Deep Learning under Privileged Information Using Heteroscedastic Dropout
-
Extended Abstract

John Lambert*, Ozan Sener*, Silvio Savarese
Department of Computer Science
Stanford University
johnwl,osener,ssilvio@stanford.edu

1 Introduction

It is a common belief that human students require far fewer training examples than any learning machine [7]. No doubt this has to do with the fact that effective teachers provide much more than the correct answer to their pupils; they provide an explanation in addition to the result.

In a typical machine learning setup, we present tuples $\{(x_i, y_i)\}_{i=1}^{n}$ to a machine learning model. One way to introduce an "explanation" to a supervised learning system would be to provide some sort of privileged information, which we entitle $x^\star$. In practice, one can incorporate the triplets $\{(x_i, x_i^\star, y_i)\}_{i=1}^{n}$ into a learning system at training time and the testing stage continues to make use of only x, without any access to $x^\star$. In other words, the "Student" has access to privileged information while interacting with the "Teacher" during training, but in the test stage the "Student" operates without the supervision of the "Teacher". This paradigm is called Learning Under Privileged Information (LUPI) and was introduced by Vapnik and Vashist [7]. Vapnik and Vashist [7] provide a LUPI algorithm which is only valid for SVM based methods. Indeed, many have shown that the privileged information can be introduced into the loss function under a multi-task or a distillation loss in an algorithm-agnostic way. However, we raise the question, could it and *should* it be fed in as an input instead of an additional task? If so, how would we go about doing so in an algorithm-agnostic way?

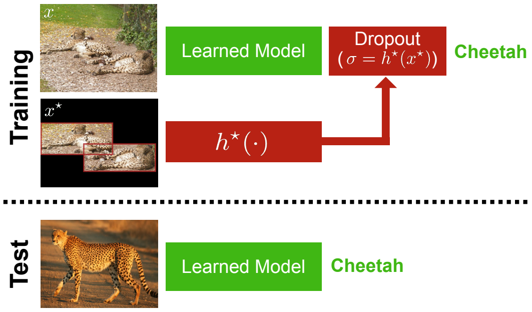

Figure 1: In LUPI paradigm, a teacher provides additional information during training. In this work, we propose to utilize this information in order to control the variance of the Dropout. Our empirical and theoretical analysis suggests that *Heteroscedastic Dropout* significantly increses sample efficiency of both CNNs and RNNs resulting in higher accuracy with much less data.

We define a new class of LUPI algorithms by making a structural specification. We consider a hypothesis class such that each hypothesis is a combination of two functions – namely, a deterministic function taking x as an input and a stochastic function taking $x^\star$ as an input. When $x^\star$ is not available in the test stage, the "Student" simply makes a Bayes optimal decision and marginalizes the model over $x^\star$. Our structural specification makes this marginalization straightforward while not compromising the expressiveness of the model. This structure is natural in the context of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) thanks to the dropout. Dropout is a widely adopted tool to regularize neural networks by multiplying the activations of a neural net-

Second workshop on Bayesian Deep Learning (NIPS 2017), Long Beach, CA, USA.

work at some layer with a random vector. We simply extend the dropout to *heteroscedastic dropout* by making its variance a function of the privileged information. In other words, dropout becomes the stochastic function taking $x^\star$ as an input and marginalizing the function corresponds to not utilizing dropout in the test phase. In order to be able to train the heteroscedastic dropout, we use Gaussian dropout instead of Bernoulli because the key technical tool we use is the re-parameterization trick [3] which is only available for some specific distributions, including the Gaussian.

2 Method

Consider a machine learning problem defined over a compact space $\mathcal{X}$ and a label space $\mathcal{Y}$. We also consider a loss function $l(\cdot,\cdot)$ which compares a prediction with a ground truth label. In learning under privileged information, we also have additional information for each data point defined over a space $\mathcal{X}^\star$, which is only available during the training. In other words, we have access to i.i.d. samples from the data distribution as $x_i, x_i^\star, y_i \sim p(x, x^\star, y)$ during training. However, in test we will only be given $x \sim p(x)$. Formally, given a function class $h(\cdot;\mathbf{w})$ parameterized by $\mathbf{w}$ and data $\{x_i, x_i^\star, y_i\}_{i\in[n]}$, a typical aim is to solve the following optimization problem; $\min_{\mathbf{w}} E_{x,y\sim p(x,y)}[l(y, h(x;\mathbf{w}))]$

We propose to do so by learning a multi-view model using both x and $x^\star$ and to use the marginalized model in test when $x^\star$ is not available. Consider a parametric function class for the multi-view data $h^+ : \mathcal{X} \times \mathcal{X}^\star \to \mathcal{Y}$. The training problem becomes: $\min_{\mathbf{w}} E_{x,x^\star,y\sim p(x,x^\star,y)}[l(y, h^+(x, x^\star;\mathbf{w}))]$

This is equivalent to a classical supervised learning problem defined over a space $\mathcal{X} \times \mathcal{X}^\star$ and any existing method like CNNs can be used. In order to solve the inference problem, we consider the following marginalization: $h(x;\mathbf{w}) \equiv E_{x^\star\sim p(x^\star|x)}[h^+(x, x^\star;\mathbf{w})]$

The major problem in this formulation is the intractability of this expectation, as $p(x^\star|x)$ is unknown. We propose to restrict the class of functions in a way that the expectation is straightforward to compute. The form we propose is a parametric family such that the privileged information controls the variance, whereas the main information (i.e. information available in both training and test) controls the mean. The specific form we use is:

$$h^+(x, x^\star;\mathbf{w}) = h^o(x;\mathbf{w^o}) \odot \mathcal{N}(\mathbf{1}, h^\star(x^\star;\mathbf{w}^\star)) \qquad (1)$$

where $\odot$ represents the Hadamard product and the stochastic function $\mathcal{N}(\mathbf{1}, h^\star(x^\star;\mathbf{w}^\star))$ is a normal random variable with a constant mean function and a covariance function parametrized by $x^\star$ and $\mathbf{w}^\star$. We also decompose $\mathbf{w}$ as two disjoint vectors as $\mathbf{w} = [\mathbf{w^o}, \mathbf{w}^\star]$. Moreover, in this formulation, the expectation defined in (3) becomes straightforward and can be shown to be $h(x;\mathbf{w}) = h^o(x;\mathbf{w^o})$. We visualize this structural specification in Figure 2.

