Unsupervised Domain Adaptation via Calibrating Uncertainties

Ligong Han[1], Yang Zou[2], Ruijiang Gao[3], Lezi Wang[1], and Dimitris Metaxas[1]

[1]Department of Computer Science, Rutgers University
[2]Department of Electrical and Computer Engineering, Carnegie Mellon University
[3]McCombs School of Business, The University of Texas at Austin
l.han@rutgers.edu yzou2@andrew.cmu.edu ruijiang@utexas.edu lw462@cs.rutgers.edu
dnm@cs.rutgers.edu

Abstract

Unsupervised domain adaptation (UDA) aims at inferring class labels for unlabeled target domain given a related labeled source dataset. Intuitively, the model trained on labeled data will produce high uncertainty estimation for unseen data. Under this assumption, models trained in the source domain would produce high uncertainties when tested on the target domain. In this work, we build on this assumption and propose to adapt from source and target domain via calibrating their predictive uncertainties. We employ variational Bayes learning for uncertainty estimation which is quantified as the predicted Rényi entropy on the target domain. We discuss the theoretical properties of our proposed framework and demonstrate its effectiveness on three domain-adaptation tasks.

1. Introduction

The ability to model uncertainty is important in unsupervised domain adaptation (UDA). For example, self-training-based approaches [13, 27] often requires the model to reliably estimate the uncertainty of its prediction on target domain in the pseudo-label selection phase. However, traditional deep neural networks (DNN) can easily assign high confidence to a wrong prediction [4, 15], thus are not able to reliably and quantitatively render the uncertainty given data.

Bayesian Neural Networks (BNN) [17, 4, 1, 9] tackles this problem by taking a Bayesian view of the training process. Instead of obtaining a point estimate of weights, BNN tries to model the distributions over weights. We leverage BNN as a powerful tool to model uncertainties and the investigation on the uncertainties among different domains provides us insights on addressing domain adaptation problem. An observation is that a BNN trained on source domain would produce much higher uncertainties when deployed on target domain. Our uncertainty-based domain-adaptation approach is built on the intuition that a model gives similar uncertainty estimations on the two domains learns to adapt from source to target well. Thus, we propose to directly match the estimated uncertainty between source and target domain during training.

Our contributions are listed as follows:

- We propose a novel framework for unsupervised domain adaptation by calibrating the predictive uncertainty.
- We adopt variational Bayes neural network for uncertainty estimation and discuss its relationship with entropy regularization [6] and self-training [13].
- Preliminary results show that the proposed BNN-based uncertainty calibration is effective and stable in training.

2. Related Work

Shannon entropy is commonly used to quantify the uncertainty of a given distribution. Entropy-based UDA has already been proposed in [23]. Unlike [23], we avoid using adversarial learning which tends to be unstable and hard to train. Also, entropy regularization is proposed in [6] for semi-supervised learning and can be directly applied to UDA. However, our framework is more general since the uncertainty is not necessarily to be the Shannon entropy. In fact, we formalize the uncertainty as Rényi entropy which is a generalization of Shannon entropy. Many other methods in UDA can be modeled under this framework, for example, self-train [13, 27] can be viewed as minimizing the min-entropy which is a special case of Rényi entropy.

As pointed out by [5], directly optimizing the estimated Shannon entropy given data requires the classifier to be locally-Lipschitz [16]. Co-DA [11] and DIRT-T [21] propose to solve this problem by incorporate the locally-Lipschitz constraint via virtual adversarial training (VAT) [16].

Another complimentary line of research employs self-ensemble and shows promising results [2]. Indeed, BNN [4] performs Bayesian ensembling by nature. This is part of the reason why BNN provides a better uncertainty estimation.

3. Uncertainty in Deep Neural Networks

BNN models the uncertainty in DNNs by estimating a posterior over the network parameters. Given the dataset $\mathcal{D} = \{x^{(i)}, y^{(i)}\}_{i=1}^{N}$, the output of BNN is denoted as $f(x|w)$ where x is input data and w is the weights (or parameters). For classification task, f is the predicted logits and the resulting probability vector is given by a softmax function: $P(y|x,w) = \text{softmax}(f(x|w))$. The predictive distribution over labels given input x is: $P(y|x) = \mathbb{E}_{P(w|\mathcal{D})} P(y|x,w)$. Thus, the predictive *uncertainty* can be quantified as the *Rényi entropy*, $\mathrm{H}_\alpha(P(y|x))$. Rényi entropy [24] of order α ($\alpha > 0$) is defined as

$$\mathrm{H}_\alpha(P) = \frac{1}{1-\alpha} \log(\sum_k P_k^\alpha). \tag{1}$$

The limiting value of H_α when $\alpha \to 1$ is the *Shannon entropy*, and $\alpha \to \infty$ corresponds to the *min-entropy*, $\mathrm{H}_\infty(P) = \min_k - \log(P_i) = -\log \max_k P_k$.

