

# Deep CORAL: Correlation Alignment for Deep Domain Adaptation

Baochen Sun\* and Kate Saenko\*\*

University of Massachusetts Lowell, Boston University

**Abstract.** Deep neural networks are able to learn powerful representations from large quantities of labeled input data, however they cannot always generalize well across changes in input distributions. Domain adaptation algorithms have been proposed to compensate for the degradation in performance due to domain shift. In this paper, we address the case when the target domain is unlabeled, requiring unsupervised adaptation. CORAL[1] is a “frustratingly easy” unsupervised domain adaptation method that aligns the second-order statistics of the source and target distributions with a linear transformation. Here, we extend CORAL to learn a nonlinear transformation that aligns correlations of layer activations in deep neural networks (Deep CORAL). Experiments on standard benchmark datasets show state-of-the-art performance.

## 1 Introduction

Many machine learning algorithms assume that the training and test data are independent and identically distributed (i.i.d.). However, this assumption rarely holds in practice as the data is likely to change over time and space. Even though state-of-the-art Deep Convolutional Neural Network features are invariant to low level cues to some degree [2,3,4], Donahue et al. [5] showed that they still are susceptible to domain shift. Instead of collecting labelled data and training a new classifier for every possible scenario, unsupervised domain adaptation methods [6,7,8,9,10,1] try to compensate for the degradation in performance by transferring knowledge from labelled source domains to unlabelled target domains. A recently proposed CORAL method [1] aligns the second-order statistics of the source and target distributions with a linear transformation. Even though it is “frustratingly easy”, it works well for unsupervised domain adaptation. However, it relies on a linear transformation and is not end-to-end: it needs to first extract features, apply the transformation, and then train an SVM classifier in a separate step.

In this work, we extend CORAL to incorporate it directly into deep networks by constructing a differentiable loss function that minimizes the difference between source and target correlations—the CORAL loss. Compared to CORAL, our proposed Deep CORAL approach learns a non-linear transformation that

\* bsun@cs.uml.edu

\*\* saenko@bu.edu

is more powerful and also works seamlessly with deep CNNs. We evaluate our method on standard benchmark datasets and show state-of-the-art performance.

## 2 Related Work

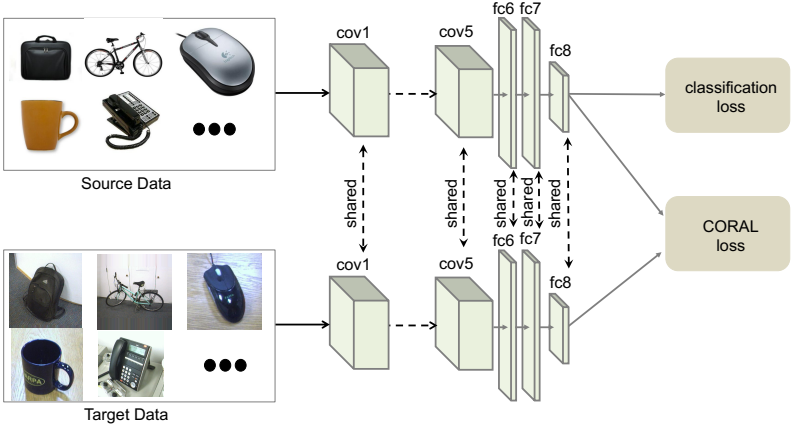
Previous techniques for unsupervised adaptation consisted of re-weighting the training point losses to more closely reflect those in the test distribution [11,12] or finding a transformation in a lower-dimensional manifold that brings the source and target subspaces closer together. Re-weighting based approaches often assume a restricted form of domain shift—selection bias—and are thus not applicable to more general scenarios. Geodesic methods [13,7] bridge the source and target domain by projecting source and target onto points along a geodesic path [13], or finding a closed-form linear map that transforms source points to target [7]. [14,8] align the subspaces by computing the linear map that minimizes the Frobenius norm of the difference between the top  $n$  eigenvectors. In contrast, CORAL [1] minimizes domain shift by aligning the second-order statistics of source and target distributions.

Adaptive deep neural networks have recently been explored for unsupervised adaptation. DLID [15] trains a joint source and target CNN architecture with two adaptation layers. DDC [16] applies a single linear kernel to one layer to minimize Maximum Mean Discrepancy (MMD) while DAN [17] minimizes MMD with multiple kernels applied to multiple layers. ReverseGrad [18] adds a binary classifier to explicitly confuse the two domains.

Our proposed Deep CORAL approach is similar to DDC, DAN, and ReverseGrad in the sense that a new loss (CORAL loss) is added to minimize the difference in learned feature covariances across domains, which is similar to minimizing MMD with a polynomial kernel. However, it is more powerful than DDC (which aligns sample means only), much simpler to optimize than DAN and ReverseGrad, and can be integrated into different layers or architectures seamlessly.

