# Drift-Aware Predictive Coding for Adaptation in Changing Environments

Haozhu Wang, Zhangxing Bian, Rui Guo, Anthony Liang, Mingyu Yang
University of Michigan, Ann Arbor, MI, USA
{hzwang,zxbian,guorui,aliangdw,mingyuy}@umich.edu

## Abstract

*In many scenarios, dataset shift continuously over time and that leads to decayed performance of machine learning models. Existing continuous domain adaptation methods either rely on learning representations that are invariant to the dataset shift, or gradually adapt to the shifting domains step by step. However, these methods neglect the temporal ordering of the collected data, and it could lead to suboptimal performance when the underlying domain shift has a learnable dynamics. In this work, we propose a drift-aware predicting coding scheme that is able to represent temporally ordered domains as a sequence of embedding vectors. With the awareness of embedding vectors from past domains, we could have a better adaption to the future domains where the labels are missing. Experiments on two toy datasets and two benchmarks (Rotating MNIST and a Portaits dataset) demonstrate that the proposed method outperforms the baselines without learning the dynamics of domain shifting.*

## 1. Introduction

Machine learning models are often deployed for an extended period of time, e.g., several weeks or a season. However, dataset shift is common for real-life applications, e.g, self-driving, chemical sensors, etc [2, 12]. Frequently labeling new data and updating the model are expensive and sensitive to noisy data. *Domain adaptation* tackles the dataset shift problem by learning representations that are invariant to domains shift [9, 16, 13, 20, 11]. However, most domain adaptation methods neglect the temporal ordering of collected data. Thus, they cannot leverage the underlying domain shift dynamics to better adapt to the shifting environment. Recently, researchers proposed to use self-training to gradually adapt a model trained on a labeled source domain to a target domain, with the aid of multiple temporally ordered intermediate domains [12]. However, the pseudo-labels generated for self-training may be inaccurate when dataset shift is large, leading to poor domain adaptation performance. In addition, the self-training approach does not directly learn the domain shift dynamics, which may lead to inferior performance than a method that directly models the shift dynamics.

We argue that domain shifting dynamics is informative for making predictions in a gradually shifting environment. In this project, we propose to directly learn the environment shift dynamics through representation learning. For each domain, our proposed approach learns a **D**rift-**A**ware **P**redictive(DAP) coding, which embeds information helpful for predicting the next domain,and modulates the base learner(for example, a classifier in a classification problem) to adapt to the shifted new domain. Our method is end-to-end trainable and during training, we sample each domains to build a chain connecting all domains. During testing, in each step, we collect the predictive embedding from last domain and modulate the base learner with that embedding. We show that this approach can capture the shifting dynamics between domains and adjust decision boundaries in each step. Meanwhile, we also show that utilizing unlabelled target domain data can promote the learning process.

To summarize, the main work of this project are: **(1)** We propose to directly learn the shifting dynamics in an gradually shifting environment and prove that these dynamics are informative for making predictions. **(2)** We improve the learned shifting dynamics by exposing unlabelled target domain data through adversarial learning. **(3)** We show the changing of decision boundaries on synthesized data and show quantitative results on Rotating-MNIST and Yearbook Portrait[7] data.

The remainder of this paper is organized as follows. A brief summary of related work is provided in Sec.2 . Details of training and the pose prediction process are explained in Sec.3. Experimental results are reported in Sec.4 and Sec. 5. The paper concludes in Sec.6.

## 2. Related Works

**Unsupervised Domain adaptation** In unsupervised domain adaptation, the goal is to directly adapt from a labeled source domain to an unlabeled target domain. Various prior literature have been proposed including discrepancy-based methods [4], adversary based methods

[17], and reconstruction-based methods [3]. Discrepancy-based methods aim to induce alignment between the source and target domains in some feature space. One popular discrepancy measure is the maximum mean discrepancy (MMD) [4] or the distance between the mean of two domains in some kernel space. Another way to define discrepancy is by training an adversarial discriminator [17] that distinguishes between two domains. However, both these approaches, MMD and adversarial training, are very difficult to optimize in training and often fail to converge and get stuck on a local minimum. Another class of methods directly transform the source images to resemble the target images with generative models [3]. These methods operate directly on image pixels rather than the latent representation space unlike discrepancy based approaches. In this project, we use adversarial methods to promote the learning of domain shift dynamics by aligning the feature distribution after feature fusion of DAP embedding and adapt the discriminator to a continuously indexed domain setting.