We use neural networks to represent h^o and $h^\star$ and learn their parameters using the information bottleneck. Since the output space is discrete (we address classification), we denote the representation of the data as $h(x;\mathbf{w})$ and compute the output as $\texttt{softmax}(h(x;\mathbf{w}))$. We explain the details of training in the following sections.

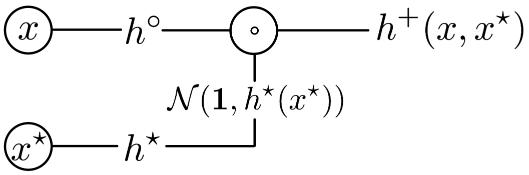

Figure 2: The structure we propose. Privileged information is only used for estimation of the variance of the heteroscedastic dropout.

2.1 Information Bottleneck for Learning

Our framework is closely related to representation learning since the main formulation is learning a stochastic representation as a function of x and $x^\star$. The information bottleneck has already been used for LUPI by controlling the role of x and $x^\star$ separately [5]; however, we do not need this explicit specification because our structural specification directly controls the role of $x^\star$. We use information bottleneck for a rather different reason, its original reason, learning a minimal and sufficient joint representation of $x, x^\star$ which capture all the information about y. This is similar to [1], and we use the same log-Normal assumption. The Lagrangian of the information bottleneck can be written as *(see [6] for details)*; $\mathcal{L} = H(y|z) + \beta I(x, x^\star; z)$ where z is the joint representation of $x, x^\star$ computed as $z = h^+(x, x^\star;\mathbf{w})$. These terms can be computed following a log-Normal prior assumption and the final optimization problem becomes;

$$\min_w \frac{1}{n}\sum_{i=1}^{n} E_{z\sim p_w(z|x,x^\star)}[\log p(y_i|z)] + \beta\|\log h^\star(x_i^\star; w^\star)\| \tag{2}$$

This minimization is simply the cross-entropy loss with regularization over the logarithm of the computed variances of the heteroscedastic dropout, and can be performed via the reparametrization trick in practice when h^o and $h^\star$ are defined as neural networks.

3 Experimental Results

In order to evaluate our method, we perform various experiments using both CNNs and LSTMs. We test our method with CNNs for the task of image classification and with LSTMs for the task of machine translation.

We compare our method with the No-$x^\star$ baseline for image classification using the ImageNet dataset. We perform experiments by varying the number of training examples logarithmically. This is key since the main motivation behind our LUPI method is *learning with less data* rather than having higher accuracy. We report the results in Table 1.

Multi-modal Machine Translation Our method results in a significant accuracy improvement measure by both BLEU and METEOR scores in multi-modal machine translation setting.

In summary, our results overperform all baselines for both multi-modal machine translation and image classification experiments using both CNNs and RNNs. These results suggest that our method is effective and generic.

Table 1: Classification Test Accuracy on 1000 ILSVRC Classes. Because the ILSVRC server prohibits large numbers of test submissions, which we required to evaluate at different sizes of sample data, we use the 50K validation set images as our test set. Where we report "No-$x^\star$," we describe the results of a classical CNN learning method. All 25K models diverged.

	Number of Training Images			
Model	25K	75K	200K	600K
Single Crop top-1				
No-$x^\star$	-	31.23	48.84	**63.35**
Our LUPI	-	**42.26**	**55.51**	62.10
Single Crop top-5				
No-$x^\star$	-	56.33	74.11	**85.14**
Our LUPI	-	**67.42**	**78.89**	84.36
Multi-Crop top-1				
No-$x^\star$	-	33.15	51.33	**65.80**
Our LUPI	-	**45.06**	**58.41**	64.64
Multi-Crop top-5				
No-$x^\star$	-	58.66	76.16	**86.69**
Our LUPI	-	**69.50**	**81.15**	85.77

Table 2: We compare our method for multi-modal machine translation with baselines. We report BLEU and METEOR metrics. Some baselines only report English(en)→German(de) results, and exclude de→en.

	en→de		de→en	
Model	BLEU	Meteor	BLEU	Meteor
No $x^\star$ (following [4])	35.5	54.0	40.19	55.8
Imagination [2]	36.8	55.8	40.5	56.0
Ours	**38.4**	**56.9**	**42.4**	**57.1**

References

[1] A. Achille and S. Soatto. Information dropout: Learning optimal representations through noisy computation. *arXiv preprint arXiv:1611.01353*, 2016. 2

[2] D. Elliott and Á. Kádár. Imagination improves multimodal translation. *CoRR*, abs/1705.04350, 2017. 3

[3] D. Kingma and M. Welling. Auto-encoding variational bayes. In *The International Conference on Learning Representations*, 2014. 2

[4] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. Opennmt: Open-source toolkit for neural machine translation. In *Proc. ACL*, 2017. 3

[5] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Information bottleneck learning using privileged information for visual recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 1496–1505, 2016. 2

[6] N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop (ITW)*, pages 1–5, April 2015. 2

[7] V. Vapnik and A. Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(56):544 – 557, 2009. 1