## 3 Deep CORAL

We address the unsupervised domain adaptation scenario where there are no labelled training data in the target domain, and propose to leverage both the deep features pre-trained on a large generic domain (such as Imagenet [19]) and the labelled source data. In the meantime, we also want the final learned features to work well on the target domain. The first goal can be achieved by initializing the network parameters from the generic pre-trained network and fine-tuning it on the labelled source data. For the second goal, we propose to minimize the difference in second-order statistics between the source and target feature activations, i.e. the CORAL loss. Figure 1 shows a sample Deep CORAL architecture using our proposed correlation alignment layer for deep domain adaptation. We refer to Deep CORAL as any deep network incorporating the CORAL loss for domain adaptation.



**Fig. 1.** Sample Deep CORAL architecture based on a CNN with a classifier layer. For generalization and simplicity, here we apply the CORAL loss to the  $fc8$  layer of AlexNet [20]. Integrating it to other layers or network architectures should be straightforward.

### 3.1 CORAL Loss

We first describe the CORAL loss between two domains for a single feature layer. Suppose we are given source-domain training examples  $D_S = \{\mathbf{x}_i\}$ ,  $\mathbf{x} \in \mathbb{R}^d$  with labels  $L_S = \{y_i\}$ ,  $i \in \{1, \dots, L\}$ , and unlabeled target data  $D_T = \{\mathbf{u}_i\}$ ,  $\mathbf{u} \in \mathbb{R}^d$ . Suppose the number of source and target data are  $n_S$  and  $n_T$  respectively. Here both  $\mathbf{x}$  and  $\mathbf{u}$  are the  $d$ -dimensional deep layer activations  $\phi(I)$  of input  $I$  that we are trying to learn. Suppose  $D_S^{ij}$  ( $D_T^{ij}$ ) indicates the  $j$ -th dimension of the  $i$ -th source (target) data example and  $C_S$  ( $C_T$ ) denote the feature covariance matrices.

We define the CORAL loss as the distance between the second-order statistics (covariances) of the source and target features:

$$\ell_{CORAL} = \frac{1}{4d^2} \|C_S - C_T\|_F^2 \quad (1)$$

where  $\|\cdot\|_F^2$  denotes the squared matrix Frobenius norm. The covariance matrices of the source and target data are given by:

$$C_S = \frac{1}{n_S - 1} (D_S^\top D_S - \frac{1}{n_S} (\mathbf{1}^\top D_S)^\top (\mathbf{1}^\top D_S)) \quad (2)$$

$$C_T = \frac{1}{n_T - 1} (D_T^\top D_T - \frac{1}{n_T} (\mathbf{1}^\top D_T)^\top (\mathbf{1}^\top D_T)) \quad (3)$$

where  $\mathbf{1}$  is a column vector with all elements equal to 1.

The gradient with respect to the input features can be calculated using the chain rule:

$$\frac{\partial \ell_{CORAL}}{\partial D_S^{ij}} = \frac{1}{d^2(n_S - 1)} ((D_S^\top - \frac{1}{n_S} (\mathbf{1}^\top D_S)^\top \mathbf{1}^\top)^\top (C_S - C_T))^{ij} \quad (4)$$

$$\frac{\partial \ell_{CORAL}}{\partial D_T^{ij}} = -\frac{1}{d^2(n_T - 1)} \left( (D_T^\top - \frac{1}{n_T} (\mathbf{1}^\top D_T)^\top \mathbf{1}^\top)^\top (C_S - C_T) \right)^{ij} \quad (5)$$

We use batch covariances and the network parameters are shared between two networks.

### 3.2 End-to-end Domain Adaptation with CORAL Loss

We describe our method by taking a multi-class classification problem as the running example. As mentioned before, the final deep features need to be both **discriminative enough to train strong classifier** and **invariant to the difference between source and target domains**. Minimizing the classification loss itself is likely to lead to overfitting to the source domain, causing reduced performance on the target domain. On the other hand, minimizing the CORAL loss alone might lead to degenerated features. For example, the network could project all of the source and target data to a single point, making the CORAL loss trivially zero. However, no strong classifier can be constructed on these features. Joint training with both the classification loss and CORAL loss is likely to learn features that work well on the target domain:

$$\ell = \ell_{CLASS} + \sum_{i=1}^t \lambda_i \ell_{CORAL} \quad (6)$$

where  $t$  denotes the number of CORAL loss layers in a deep network and  $\lambda$  is a weight that trades off the adaptation with classification accuracy on the source domain. As we show below, these two losses play counterparts and reach an *equilibrium* at the end of training, where the final features are expected to work well on the target domain.