**Incremental Domain Adaptation** Closely related to our work are the incremental domain adaptation problems, where the domain is assumed to shift smoothly over time and the goal is to adapt the source domain to multiple target domains incrementally. Different methods for traditional static domain adaptations are applied to perform pair-wise domain adaptation, such as optimal transport [10], adversarial loss[1], generative adversarial networks[19] and linear transform[8]. Recently, self-training is shown to work well for domain shifts with small Wasserstein-infinity distance, both theoretically and empirically [12]. Different from other domain adaptation methods where the goal is to match the distribution of feature vectors, self-training directly makes use of the classifier from the previous domain to provide pseudo labels for current domain, with which the classifier is fine-tuned afterwards. Here, the main difference between our approach and the methods above is that our method explicitly represents the dynamics of the domain shift and directly uses them in both training and testing.

# 3. Method

## 3.1. Problem Settings

We consider a sequence of $N$ source domains $\{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{N-1}\}$, and a sequence of $M$ testing domains $\{\mathcal{D}_N, \mathcal{D}_{N+1}, \ldots, \mathcal{D}_{N+M-1}\}$. The source domains are all labeled while the testing domains are unlabeled. Starting from the first domain, the dataset is gradually shifting. We assume that the underlying domain shift is not completely random, and propose to directly learn the drift dynamics through **D**rift-**A**ware **P**redictive (DAP) coding.

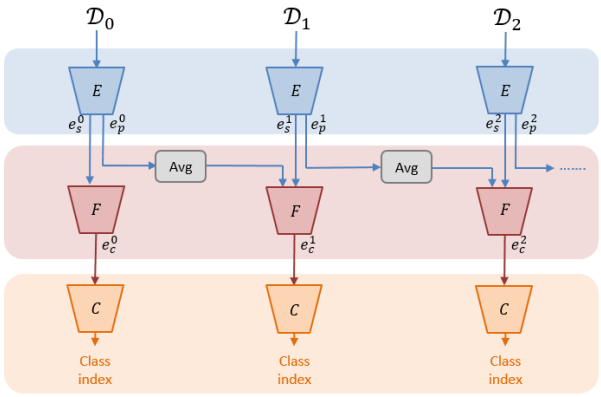


Figure 1. The general structure of Vanilla DAP, which contains three modules: Feature Extraction Module (blue), Feature Fusion module (red) and task specific module (orange). The feature extraction module extracts the domain-specific vectors $e_s$ and drift-predictive vectors $e_p$; the feature fusion module fuses the domain-specific vectors and the drift-predictive vectors from the previous domain; the task specific module defines the task we want to solve. Here we are dealing with a classification problem

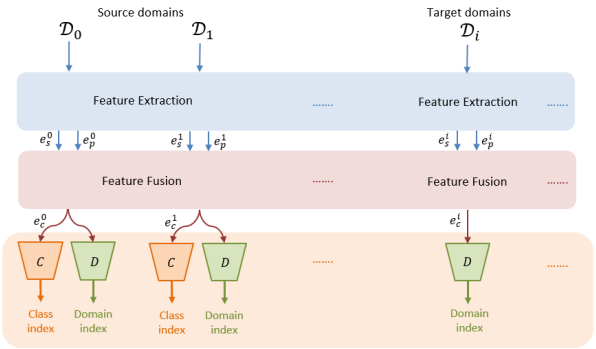


Figure 2. DAP with distribution alignment. Another discriminator (green) is added to match the fused feature vectors $e_c$ for all domains, including the test domains, while the feature extraction module and feature fusion module stay the same.

