Street View House Numbers) [26] → MNIST and Sythetic Digits [10] to SVHN [26]. Next, we adapt for street sign classification from Synthetic Signs dataset [25] to German Traffic Sign Recognition Benchmark [29]. As a last benchmark, CIFAR-10 [20] → STL-10 [7] adaptation is performed. Out of the 10 classes present in STL-10 and CIFAR-10, 9 of these overlap so they can be used for domain adaptation.

We report experiments after changing KL-divergence between the source and target class distributions, to quantify the effect of unequal class distributions for domain adaptation. In Table 1 we report the accuracies over the datasets when class distribution divergence increases for associative domain adaptation, as well as the proposed method.

Table 1. Adaptation accuracy as KL-divergence of source to target class distributions in a batch increases. The oracle version uses the true target class probabilities and serves as an upper bound.

Src -Tgt divergence	Method	Datasets				
		MNIST-MNISTM	SVHN-MNIST	Synth Dig.-SVHN	Synth Signs-GTSRB	CIFAR10-STL10
	Source only	64.0	69.4	85.8	95.4	52.7
	Target only	93.6	99.5	94.2	98.1	99.8
KL = 0.05	Adapted using [1]	87.6	97.0	91.9	96.2	61.3
	Ours	88.3	97.2	92.6	96.5	61.2
	Ours with oracle*	90.0	97.2	92.8	97.5	61.5
KL = 0.2	Adapted using [1]	85.2	94.3	87.6	95.9	57.6
	Ours	87.6	96.9	89.9	95.6	58.3
	Ours with oracle*	90.1	97.8	92.8	97.3	61.2
KL = 0.4	Adapted using [1]	81.7	94.2	87.1	95.5	53.4
	Ours	83.8	94.9	88.0	95.3	56.2
	Ours with oracle*	89.8	96.6	92.6	94.1	61.4

First, as expected, larger KL-divergence between source and target usually leads to worse accuracy for associative domain adaptation. Second, the proposed method improves recognition after domain adaptation, especially for larger class distribution divergence, and especially for tasks where the classifiers are not already near maximal adaptation capacity.

To further derive insights, we also include results with an oracle-weighted visit loss that use the target class distributions (theoretical upper bound). Although our off-the-shelf agglomerative clustering does not always approximate the batch statistics perfectly, it does come considerably close to the oracle-weighted score and almost always outperforms the unweighted approach. In addition, using oracle test statistics the proposed method often comes close to the recognition accuracies of classifiers trained directly on the target domain indicating that our theoretical reasoning is correct. We conclude that when we expect a dynamical domain shift, where class distributions between the source and target change, our approach is more robust to for domain alignment.

4.2 Streaming Data Classification

Next, we evaluate the method on a streaming data scenario, where the class distributions are expected to be different between source and target. To simulate a streaming data scenario, we note that the popular synthetically generated and finely annotated GTA5 dataset [27] is in fact ordered sequentially. Video-like fragments can be observed throughout the dataset, and a shift in distribution over time can also be observed, as shown in Fig. 1. We therefore extract patches from GTA5 frame sequences and adapt to a patch-wise classification task, where the label for each patch is equivalent to the dense label for the middle pixel. We use 65×65 patches cropped from a 256×512 downsampled version of the original GTA5 dataset.

Table 2. Streaming classification accuracy per adaptation round. Cells marked "-" indicate the batch hasn't yet entered the stream.

Adaptation Set	Source only	Lag from adaptation timestep						Adapted with [1]
		5	4	3	2	1	0	(lag=0)
SB1	42.72	-	-	-	-	-	**46.72**	45.58
SB2	40.30	-	-	-	-	44.02	**45.22**	44.50
SB3	38.58	-	-	-	40.88	41.48	42.02	**42.13**
SB4	37.85	-	-	40.88	41.52	41.98	**43.00**	42.45
SB5	42.02	-	45.22	45.90	45.93	45.73	**47.51**	46.13
SB6	46.78	50.65	50.70	51.47	51.27	51.33	**52.73**	51.83

We consider a streaming data scenario where a small set of stationary labeled data is pre-collected and available for training. For the stationary data, we sample patches from the first 5,000 images in the GTA5 dataset. About 32,000 patches of 65×65 dimensions are sampled. For the incoming stream we sample patches from bundles of 1,000 images each, collected sequentially. 6,000 patches are sampled from every bundle of images and accumulated in a *streaming batch*. We experiment with adapting six of these sets following the stationary training set.

Several observations follow from the results in Table 2. First, there is indeed a dynamical domain shift when considering visual streams instead of static datasets. When considering the classifiers trained only on the source, there is considerable fluctuation on the recognition accuracy over time. Note that this is not always harmful, *e.g.* for streaming batches 5 and 6 accuracy improves, presumably because the shift between target and source is smaller.

