Distilling

shuang ao

August 9, 2016

1 Background

Distillation for domain adaptation [3] and *privileged information* [9] are two techniques that enable machines to learn from other machines. Both methods address the problem how to build a student model that can learn from the advanced teacher models. Recently, Lopez *et al.* [5] proposed a framework called *generalized distillation* that unifies both techniques and show that it can be applied in many scenarios.

1.1 Privileged information

In the original work of the privileged information [9], the data can be represented as a collection of the triples:

$$\{(x_1, x_1^*, y_1), (x_2, x_2^*, y_2) \ldots (x_n, x_n^*, y_n)\}$$

where each (x_i, y_i) is a feature-label pair, and the novel element x_i is additional information about the example (x_i, y_i) provided by an intelligent teacher, such as to support the learning process. However, in the testing procedure, the learning machine is not able to obtain the privileged information from the teacher. Vapnik?s learning using privileged information is one example of what we call machines-teaching machines: the paradigm where machines learn from other machines, in addition to training data.

1.2 Distillation

Distillation uses a simple (student) machine learns a complex task by imitating the solution of a flexible (teacher) machine. For a c-class scenario, distillation works as follow: consider the data

$$\{x_i, y_i\}_{i=1}^n, \qquad x \in R^d, y \in \Delta^c$$

where Δ^c is a c-dimensional probability vector. In traditional learning, we try to learn a function f_t that:

$$f_t = \arg\min_{f_t \in \mathcal{F}_t} \frac{1}{n} \sum_{i=1}^{n} \ell\left(y_i, \sigma(f_t(x_i))\right) + \Omega(||f_t||) \tag{1}$$

here σ is the softmax operation:

$$\sigma(z)_k = \frac{e^{z_k}}{\sum_{j=1}^{c} e^{z_j}}$$

and ℓ is the cross-entropy loss function:

$$\ell(y, \hat{y}) = -\sum_{i=1}^{c} y_i \log \hat{y}_i$$

In distillation, we try to learn the student machine f_s to imitate the teacher machine f_t. During the learning process, f_s can receive the soft label s_i from the teacher machine f_t for each training example x_i.

$$s_i = \sigma(f_t(x_i)/T) \tag{2}$$

where T is the temperature for distillation. It is worthy to note that in distillation, f_t can either be a single large complex neural network or an ensemble of some complex classifiers. We can see that the soft label s_i, which reveals the class dependency, is more informative than the class label y_i.

The student machine can be learned by:

$$f_s = \underset{f_s \in \mathcal{F}_s}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \left[\lambda \ell \left(y_i, \sigma(f_s(x_i)) \right) + (1-\lambda) \ell \left(s_i, \sigma(f_s(x_i)) \right) \right] \tag{3}$$

here, $\mathcal{F}_s$ is a simpler function class than $\mathcal{F}_t$ and λ is the imitation parameter to balance the importance between the hard label y_i and the soft label s_i. The temperature $T > 0$ controls how do we want to smooth the probability from the teacher f_t. Higher temperature leads to softer label.

1.3 Generalized distillation

Generalized distillation (GD) tries to unify the two frameworks together while using the soft label as the privileged information. The process of generalized distillation is as follows:

1. Learn teacher f_t using the input-output pairs $\{x_i^*, y_i\}_{i=1}^n$ and Eq. 1.

2. Compute teacher soft labels s_i, using the temperature parameter $T > 0$.

3. Learn the student f_s using the pair $\{(x_i, y_i), (x_i, s_i)\}_{i=1}^n$ and imitation parameter λ.

2 Domain adaptation with distilling SVMs

GD can be used in many scenario such as multi-task learning, semi-supervised learning and reinforcement learning. As generalized distillation only required for the training inputs $\{x_i, y_i\}_{i=1}^n$, a teacher function f_t, it is obvious that it can be used for domain adaptation, called *Generalized Distillation Domain Adaptation* (**GDDA**), where the a source model can be used as the teacher to output the soft labels and the student model can be used as the target model.