## 3.2. Drift-Aware Predictive Coding

DAP is inspired by predictive coding which is widely studied in neuroscience [15]. The key idea is to learn representations that contain predictive information about future domains. Towards this end, we propose a pipeline that allows embedding sequential domains into a sequence of embedding vectors, which is shown in Fig. 1. Similar to *Domain2Vec* [14], we train a feature extraction network $E$ that extract the domain-specific vector $e_s^{i,j}$ and drift-predictive vector $e_p^{i,j}$ for the $i$th data from $j$the domain. The domain-specific vector $e_s^{i,j}$ contains information that is useful for making accurate predictions on domain $\mathcal{D}_j$. In the meanwhile, the drift-predictive vector $e_p^{i,j}$ is predictive for the domain shift from $\mathcal{D}_j$ to $\mathcal{D}_{j+1}$. Next, we train a feature

fusion network $F$ that fuses both the domain-specific vectors from current domain $\mathcal{D}_j$ and the drift-predictive vectors from the previous domain $\mathcal{D}_{j-1}$. To represent the predictive information for the entire domain, we take the average of all drift-predictive vectors from $\mathcal{D}_{i-1}$ as the predictive embedding vector for domain $\mathcal{D}_{j-1}$. That is, $e_p^{j-1} = \frac{1}{|\mathcal{D}_j|}\sum_{i=0}^{|\mathcal{D}_{j-1}|-1} e_p^{i,j-1}$. After that, we concatenate $e_p^{j-1}$ with every domain specific vector $e_s^{i,j}$ and feed them to the feature fusion network. For domain $\mathcal{D}_0$ that doesn't have drift-predictive vectors from the previous domain, we replace it by some special hand-craft vectors. So far, we use zero vectors as $e_p^{-1}$. At the end, we pass the fused features $e_c^{i,j}$ to a classification network $C$ for the prediction.

**Drift-Aware Predictive Loss** For domain $j$, we define the Drift-Aware Predictive Loss as:

$$\mathcal{L}_{DAP}^{j} = \sum_{i=0}^{|\mathcal{D}_j|-1} \mathcal{L}_{CE}\left(C(e_c^{i,j}), y^{i,j}\right) \tag{1}$$

where $y^{i,j}$ denotes the ground truth class label for $i$th data from domain $\mathcal{D}_j$. Note that we need the ground truth labels to calculate the Drift-Aware Predictive Loss. Thus, we only have the loss for all source domains $\{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{N-1}\}$. And the total Drift-Aware Predictive Loss can be represented as their sum, i.e.,

$$\mathcal{L}_{DAP} = \sum_{j=0}^{N-1} \mathcal{L}_{DAP}^{j} \tag{2}$$

If the above loss can be successfully minimized, the drift-predictive embedding will be able to improve the next base learner before any data is collected from the shifted domain, i.e., proactively adapting to the changing environment. Since we only receive the drift-predictive vectors from the previous domain, our adaptation pipeline forms a first-order Markov chain. However, for more complicated domain shift dynamics, a higher-order might be necessary.

### 3.3. Aligning Feature Distribution with Adversarial Training

The current DAP coding is supervised to capture the gradual domain shifting within source domains. However, without any exposure to target domains, the model can only make guesses about future target domains and the feature extractor can fail on high dimensional data, since they are harder to track and predict. As a result, to improve the performance of DAP coding, we introduce an adversarial training mechanism from [18] to force the network to learn a more oriented domain shift by exposing target domain data during training.

Our aim is to align the embedding vector $e_c$ after feature extraction and fusion network from all $N+M$ source and target domains to the same distribution. We want the distribution $p(e_c|u_1) = p(e_c|u_2), \forall u_1, u_2 \in \mathcal{U}$. $\mathcal{U}$ is the set of all domain indices. This implies $p(e_c|u) = p(e_c)$ and $e_c$ and u are independent. Similar with many adversarial approaches [5][13], we train a discriminator $D$ to predict each domain while our feature extraction and fusion network aim to fool the discriminator, as shown in Figure 2

The optimization objective can be written as:

$$\min_{E,F,C} \max_{D} \mathcal{L}_{DAP} - \lambda \mathcal{L}_d(D(e_c), u), \tag{3}$$

where $\mathcal{L}_{DAP}$ is the loss for drift-aware predictive coding, u is the domain index, $e_c$ is the feature embedding before last classifier C and $\lambda$ is a weighting term. Note that $\mathcal{L}_{DAP}$ is only calculated on the labelled source domains and $\mathcal{L}_d$ is calculated on both source and target domains.