## 4 Experiments

We evaluate our method on a standard domain adaptation benchmark – the Office dataset [6]. The Office dataset contains 31 object categories from an office environment in 3 image domains: *Amazon*, *DSLR*, and *Webcam*.

We follow the standard protocol of [7,17,5,16,18] and use all the labelled source data and all the target data without labels. Since there are 3 domains, we conduct experiments on all 6 shifts, taking one domain as the source and another as the target.

In this experiment, we apply the CORAL loss to the last classification layer as it is the most general case—most deep classifier architectures (e.g., convolutional, recurrent) contain a fully connected layer for classification. Applying the CORAL loss to other layers or other network architectures should be straightforward.

The dimension of last fully connected layer (*fc8*) was set to the number of categories (31) and initialized with  $\mathcal{N}(0, 0.005)$ . The learning rate of *fc8* is set to 10 times the other layers as it was training from scratch. We initialized the other layers with the parameters pre-trained on ImageNet [19] and kept the original

layer-wise parameter settings. In the training phase, we set the batch size to 128, base learning rate to  $10^{-3}$ , weight decay to  $5 \times 10^{-4}$ , and momentum to 0.9. The weight of the CORAL loss ( $\lambda$ ) is set in such way that at the end of training the classification loss and CORAL loss are roughly the same. It seems be a reasonable choice as we want to have a feature representation that is both discriminative and also minimizes the distance between the source and target domains. We used Caffe [21] and BVLC Reference CaffeNet for all of our experiments.

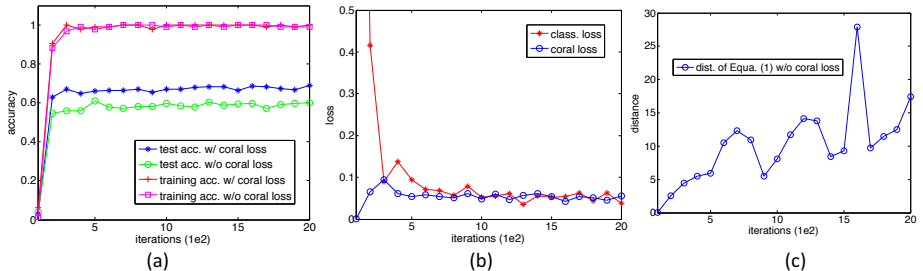
We compare to 7 recently published methods: CNN [20] (no adaptation), GFK [7], SA [8], TCA [22], CORAL [1], DDC [16], DAN [17]. GFK, SA, and TCA are manifold based methods that project the source and target distributions into a lower-dimensional manifold and are not end-to-end deep methods. DDC adds a domain confusion loss to AlexNet and fine-tunes it on both the source and target domain. DAN is similar to DDC but utilizes a multi-kernel selection method for better mean embedding matching and adapts in multiple layers. For direct comparison, DAN in this paper uses the hidden layer *fc8*. For GFK, SA, TCA, and CORAL, we use the *fc7* feature fine-tuned on the source domain (*FT7* in [1]) as it achieves better performance than generic pre-trained features, and train a linear SVM [8,1]. To have a fair comparison, we use accuracies reported by other authors with exactly the same setting or conduct experiments using the source code provided by the authors.

From Table 1 we can see that Deep CORAL (D-CORAL) achieves better average performance than CORAL and the other 6 baseline methods. In three 3 out of 6 shifts, it achieves the highest accuracy. For the other 3 shifts, the margin between D-CORAL and the best baseline method is very small ( $\leq 0.7$ ).

	A→D	A→W	D→A	D→W	W→A	W→D	AVG
GFK	52.4±0.0	54.7±0.0	43.2±0.0	92.1±0.0	41.8±0.0	96.2±0.0	63.4
SA	50.6±0.0	47.4±0.0	39.5±0.0	89.1±0.0	37.6±0.0	93.8±0.0	59.7
TCA	46.8±0.0	45.5±0.0	36.4±0.0	81.1±0.0	39.5±0.0	92.2±0.0	56.9
CORAL	65.7±0.0	64.3±0.0	48.5±0.0	<b>96.1±0.0</b>	48.2±0.0	<b>99.8±0.0</b>	70.4
CNN	63.8±0.5	61.6±0.5	51.1±0.6	95.4±0.3	49.8±0.4	99.0±0.2	70.1
DDC	64.4±0.3	61.8±0.4	52.1±0.8	95.0±0.5	<b>52.2±0.4</b>	98.5±0.4	70.6
DAN	65.8±0.4	63.8±0.4	<b>52.8±0.4</b>	94.6±0.5	51.9±0.5	98.8±0.6	71.3
D-CORAL	<b>66.8±0.6</b>	<b>66.4±0.4</b>	<b>52.8±0.2</b>	95.7±0.3	51.5±0.3	99.2±0.1	<b>72.1</b>

**Table 1.** Object recognition accuracies for all 6 domain shifts on the standard Office dataset with deep features, following the standard unsupervised adaptation protocol.