There are several advantages for applying generalized distillation for domain adaptation:

1. Compatible with **various of scenarios**. GD only requires the outputs of the teacher model instead of asking for the specific parameters of the teacher model. Therefore, we can treat the teacher model, either a single model (single source) or an ensemble of models (multi-source), as a black-box. This means GDDA can be compatible with almost any types of source model.

2. Effective in **small data regimes**. The theoretical analysis shows that the teacher is most helpful when working with small datasets, or in the initial stages of online learning. In domain adaptation, we also requires to learn a effective classifier with less training data.

3. Compatible with **unlabeled data**. GD can be used in a semi-supervised scenario. Similarly, GDDA can work with unlabeled data to get improved performance.

4. effective in practice. According to GD, the student model can be a much simple function than the teacher model. Thus, it can be learned and deployed effectively in practice.

In GD, we have to decide the value of 2 parameters, the temperature T and imitation parameter λ. the temperature T control the smoothness of the soft label and the imitation parameter λ balance the losses from soft and hard labels. It is clear that the imitation parameter λ is more important for the performance of the student model. In the GD, this parameter is determined by brute force search which greatly reduces the effectiveness of the GDDA for a multi-source scenario (multiple λ to be determined).

To solve this problem, we propose a novel method that can determined the imitation parameter λ autonomously, called GDDA-SVM. In our GDDA-SVM, instead of using cross-entropy loss, we use Mean Squared Error (MSE) as our loss function for the following two reasons: (1) Some recently work [1] [6] [7] [8] show that MSE is also an efficient measurement for a student model to mimic the behavior of the teacher model. (2) MSE can provide a closed form cross-validation error estimation of the model for model selection.

2.1 Multi-distill

Suppose we have L examples from and N classes. There are $M-1$ the source (teacher) models providing the soft labels for each of the L examples. label matrix: $S = R^{L\times M\times N}$. For the nth binary SVM

$$\begin{aligned} \min \quad & \frac{1}{2}|w_n|^2 + C\sum_{i,j}\lambda_j e_{ijn}^2 \\ s.t. \quad & e_{ijn} = s_{ijn} - w_n x_i \\ & \sum_j \lambda_j = 1 \\ & \lambda \in [0,1]; i \in L; j \in M \end{aligned} \tag{4}$$

Lagrangian:

$$\mathcal{L} = \frac{1}{2}|w_n|^2 + C\sum_{i,j}\lambda_j e_{ijn}^2 + \sum_{i,j}\alpha_{ij}^{(n)}\left(s_{ij} - wx_i - e_{ij}\right) + \beta^{(n)}\left(\sum_j \lambda_j - 1\right) \tag{5}$$

$$\begin{aligned} \frac{\partial L}{\partial w_n} &= w_n - \sum_i \alpha_{ij}^{(n)} x_i = 0 \to w_n = \sum_i \alpha_{ij}^{(n)} x_i \\ \frac{\partial L}{\partial e_{ijn}} &= 2C\lambda_j e_{ijn} - \alpha_{ij}^{(n)} = 0 \to \alpha_{ij}^{(n)} = 2C\lambda_j e_{ijn} \end{aligned} \tag{6}$$

$$\begin{aligned} x_i \sum_k \alpha_{kj}^{(n)} x_k + \frac{\alpha_{ij}^{(n)}}{2C\lambda_j} &= s_{ijn} \\ \lambda_j x_i \sum_k \alpha_{kj}^{(n)} x_k + \frac{\alpha_{ij}^{(n)}}{2C} &= \lambda_j s_{ijn} \\ x_i \sum_k \alpha_{kj}^{(n)} x_k + \sum_j \frac{\alpha_{ij}^{(n)}}{2C} &= \sum_j \lambda_j s_{ijn} \end{aligned} \tag{7}$$

Let $\eta_{in} = \sum_j \alpha_{ij}^{(n)}$, we have:

$$\begin{aligned} \sum_j \eta_{jn} x_j x_i + \frac{\eta_{in}}{2C} &= \sum_j \lambda_j s_{ijn} \\ M\eta_n &= S_n \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix} \end{aligned} \tag{8}$$