In our problem setting, the data is gradually shifting and our domain shift is continuous and ordinal. Therefore, instead of using classification, similar with [18], we use the discriminator to regress the domain index. For simplicity, we use $\mathcal{L}_2$ loss for $\mathcal{L}_d$,

$$\mathcal{L}_d(D(e_c), u) = (D(e_c) - u)^2 \tag{4}$$

### 3.4. Training Procedure

During training, in each step, we sample equal number of samples from each domain and build a chain as in Figure 1. The base learner share its weight across all domains and we modulate it with different DAP coding $e_p^j$ to adapt it to different domains. We either use single $\mathcal{L}_{DAP}$ or the adversarial objective in 3 to train the network, depending on whether data in target domains is available.

### 3.5. Adapting to Test domains

After the predictive domain embedding model is trained on the sampled sequence, we can apply it for model adaptation in the testing phase. Specifically, for each new test domain $\mathcal{D}_i, \forall i \in \{N, N+1, \ldots, N+M-1\}$, we will obtain a new base learner model $f_{\theta_i|e_p^i}$ by conditioning the initial model parameters $\theta_0$ on the drift-predictive embedding vector of the previous domain $e_p^{i-1}$.

## 4. Experiment

In this section, we experiment our Drift-Aware Predictive Domain Adaptation on three datasets, (1) Toy ellipse data and sine data (2) Rotated MNIST data and (3) American High School Portraits data(Yearbook)[7]. We show that our model with drift-aware predictive coding can still perform well during gradual domain shifting and the decision boundaries can gradually adapting according to the change of data distribution.

### 4.1. Synthesized Toy Data

**Toy ellipse data** The toy ellipse data consists of 40 domains indexed from 1 to 40, shifting from left to right as in Figure 5 first column. In the first three groups of data, the distribution of left half(1st-20th domains) of the data are kept the same. Data are distributed normally along the boundary of a $\frac{1}{4}$ circle with radius 5. While on the right half(21st-40th domain), the data are distributed on a $\frac{1}{4}$ ellipse with fixed short axis of 5 and varying long axis $\{5,10,15\}$. We train on 10 domains and test on the remaining 30 domains. Note these three experiments have training data with the same distribution in source domains while they have different data in the target domains. We use only $\mathcal{L}_{DAP}$ for training and data in target domains is not visible during training. Figure 5 shows the predictions of our methods and predictions of a simple baseline without adaptation. Figure 7 shows the decision boundary of these three models under 1st,15th and 30th target domains. From these results, we show that DAP coding can capture and forecast the future trend of data even though data distributions in target domains are kept the same. DAP coding can adjust the decision boundaries during gradual domain shifting and can have dynamic decision boundaries subject to the actual domain shift.

**Toy sine data** The toy sine data also contains 40 domains and data points are normally distributed along a sine curve with three cycles. All the settings are the same with toy ellipse data except 15 domains are used for training. Last two rows in Figure 5 shows the predictions of DAP model on toy sine data and its dynamically changing decision boundaries.

### 4.2. Rotated MNIST

We also evaluate our method on the MNIST handwritten digits dataset. The dataset contains 60,000 images. Each image is 28 x 28 pixels. We rotate these digits by a certain angle and split the dataset into a total of 20 domains divided between training and testing. We take images rotated between $0°$ and $15°$ to be our training source domains and images between $15°$ and $60°$ for target domains. Note that each image is seen at exactly one angle, so the training procedure cannot track a single image across different angles. Examples of the data from the Rotated MNIST dataset is shown in Figure 3.