As estimating the posterior $P(w|\mathcal{D})$ is often intractable [1, 9], the variational inference is proposed to address this problem, where the posterior of weights is approximated by $Q_\theta(w) \approx P(w|\mathcal{D})$ with parameter θ. Specifically, $Q_\theta(w)$ is estimated by training the model with objective of maximizing the evidence lower bound (ELBO) [10, 4]:

$$ELBO = \underbrace{\mathbb{E}_{Q_\theta(w)} \log P(y|x,w)}_{\text{(I)}} - \underbrace{\mathrm{D}_{KL}(Q_\theta \| P(w))}_{\text{(II)}}, \tag{2}$$

where $P(w)$ is the prior, and the term (I) is the standard cross-entropy loss evaluated at w. Gal *et al*. [3, 4] proposes to view dropout together with weight decay as Bayesian approximation, where sampling from Q_θ is equivalent to performing dropout and the (II) KL divergence term becomes L_2 regularization (or weight decay) on θ.

We adopt the method from [9], where aleatoric and epistemic uncertainty are jointly modeled. In [9], the logits are assumed to be a Gaussian and the reparameterization trick is utilized. The predicted logit is $\hat{f}(x) = \mu_\theta(x) + \sigma_\theta(x)\epsilon$ with $\epsilon \sim N(0, I)$. With a slight abuse of notation, the final predicted probability vector $P(y|x)$ is approximated by Monte Carlo sampling,

$$P(y|x;\theta) = \frac{1}{M} \sum_{m=1}^{M} \text{softmax}(\hat{f}^{(m)}(x)). \tag{3}$$

Plugging the above equation into the MLE term in ELBO, the BNN is trained via a cross-entropy (CE) loss plus weight decay.

4. Domain Adaptation via Calibrating Uncertainties

Denote source and target dataset as $\mathcal{D}_S = \{x^{(s)}, y^{(s)}\}_{s \in \mathcal{S}}$ and $\mathcal{D}_T = \{x^{(t)}\}_{t \in \mathcal{T}}$ respectively, where x^s, x^t indicate the samples and y^s is the label in source domain, and $\mathcal{D} = \mathcal{D}_S \cup \mathcal{D}_T$. We propose to calibrate the predictive uncertainty of target dataset with the source domain uncertainties. Concretely, we minimize the cross-entropy loss in the source domain with the constraint of the predicted entropy (uncertainty) in the target domain:

$$\begin{aligned} \min_\theta \mathcal{L}_{CE} &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathrm{H}_{CE}(y^{(s)}, P(y|x^{(s)};\theta)) \\ \text{s.t.} \quad & \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \mathrm{H}_\alpha(P(y|x^{(t)};\theta)) \le C, \end{aligned} \tag{4}$$

where $\mathrm{H}_{CE}(\cdot,\cdot)$ is the cross-entropy and C indicates the strength of the applied constraint. Rewriting Eq. 4 as a Lagrangian with a multiplier β,

$$\begin{aligned} \mathcal{F} =& \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathrm{H}_{CE}(y^{(s)}, P(y|x^{(s)};\theta)) + \\ & \beta \left(\frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \mathrm{H}_\alpha(P(y|x^{(t)};\theta)) - C \right). \end{aligned} \tag{5}$$

Since $\beta, C \ge 0$ an upper bound on $\mathcal{F}$ is obtained,

$$\begin{aligned} \mathcal{F} \le& \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \mathrm{H}_{CE}(y^{(s)}, P(y|x^{(s)};\theta)) + \\ & \frac{\beta}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \mathrm{H}_\alpha(P(y|x^{(t)};\theta)) = \mathcal{L}_\alpha. \end{aligned} \tag{6}$$

In theory, Eq. 5 can be optimized via dual gradient descent and β is jointly updated along with θ. For simplicity, we follow the work of [8] and fix β as a hyper-parameter in the experiment and minimize the upper bound $\mathcal{L}_\alpha$.