Let $\eta'_n = M^{-1} S_n \in R^{L\times M}$

$$\eta_{in} = \sum_j \lambda_j \eta'_{ijn}$$

According to LOO and the tricks above:

$$\sum_j \lambda_j s_{ijn} - \hat{Y}_{in} = \frac{\sum_j \lambda_j \eta'_{ijn}}{M_{ii}^{-1}}$$

$$\hat{Y}_{in} = \sum_j \lambda_j \left(s_{ijn} - \frac{\eta'_{ijn}}{M_{ii}^{-1}} \right)$$

2.1.1 softmax

Let:

$$\begin{aligned} \mu_{ijn} &= s_{ijn} - \frac{\eta'_{ijn}}{M_{ii}^{-1}} \\ P_{in} &= \frac{e^{\hat{Y}_{in}}}{\sum_h e^{\hat{Y}_{ih}}} \\ \frac{\partial L_c^{(i)}}{\partial \hat{Y}_{in}} &= (P_{in} - Y_{in}) \\ \frac{\partial L_c^{(i)}}{\partial \lambda_j} &= \sum_n \mu_{ijn} (P_{in} - Y_{in}) \\ \frac{\partial L_c^{(i)}}{\partial \lambda_j} &= \sum_{i,n} \mu_{ijn} (P_{in} - Y_{in}) \end{aligned} \tag{9}$$

2.1.2 Hinge

$$L_h(\lambda, i) = \max_r \left[1 - \varepsilon_{r,y_i} + \hat{Y}_{ir}(\lambda) - \hat{Y}_{iy_i}(\lambda) \right] \tag{10}$$

$$\frac{\partial L_h^{(i)}}{\partial \lambda_j} = \begin{cases} 0 & L_h^{(i)} = 0 \\ \mu_{ijr} - \mu_{ijy_i} & \text{otherwise} \end{cases} \tag{11}$$

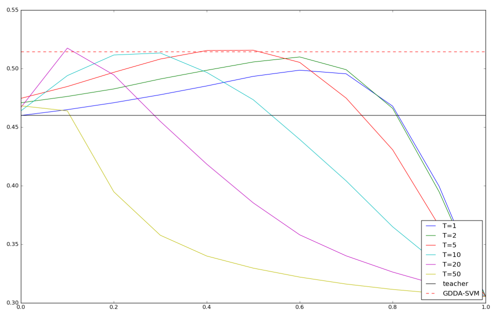
3 Experiments

3.1 Single Source for Office datasets

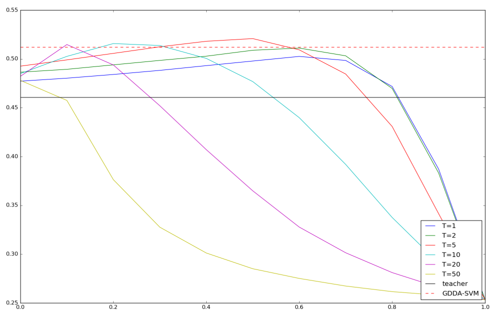
There are 3 subsets in offices datasets, webcam (795), amazon (2817) and dslr (498), sharing 31 classes. We perform 2 adaptation experiments, transferring from webcam to amazon and from dslr to amazon.

We use the features extracted from Alexnet [4] FC7 as the input features for both source and target domain. The teacher (source) model is trained with MLP on the whole source dataset and student (target) model is trained with GDDA-SVM. We also use brute force to search the imitation parameter λ in range $[0, 0.1, ..., 1]$ and show the results. Here our experiments run in a semi-supervised scenario where there are some unlabeled data in the target training set in each experiment. We also show the performance of the teacher model on the target task, denoted as "teacher". To avoid the randomness, we perform each experiments 10 times and report the results.