### 4.3. Yearbook Portraits

We additionally study our method on the publicly available American High School Yearbooks dataset. The dataset contains 37,921 front-facing American high school yearbook photos over 120 years. Like the gradual domain adaption paper, we use the first 2000 images (1905-1935) as the source, the next 14,000 (1935-1969) as the intermediate domains, and the next 2000 images as the target (1969 - 1973). The goal of our model is to classify the gender of the student. Examples of the data from the Yearbook Portraits dataset are shown in Figure 3.

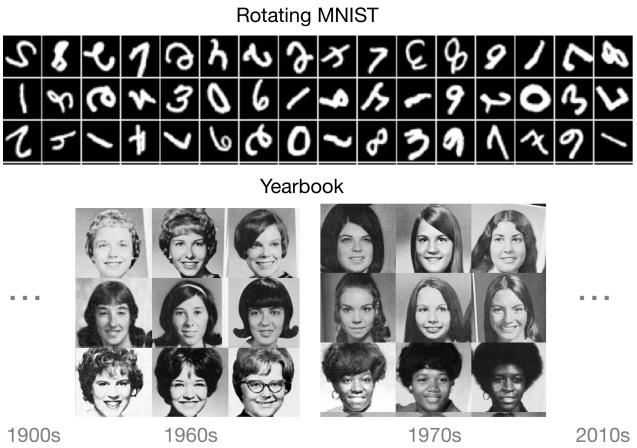


Figure 3. Examples of data from Yearbook[6] and Rotated MNIST.

## 5. Results

### 5.1. Gradual domain adaptation

We took Gradual Domain Adaptation(GDA) [12] as one of the baselines. We follow the dataset split introduced in section.4. We tested it on two datasets, Yearbook and Rotated MNIST. In our proposed method, we take at least two domains as the source domains so that the model can learn the "trending"; however, in GDA, it only needs one as source domain and then it can iteratively train (in an unsupervised way) on the following successive domains. So for a fair comparison, we group the same number of domains as we used for training our model into one source domain to test the GDA method. Figure.4 shows our method outperforms the baseline by a large margin, which further proves the efficacy of our method.

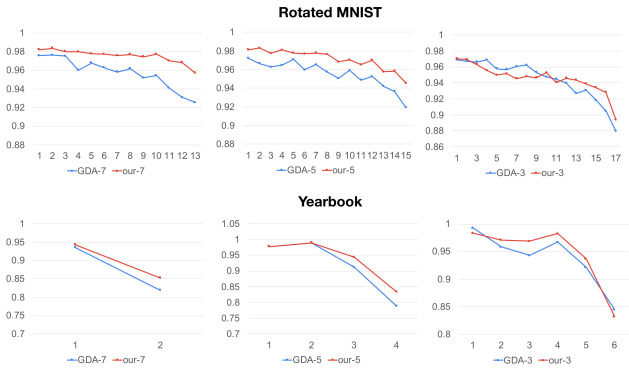


Figure 4. Comparison between Gradual Domain Adaptation(GDA) and our method when training includes different number of domains.