To get a better understanding of Deep CORAL, we generate three plots for domain shift A→W. In Figure 2(a) we show the training (source) and testing (target) accuracies for training with vs. without CORAL loss. We can clearly see that adding the CORAL loss helps achieve much better performance on the target domain while maintaining strong classification accuracy on the source domain.



**Fig. 2.** Detailed analysis of shift A→W for training w/ vs. w/o CORAL loss. (a): training and test accuracies for training w/ vs. w/o CORAL loss. We can see that adding CORAL loss helps achieve much better performance on the target domain while maintaining strong classification accuracy on the source domain. (b): classification loss and CORAL loss for training w/ CORAL loss. As the last fully connected layer is randomly initialized with  $\mathcal{N}(0, 0.005)$ , CORAL loss is very small while classification loss is very large at the beginning. After training for a few hundred iterations, these two losses are about the same. (c): CORAL distance for training w/o CORAL loss (setting the weight to 0). The distance is getting much larger ( $\geq 100$  times larger compared to training w/ CORAL loss).

In Figure 2(b) we visualize both the classification loss and the CORAL loss for training w/ CORAL loss. As the last fully connected layer is randomly initialized with  $\mathcal{N}(0, 0.005)$ , in the beginning the CORAL loss is very small while the classification loss is very large. After training for a few hundred iterations, these two losses are about the same. In Figure 2(c) we show the CORAL distance between the domains for training w/o CORAL loss (setting the weight to 0). We can see that the distance is getting much larger ( $\geq 100$  times larger compared to training w/ CORAL loss). Comparing Figure 2(b) and Figure 2(c), we can see that even though the CORAL loss is not always decreasing during training, if we set its weight to 0, the distance between source and target domains becomes much larger. This is reasonable as fine-tuning without domain adaptation is likely to overfit the features to the source domain. Our CORAL loss constrains the distance between source and target domain during the fine-tuning process and helps to maintain an *equilibrium* where the final features work well on the target domain.

## 5 Conclusion

In this work, we extended CORAL, a simple yet effective unsupervised domain adaptation method, to perform end-to-end adaptation in deep neural networks. Experiments on standard benchmark datasets show state-of-the-art performance. Deep CORAL works seamlessly with deep networks and can be easily integrated into different layers or network architectures.

## References

1. Sun, B., Feng, J., Saenko, K.: Return of frustratingly easy domain adaptation. In: AAAI. (2016)
2. Peng, X., Sun, B., Ali, K., Saenko, K.: What do deep cnns learn about objects? In: ICLR Workshop Track. (2015)
3. Peng, X., Sun, B., Ali, K., Saenko, K.: Learning deep object detectors from 3d models. In: ICCV. (2015)
4. Sun, B., Peng, X., Saenko, K.: Generating large scale image datasets from 3d cad models. In: CVPR'15 Workshop on The Future of Datasets in Vision. (2015)
5. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: ICML. (2014)
6. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: ECCV. (2010)
7. Gong, B., Shi, Y., Sha, F., Grauman, K.: Geodesic flow kernel for unsupervised domain adaptation. In: CVPR. (2012)
8. Fernando, B., Habrard, A., Sebban, M., Tuytelaars, T.: Unsupervised visual domain adaptation using subspace alignment. In: ICCV. (2013)
9. Sun, B., Saenko, K.: From virtual to reality: Fast adaptation of virtual object detectors to real domains. In: BMVC. (2014)
10. Sun, B., Saenko, K.: Subspace distribution alignment for unsupervised domain adaptation. In: BMVC. (2015)
11. Jiang, J., Zhai, C.: Instance Weighting for Domain Adaptation in NLP. In: ACL. (2007)
12. Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., Schölkopf, B.: Correcting sample selection bias by unlabeled data. In: NIPS. (2006)
13. Gopalan, R., Li, R., Chellappa, R.: Domain adaptation for object recognition: An unsupervised approach. In: ICCV. (2011)
14. Harel, M., Mannor, S.: Learning from multiple outlooks. In: ICML. (2011)
15. Chopra, S., Balakrishnan, S., Gopalan, R.: Dlid: Deep learning for domain adaptation by interpolating between domains. In: ICML Workshop. (2013)
16. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance. CoRR **abs/1412.3474** (2014)
17. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. In: ICML. (2015)
18. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: ICML. (2015)
19. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. (2009)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012)
21. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
22. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. In: IJCAI. (2009)