Second, the proposed streaming adaptation method yields considerable and constant accuracy improvements over the source-only scores, no matter the source-only recognition accuracy. Also, the proposed method yields modest but consistent improvements over standard associative domain adaptation [15].

Third, as expected, best adaptation is achieved when adapting and testing on the same stream batch (lag = 0). However, adapting with some lag allows for accurate adaptation as well. We conclude that for visual streams, where we cannot store the data and we cannot always immediately adapt, dynamical domain adaptation is valuable.

4.3 Semantic Segmentation

Last, we validate the proposed method on the task of domain adaptation for semantic segmentation of urban street scenes. This is an application where source and target class statistics cannot be expected to align, especially on batch level where adaptation happens.

We adapt on the GTA5 $\rightarrow$ Cityscapes adaptation benchmark, which is important to domain adaptation as adapting from synthetic to real data

Table 3. GTA5 to Cityscapes domain adaptation. The last two rows show results on adapting with the unweighted version of the method and the distribution independent one.

	road	sidewalk	building	wall	fence	pole	light	sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorb.	bike	mIoU	Pixel Acc
NoAdapt	33.8	23.2	67.5	18.2	**20.1**	18.1	15.9	**21.8**	66.9	18.0	72.4	33.0	6.5	25.0	**15.8**	19.3	6.0	8.4	**5.8**	26.1	72.8
Adapt (no wght.)	59.9	29.8	67.1	16.2	10.7	22.9	13.2	9.1	**78.0**	**33.4**	**75.7**	41.9	0.3	32.3	12.4	16.5	5.7	2.9	0.1	27.8	78.8
Adapt (est. wght.)	**63.8**	**31.3**	**68.4**	**19.4**	19.6	**23.2**	**17.6**	11.8	62.9	22.7	61.0	**52.1**	**7.8**	**42.5**	13.4	**22.1**	**6.2**	**9.1**	0.1	**29.2**	**81.9**

provides potential for exploiting very easily rendered synthetic sets. GTA5 contains 24,966 images with resolution 1914×1052, of which 12,500 are used for training and around 6,800 for validation. Cityscapes contains 5,000 pixel-level annotated images of 2048×1024 resolution, of which 2,975 images for the training set and 500 images for validation are available. We run our experiments with images from both datasets downsampled to 512×256 size.

As a base segmentation network we use DeepLab-V2 [4] with a ResNet-50 [17] backbone. We extend the original DeepLab-V2 architecture with a D-dimensional embedding layer that can be adjusted for experiment purposes and report results with $D = 64$. The embedding layer is placed before the bilinear upsampling part of the decoder, yielding embeddings that are 8 times downsampled in each spatial dimension. In this way we can not only adapt to more compressed information on pixel level embeddings, but also fit embedding metrics in reasonable memory even for large datasets.

We use $\beta = 0.5$ for the visit loss, adjusted for the magnitude of the loss values. We use the respective training sets of GTA5 and Cityscapes as the domains for training, test on the Cityscapes *val* set, and report the results in Table 3.

First, we train for 30K iterations on source only for GTA5, using pre-trained ImageNet [9] weights for the ResNet-50 encoder part of the network. We observe that the proposed distribution independent approach consistently improves standard associative domain adaptation, both in terms of mIoU and pixel accuracy.

Further, the proposed method improves standard domain adaptation on 15 out of the 19 categories. Standard associative domain adaptation is still better for large classes with near constant class frequency (e.g. *vegetation*, *terrain*, *sky*), since adaptation over these would overrule smaller classes in a batch. Interestingly, the proposed method seem to improve significantly (6–10%) over mid-size classes, such as *car*, *bus* and *person*, where indeed we expect larger class frequency fluctuations. We conclude that our approach is promising for domain adaptation of complex dense prediction tasks such as semantic segmentation, and potentially, integrating with the streaming techniques above, to video semantic segmentation.

5 Conclusion

We have presented a robust and distribution independent associative learning method for domain adaptation. Our formulation accounts for realistic scenarios where source and target data distribution in a batch cannot be approximated to be equal. A novel setup for dynamic domain adaptation that adapts over unlabeled data in order to improve classifier prediction over time for streaming data has been proposed. We have shown that we can exploit unsupervised data to achieve improvements over several streaming batches without additionally annotated samples. Using our associative domain adaptation formulation and architecture considerations we achieve competitive results for semantic segmentation.

Having considered a dynamic time-shifting distribution setup and shown dense prediction adaptation results, we lay the grounds for a framework that can potentially work well with dense prediction tasks for streaming video data such as video segmentation.