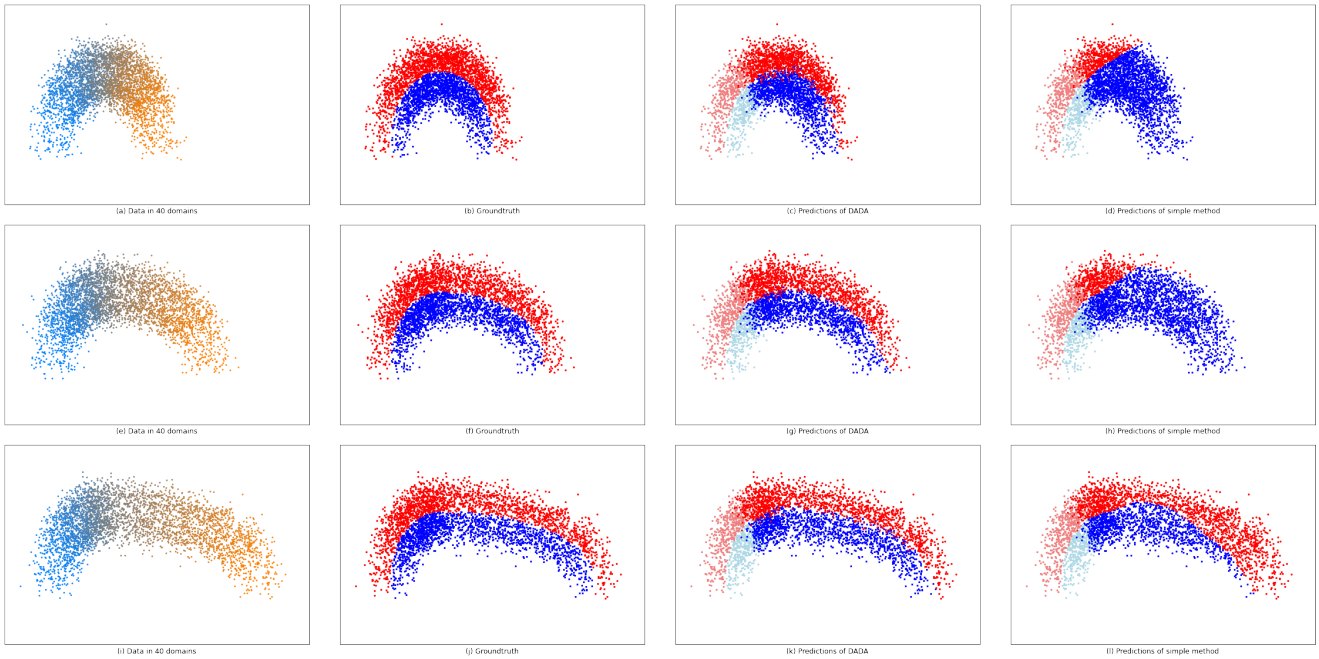

Figure 5. Results from ablation study on varying the radius of toy ellipse data. The radius values are $\{5, 10, 15\}$ respectively for each row. The first column is the data. Blue is the source domain and orange is the target domain. The second column shows the ground truth classifications and decision boundary. The third column is the predicted decision boundary using our proposed method. The final column shows the results without using the predictive and stationary embedding vectors and simply training on the source domain.

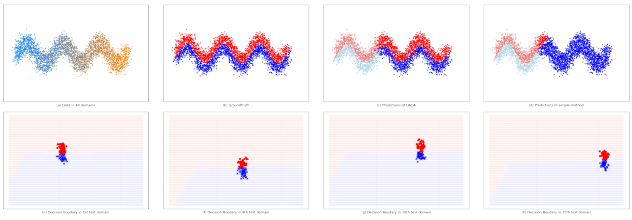

Figure 6. Results on the toy sine data. The first row shows the data, groundtruth and predictions. The second row shows the decision boundary in several different domains

## 5.2. Ablation Studies

We provide a series of ablation studies to understand the effects of several variables in our method including the number of training domains and the batch size of each domain.

Intuitively, we expect that using a larger batch size from each domain for training will result in higher accuracy and similarly for using more training domains. We report in Table 1 the average accuracies over our test domains as well as the accuracy of the last test domain for the Rotated-MNIST dataset. We consistently achieve over $96\%$ test accuracy.

We also studied the effect of varying the number of domains used for training and find as expected that performance improves with more training domains. Recall there are a total of 20 domains for the Rotated-MNIST dataset. We record our results in Table 2. The trends we observe from our ablation agrees with our intuition that performance should increase with higher batch size and more training domains.

| Batch size | Average test acc (%) | Last domain acc (%) |
|---|---|---|
| 64 | 0.9627 | 0.9256 |
| 128 | **0.9693** | 0.9347 |
| 256 | 0.9679 | **0.9397** |

Table 1. Ablation over batch size

| # | Average test acc (%) | Last domain acc (%) |
|---|---|---|
| 1 | 0.6935 | 0.3223 |
| 3 | 0.9433 | 0.915 |
| 5 | 0.9711 | 0.9453 |
| 7 | **0.9753** | **0.957** |

