Latent Task Adaptation with Large-scale Hierarchies

Yufeng Jiang

1. Linear Time MAP Inference

A conventional way to do probabilistic inference with nested latent variables is to use variational inference or Gibbs sampling to find a lower bound of the posterior probability. This, however, may involve multiple iterations over the hidden variables and may be slow. Authors show that when the latent task space is organized in a DAG structure, the exact MAP solution could be found with an efficient dynamic programming algorithm that has complexity linear to the number of possible tasks.

They first note that the logarithm of posterior probability could be expanded as

$$\log P(h, \mathcal{Y}|\mathcal{X}) \propto \log \alpha_h + \sum_{i=1}^{N} \log(\beta_{hy_i} C_{y_i f(x_i)}). \tag{1}$$

Notice that the size constraint defining the latent task space gives as $\beta_{hy_i} = \frac{1}{|h|} I(y_i \in h)$, the equation above could further be written as

$$\log \alpha_h - N \log |h| + \sum_{i=1}^{n} (\log C_{y_i f(x_i)} + \log I(y_i \in h)), \tag{2}$$

where one can observe that h and $\mathcal{Y}$ decouples except for the $I(y_i \in h)$ term. As the latent tasks are organized as a tree-based hierarchy, they can define auxiliary functions

$$q_i(h) = \max_{\mathrm{y}} [\log C_{y_i f(x_i)} + \log I(y_i \in h)], \tag{3}$$

which could be computed recursively as

$$q_i(h) = max_{\{h' \in child(h)\}} q_i(h'), \tag{4}$$

where $child(h)$ is the set of children of h in the tree. Finally, the latent task could be estimated as

$$\hat{h} = \arg\max_{h} [\log(\alpha_h) - N \log |h| + \gamma \sum_{i=1}^{N} q_i(h)], \tag{5}$$

and the corresponding $\hat{y}_i$s could be identified by taking the $argmax$ of the corresponding $q_i(h)$.

2. Experiments

Authors conduct their experiment on the ILSVRC 2010 dataset [1], where both validation and test data are available. For all the experiments, they learn the parameters of the model on the training and validation data, and report the performance on the test images.

They note that more comprehensive features and better classification pipelines may lead to better 1-vs-all accuracy on ImageNet, but it is not the main goal of the paper, as they focus on the adaptation on top of the base classifiers. Recent efforts on learning better classifiers, such as the ones presented in [4, 2] could be seamlessly incorporated into their learning framework for general performance increases.

The encoded features were then max pooled over 10 spatial bins: the whole image and the 3×3 regular grid. This yielded 160K feature dimensions per image, and a total of about 1.5TB for the training data in double precision format. The overall performance is 41.33% top-1 accuracy and a 61.91% top-5 accuracy on the validation data, and 41.28% and 61.69% respectively on the testing data. For the computation time, training with their toolbox took only about 24 hours with 10 commodity computers connected on a LAN. Their toolkit is implemented in Python and will be publicly available[1], and they refer to the supplementary materials for more technical details.

2.1. Estimating the Confusion Matrix

An good estimation of the confusion matrix C is crucial for the probabilistic inference. Authors evaluate the quality of different estimations using the test data: for each tesing pair $(y, \hat{y})$, where $\hat{y}$ is the classifier output, its probability is given by the confusion mateix entry $C_{y\hat{y}}$. THe perplexity measure [3] then evaluates how "surprising" the confusion matrix sees the testing data results (a smaller value indicates a better fit):

$$prep = Power(2, (\sum_{i=1}^{N_{te}} \log_2 C_{y_i \hat{y}_i})/N_{te}), \tag{6}$$

where N_{te} is the number of testing images. Overall, they obtained a perplexity of 46.27 using our unlearning algo-

[1] http://www.eecs.berkeley.edu/~jiayq/

Method	query size=5		query size=100	
	$s(h, \hat{h})$	Accuracy	$s(h, \hat{h})$	Accuracy
Naive	1.54	42.75	1.50	42.68
Proto	8.14	443.16	60.39	50.28
Hist	22.21	44.84	96.61	59.87
Hedging	39.12	44.81	50.34	51.83
Ours	**84.43**	**65.89**	**99.37**	**70.70**
Oracle	100.0	70.36	100.0	70.88

Table 1. The average task overlap score and the average accuracy for the algorithms, under query sizes 5 and 100 respectively. All numbers are in percentage. The last row provides the oracle performance in which the ground truth task is given.

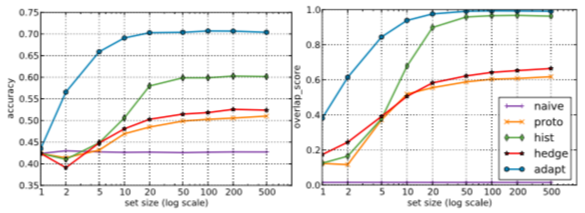

Figure 1. Classification accuracy (left) and the task overlap score (right) with different query set sizes for our method and the baselines.

rithm, while the validation data gave a value of 68.36 and the training data gave 94.69, bothe worse than their unlearning algorithm.

Table 1 summarizes the performance of the methods above with a small query set size and a relatively large size. Further, Figure 1 shows the performance when we vary the size from 1 to 500. It could be observed that when we have a reasonable amount of testing queries, identifying the latent task leads to a significant performance gain than the baseline method that does classification against all possible labels, with an increase of near 30% percent. Even with a small query size, the performance gain is already noticably high, indicating the ability of the algorithm to perform task adaptation with very few images from the latent task.

References

[1] A. Berg, J. Deng, and L. Fei-Fei. ILSVRC2010. http://www.image-net.org/challenges/LSVRC/2010/, 2010. 1

[2] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[3] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011. 1

[4] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR*, 2011. 1