Latent Task Adaptation with Large-scale Hierarchies

Yufeng Jiang

1. Generating Query Images

Given a task, authors assume that the query images they encounter are then randomly sampled from the object calsses that belong to the given task. For each query image, they first sample the object class label from the set of possible labels that belong to the task. The conditional probability $P(y_i|h)$ follows from assuming strong sampling [2]: labels are generated uniformly at random using the corresponding parameter β_h as follows:

$$P(y_i|h) = \beta_h y_i = \begin{cases} 1/|h|, & \text{if } task\ h\ contains\ label\ y_i \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $|h|$ is the size of the number of lesf node classes in the task. The size principle plays a critical role in inferrring the latent task, as larger tasks will fenerate lower probabilities for each individual object class.

To generate an actual image x_i frm a given class label y_i, it is relatively difficult to fully model the conditional probability $P(x_i|y_i)$ to the pixel level of the images. Thus, authors use a mixed approach by having a classifier trained on all the leaf node objects, and obtain the classifier prediction

$$f(x_i) = argmax_j \quad \theta_j^T x_i, \tag{2}$$

where they simplify the notation by using x_i as both the image and the feature extracted from it, and assuming that a linear classifier with parameter $\{\theta_j\}_{j=1}^K$ is used. The conditional probability is then defined as

$$P(x_i|y_i) = C_{y_i f(x_i)} \tag{3}$$

where C is the confusion matrix of the classsifier, and C_{ij} is the probability that an image of object class i is classified as class j.

Given a set of testing images $X = \{x_1, x_2, \ldots, x_N\}$, they goal is to jointly identify the hidden task h and the hidden labels $Y = \{y_1, y_2, \ldots, y_N\}$ that maximizes the poserior probability

$$(\hat{h}, \hat{Y}) = argmax\ P(h, Y|X) \tag{4}$$

Authors will discuss in the next section how the various parameters, especially the parameters θ for the classifiers and the confusion matrix C, can be estimated from training data, and how to carry out efficient inference to find the solution to Eqn. 4

2. Efficient Learning and Inference

The probabilistic model involves multiple parameters to be estimated and nested hidden variables during the inference phase. In this section, authors present a novel approach to estimate the confusion matrix for the classifier, and a linear-time inference algorithm that jointly identifies the latent task and predictions for individual images.

Instead of these methods, authors propose to approximate its leave-one-out (LOO) error on the training data with a simple gradient descent step to "unlearn" each image to estimate its LOO prediction, similar to the early unlearning ideas [3] proposed for neural networks. They will focus on the use of multinomial logistic regression, which minimizes $\mathcal{L}(\theta) = \lambda||\theta||_2^2 - \sum_{i=1}^M t_i \log u_i$, where t_i is a 0-1 indicator vector where only the $y_i - th$ elememt is l, and u_i is the softmax of the linear outputs $u_{ij} = exp(\theta_j^T x_i)/\sum_{j'=1}^K = exp(\theta_{j'}^K x_i)$, with x_i being the feature for the i-th training image.

Specifically, given the trained classifier parameters θ, it is safe to assume that the gradient $g(\theta) = 0$. Thus, the gradient for the logistic regression loss when removing a training image x_i could be computed simply as $g_{x_i}(\theta) = (u_i - t_i)x_i^T$. Given the Hessian matrix $\mathbf{H}$ at θ, one can perform one-step quasi-Newton least-square update as [1]

$$\theta_{x_i} = \theta - \rho' H^2 g_{x_i} \tag{5}$$

Note that they put an additional step size ρ' instead of $\rho' = 1$ as would be the case for exact least squares. They set ρ' to the value that yields the same LOO approximation accuracy as the validation accuracy.

[1] In practice we used the accumulated matrix $\mathbf{H}$ obtained from Adagrad [1] as a good approximation of the Hessian matrix. See supplementary material for details.

Task	Naive	Retraining	FC	Ours
building	55.48	78.67	81.48	**82.19**
dog	35.37	39.94	42.95	**43.76**
feline	47.13	61.07	62.67	**63.54**
home app	50.78	67.30	69.26	**70.52**
vehicle	55.62	61.43	**63.41**	63.28

Table 1. Classification accuracy on given tasks (subtrees) of the whole ILSVRC data.

3. Adapting Classifiers with Known Tasks

An important question to ask is, even if they are allowed to retrain task-specific classifiers, do they want to do the retraining? Authors first analyze the benefits of retraining versus their adaptation method. To this end, they specify 5 subtrees from the ILSVRC hierarchy: building, dogs, feline, home appliance, and vehicle. They explicitly trained classifiers on these three subtrees only, and compared the retrained accuracy against their adapted classifier with the given task. They also test the naive baseline that uses the raw 1000 class predictions, and the forced choice baseline (FC) which simply selects the class under the task that has the largest output from the original classifiers. Table 1 summarizes the performance of the algorithms.

References

[1] R. Salakhutdinov, A. Torralba, and J. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR*, 2011. 1

[2] J. B. Tenenbaum and T. L. Griffiths. Generalization, similarity, and Bayesian inference. *Behavioral and Brain Sciences*, 24(4):629–640, 2001. 1

[3] F. Xu and J. B. Tenenbaum. Word learning as Bayesian inference. *Psychological Review*, 114(2):245, 2007. 1