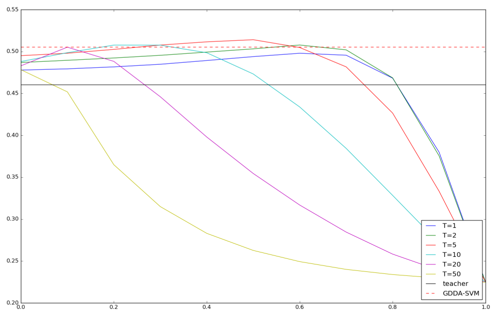
3.1.1 From DSLR to Amazon



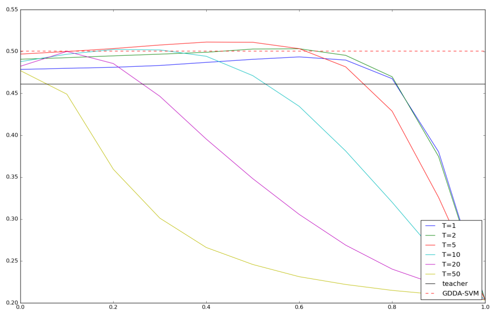
(a) 5 unlabeled



(b) 10 unlabeled



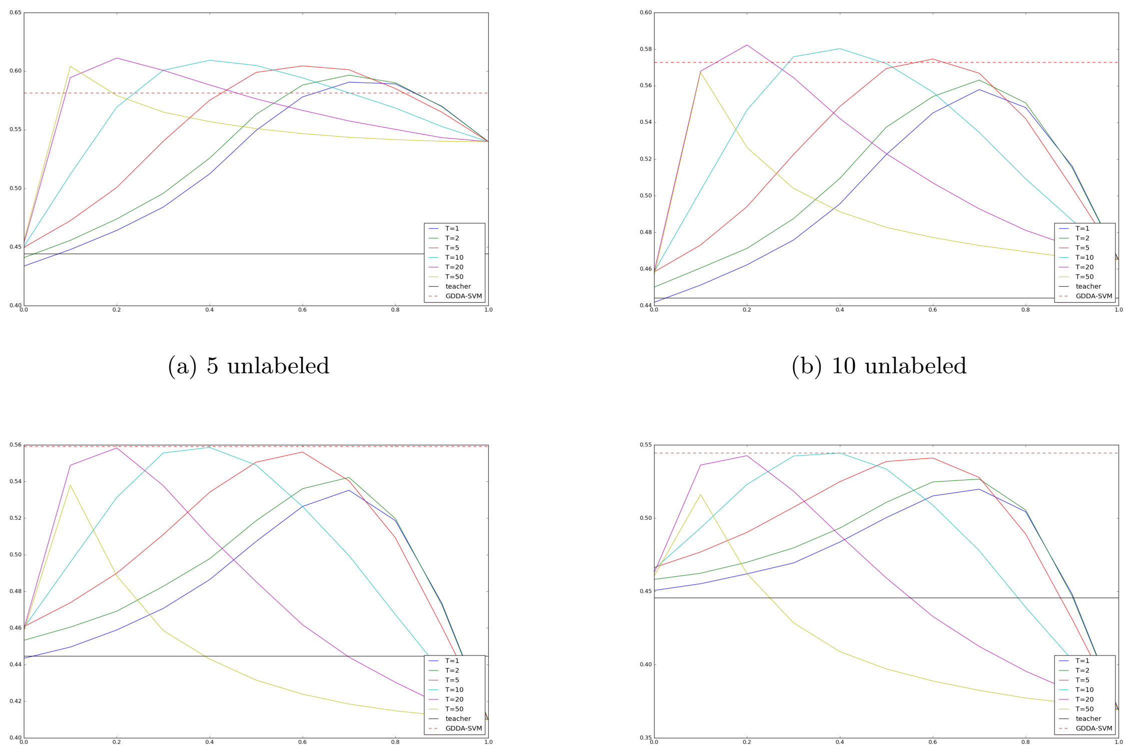
(c) 15 unlabeled



(d) 20 unlabeled

Figure 1: DSLR $\rightarrow$ Amazon. Semi-supervised adaptation with one labeled instance per class.

3.1.2 From Webcam to Amazon



(a) 5 unlabeled

(b) 10 unlabeled

(c) 15 unlabeled

(d) 20 unlabeled

Figure 2: Webcam → Amazon. Semi-supervised adaptation with one labeled instance per class.