References

1. Aggarwal, C.C.: A survey of stream classification algorithms (2014)
2. Annapoorna, P.S., Mirnalinee, T.: Streaming data classification. In: 2016 International Conference on Recent Trends in Information Technology (ICRTIT), pp. 1–7. IEEE (2016)
3. Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., Krishnan, D.: Unsupervised pixel-level domain adaptation with generative adversarial networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, p. 7 (2017)
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2018)
5. Chen, M., Xu, Z., Weinberger, K., Sha, F.: Marginalized denoising autoencoders for domain adaptation. arXiv preprint arXiv:1206.4683 (2012)
6. Chen, Y., Li, W., Gool, L.V.: ROAD: reality oriented adaptation for semantic segmentation of urban scenes. In: CVPR (2018)
7. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 215–223 (2011)
8. Csurka, G.: Domain adaptation for visual applications: a comprehensive survey. arXiv preprint arXiv:1702.05374 (2017)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 248–255. IEEE (2009)
10. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(1), 2096–2130 (2016)
11. Gavves, E., Mensink, T., Tommasi, T., Snoek, C., Tuytelaars, T.: Active transfer learning with zero-shot priors: reusing past datasets for future tasks. In: Proceedings ICCV 2015, pp. 2731–2739 (2015)

12. Ghifary, M., Kleijn, W.B., Zhang, M., Balduzzi, D., Li, W.: Deep reconstruction-classification networks for unsupervised domain adaptation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 597–613. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_36
13. Glorot, X., Bordes, A., Bengio, Y.: Domain adaptation for large-scale sentiment classification: a deep learning approach. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 513–520 (2011)
14. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
15. Haeusser, P., Frerix, T., Mordvintsev, A., Cremers, D.: Associative domain adaptation. In: International Conference on Computer Vision (ICCV), vol. 2, p. 6 (2017)
16. Haeusser, P., Mordvintsev, A., Cremers, D.: Learning by association-a versatile semi-supervised training method for neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
18. Hoffman, J., et al.: CyCADA: cycle-consistent adversarial domain adaptation. arXiv preprint arXiv:1711.03213 (2017)
19. Hoffman, J., Wang, D., Yu, F., Darrell, T.: FCNs in the wild: pixel-level adversarial and constraint-based adaptation. arXiv:1612.02649 (2016)
20. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
21. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
22. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. arXiv preprint arXiv:1502.02791 (2015)
23. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. arXiv preprint arXiv:1605.06636 (2016)
24. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Unsupervised domain adaptation with residual transfer networks. In: Advances in Neural Information Processing Systems, pp. 136–144 (2016)
25. Moiseev, B., Konev, A., Chigorin, A., Konushin, A.: Evaluation of traffic sign recognition methods trained on synthetically generated data. In: Blanc-Talon, J., Kasinski, A., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2013. LNCS, vol. 8192, pp. 576–583. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-02895-8_52
26. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, vol. 2011, p. 5 (2011)
27. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: ground truth from computer games. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 102–118. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_7
28. Sankaranarayanan, S., Balaji, Y., Jain, A., Lim, S.N., Chellappa, R.: Unsupervised domain adaptation for semantic segmentation with GANs. arXiv preprint arXiv:1711.06969 (2017)
29. Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C.: The German traffic sign recognition benchmark: a multi-class classification competition. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 1453–1460. IEEE (2011)

30. Street, W.N., Kim, Y.: A streaming ensemble algorithm (SEA) for large-scale classification. In: Proceedings of the seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 377–382. ACM (2001)
31. Sun, B., Saenko, K.: Deep CORAL: correlation alignment for deep domain adaptation. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 443–450. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_35
32. Tan, C., Ji, G.: Semi-supervised incremental feature extraction algorithm for large-scale data stream. Concurr. Comput.: Pract. Exp. **29**(6), e3914 (2017)
33. Tennant, M., Stahl, F., Rana, O., Gomes, J.B.: Scalable real-time classification of data streams with concept drift. Future Gener. Comput. Syst. **75**, 187–199 (2017)
34. Tommasi, T., Patricia, N., Caputo, B., Tuytelaars, T.: A deeper look at dataset bias. In: Csurka, G. (ed.) Domain Adaptation in Computer Vision Applications. ACVPR, pp. 37–55. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58347-1_2
35. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Computer Vision and Pattern Recognition (CVPR), vol. 1, p. 4 (2017)
36. Wang, M., Deng, W.: Deep visual domain adaptation: a survey. arXiv preprint arXiv:1802.03601 (2018)
37. Wang, Y., Li, H., Wang, H., Zhou, B., Zhang, Y.: Multi-window based ensemble learning for classification of imbalanced streaming data. In: Wang, J., et al. (eds.) WISE 2015. LNCS, vol. 9419, pp. 78–92. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-26187-4_6
38. Zhang, J., Liang, C., Kuo, C.C.J.: A fully convolutional tri-branch network (FCTN) for domain adaptation. arXiv preprint arXiv:1711.03694 (2017)
39. Zhang, Y., David, P., Gong, B.: Curriculum domain adaptation for semantic segmentation of urban scenes. In: The IEEE International Conference on Computer Vision (ICCV), vol. 2, p. 6 (2017)