Table 2. Ablation over the number of training domains

## 5.3. Future Works

For our final report, we aim to conduct more extensive analysis of our methodology and implement more baselines to further evaluate the benefits of our approach. One paper we would like to investigate further is the Conditional Adversarial Domain Adaptation paper. They propose a frame-

Figure 7. Decision boundaries at different domains along the ellipse boundary

work that conditions the adversarial domain adaptation on discriminative information to enable alignment of multi-modal distributions. Another relevant benchmark is the CyCADA paper which uses a cycle-consistency loss to adapt representations at both the pixel-level and feature-level. We will adapt these methods into our continuously indexed domain setup and compare their Additionally, we would like to apply our method onto more challenging visual datasets such as the OfficeHome dataset and the VisDA-2017 sim-to-real dataset. We also plan to conduct further ablations to study each component of our architecture and understand it's effect on the model performance. For example, we will try different strategies for sampling data (random, gaussian, uniform, etc) and see whether the sampling process has an effect on learning the domain shift. Lastly, we'd like to investigate further into some of our current failure cases. Specifically, we find that our model is not able to fully capture the optimal decision boundary for the sine curve dataset which should be the tangent to the curve. We'd like to further explore these scenarios and address them for the final report.

## 6. Conclusion

In this project, we investigate the problem of domain adaptation in a gradual shifting environment. We propose to learn the shifting dynamics through drift-aware predictive coding and our work shows that DAP coding is informative and can be used to adapt decision boundaries to each domain and further improve the performance. We quantitative evaluate our approach against several datasets and we compare it against a strong state-of-the-art baseline, gradual domain adaptation. Future work can investigate different kinds of domain shifts and more experiments can be conducted on real dataset.

## References


[1] Adeleh Bitarafan, Mahdieh Soleymani Baghshah, and Marzieh Gheisari. Incremental evolving domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):2128–2141, 2016. 2

[2] Andreea Bobu, Eric Tzeng, Judy Hoffman, and Trevor Darrell. Adapting to continuously shifting domains. In *International Conference on Learning Representations Workshop*, 2018. 1

[3] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks, 2017. 2

[4] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks, 2016. 1, 2

[5] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016. 3

[6] Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, and Alexei A Efros. A century of portraits: A visual historical record of american high school yearbooks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1–7, 2015. 4

[7] S. Ginosar, K. Rakelly, S. M. Sachs, B. Yin, C. Lee, P. Krähenbühl, and A. A. Efros. A century of portraits: A visual historical record of american high school yearbooks. *IEEE Transactions on Computational Imaging*, 3(3):421–431, 2017. 1, 3

[8] Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous manifold based adaptation for evolving visual domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 867–874, 2014. 2

[9] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. PMLR, 2018. 1

[10] Guillermo Ortiz Jimenez, Mireille El Gheche, Effrosyni Simou, Hermina Petric Maretic, and Pascal Frossard. Cdot: Continuous domain adaptation using optimal transport. *arXiv preprint arXiv:1909.11448*, 2019. 2

[11] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. 1

[12] Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. *arXiv preprint arXiv:2002.11361*, 2020. 1, 2, 4

[13] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1640–1650, 2018. 1, 3

[14] Xingchao Peng, Yichen Li, and Kate Saenko. Domain2vec: Domain embedding for unsupervised domain adaptation. In *European conference on computer vision*, 2020. 2

[15] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87, 1999. 2

[16] Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018. 1

[17] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation, 2017. 2

[18] Hao Wang, Hao He, and Dina Katabi. Continuously indexed domain adaptation. In *International Conference of Machine Learning*, 2020. 3

[19] Markus Wulfmeier, Alex Bewley, and Ingmar Posner. Incremental adversarial domain adaptation for continually changing environments. In *2018 IEEE International conference on robotics and automation (ICRA)*, pages 1–9. IEEE, 2018. 2

[20] Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoffrey J Gordon. On learning invariant representation for domain adaptation. *arXiv preprint arXiv:1901.09453*, 2019. 1