3.2 Multi-Source for Office datasets

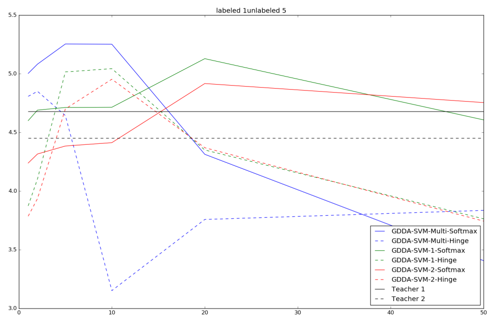
We transfer DSLR and Webcam to Amazon datasets with different settings:

4 Future works

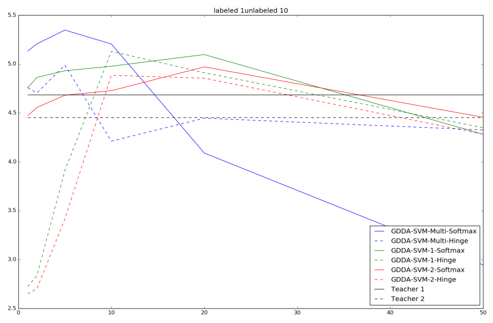
From current experiments we can see that GDDA can be a good framework for domain adaptation in single source scenario. We show that GDDA-SVM can fully exploit the knowledge from the source model. There are still a lot of work to be done within the GDDA-SVM framework:

1. If there are multiple sources and multiple imitation parameters, can GDDA-SVM provide accurate estimation?

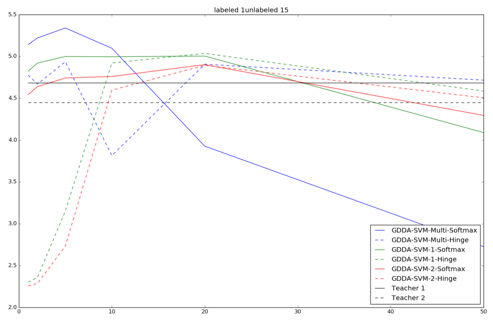
2. Can we aggregate GDDA-SVM framework into the deep neural network to train a deep GDDA network? Currently, GDDAA-SVM tries to estimate the imitation parameter by solving a convex problem. How to effectively estimate the imitation parameter when it is integrated into a GDDA network.



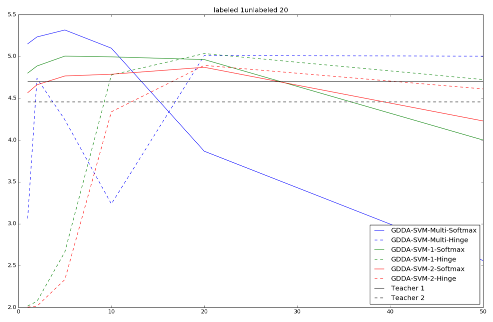
(a) 5 unlabeled



(b) 10 unlabeled



(c) 15 unlabeled



(d) 20 unlabeled

Figure 3: Webcam → Amazon. Semi-supervised adaptation with one labeled instance per class.

References

[1] J. Ba and R. Caruana. Do deep nets really need to be deep? In *Advances in neural information processing systems*, pages 2654–2662, 2014.

[2] G. C. Cawley. Leave-one-out cross-validation based model selection criteria for weighted ls-svms. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 1661–1668. IEEE, 2006.

[3] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[5] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*, 2015.

[6] P. Luo, Z. Zhu, Z. Liu, X. Wang, and X. Tang. Face model compression by distilling knowledge from neurons. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[7] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[8] G. Urban, K. J. Geras, S. E. Kahou, O. Aslan, S. Wang, R. Caruana, A. Mohamed, M. Philipose, and M. Richardson. Do deep convolutional nets really need to be deep (or even convolutional)? *arXiv preprint arXiv:1603.05691*, 2016.

[9] V. Vapnik and R. Izmailov. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research*, 16:2023–2049, 2015.