Note that letting $\alpha \to 1$ in Eq. 6 is in fact the (Shannon) entropy regularization as described in [5, 6], except that here we consider a variational BNN. As pointed out in [6], directly optimizing Eq. 6 can be difficult and expectation maximization (EM) algorithms are often used. Proposed in [25, 6], deterministic annealing EM anneals the predicted probabilities as soft-labels and minimizes the resulting cross-entropy. In the extreme case, soft-labels become one-hot vectors and the algorithm terms out to be self-training with pseudo-labels [13]. In our Rényi entropy regularization framework, self-training is essentially optimizing the min-entropy ($\alpha \to \infty$). Then the objective reads

$$\mathcal{L}_\infty = \frac{1}{|\mathcal{S}|}\sum_{s\in\mathcal{S}} \mathrm{H}_{CE}(y^{(s)}, P(y|x^{(s)};\theta)) + \frac{\beta}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \mathrm{H}_{CE}(\hat{y}^{(t)}, P(y|x^{(t)};\theta)), \quad (7)$$

with $\hat{y}^{(t)} = \text{onehot}(\arg\max_{k\in\{1,\dots,K\}} P(y_k|x^{(t)};\theta))$ to be pseudo-labels in target domain. Subscript k denotes the k-th element in a given K-dim vector. The relationship between $\mathcal{L}_1$ and $\mathcal{L}_\infty$ can be immediately realized by noticing that the Shannon entropy is an upper bound of the min-entropy:

$$\begin{aligned}\mathrm{H}_1(P) &= -\sum_k P_k \log(P_k) \geq -\sum_k P_k \log(\max_k P_k) \\ &= -\log(\max_k P_k) = \mathrm{H}_\infty(P) = \mathrm{H}_{CE}(\hat{y}, P) \quad (8)\end{aligned}$$

We build our method on top of class-balanced self-training (CBST) proposed in [27]. CBST seeks to generate pseudo-labels from the most confident predictions that follows an "easy-to-hard" scheme, since jointly learning the model and optimizing pseudo-labels on all unlabeled data is naturally difficult. The authors also propose to normalize the class-wise confidence levels in pseudo-label generation to balance the class distribution. For a detailed formulation, we suggest readers referring Section 4.1 and 4.2 in [27].

5. Experiments

We first show results on three toy datasets MNIST [12], USPS and SVHN [18], where we consider **MNIST→USPS** and **SVHN→MNIST**. Then we present preliminary results on a challenging benchmark: **VisDA17** (classification) [19] which contains 12 classes. We follow the standard protocol in [19, 22, 20].

The accuracies on source and target domains for base models are reported in Table 1. We use DTN [26] as our base model for MNIST→USPS and SVHN→MNIST. To implement its Bayesian variant (BDTN), we add another classifier to predict the logarithm of variance.

Domain adaptation results are shown in Table 2. We can see self-training with pseudo-labels (CBST-BNN-∞) are more stable than directly minimizing the predicted Shannon entropy (CBST-BNN-1). Mean accuracies on VisDA17 dataset are reported in Table 3. Following the protocol in [27], we train a standard ResNet101 [7] as the base model and add a second classifier (denoted as BRes101) to predict logarithm of variance on logits.

6. Conclusion

In this work, we propose to calibrate the predictive uncertainty for unsupervised domain adaptation. The uncertainty is quantified via Bayesian networks under a general Rényi entropy regularization framework. Results show the uncertainty estimation by Bayesian networks is effective and leads to stable performance in unsupervised domain adaptation.

(a) **MNIST**

Model	Source Acc	Target Acc
DTN	100.00	83.94
BDTN-M1	100.00	83.78
BDTN-M5	100.00	86.83
BDTN-M10	100.00	86.28
BDTN-M20	100.00	86.78
BDTN-M100	100.00	87.06

(b) **SVHN**

Model	Source Acc	Target Acc
DTN	97.42	72.91
BDTN-M1	95.91	65.51
BDTN-M5	99.16	71.12
BDTN-M10	99.42	71.38
BDTN-M20	99.50	73.64
BDTN-M100	99.33	74.91

Table 1: Training base models on MNIST and SVHN. BDTN is a modified Bayesian DTN [26], with different M values.

(a) **MNIST→USPS**

Model	Target Acc	Acc Gain
Source-DTN	83.94	-
Source-BDTN	84.78	-
CBST	93.20±0.59	9.26
CBST-BNN-1	89.31±2.02	4.53
CBST-BNN-∞	93.85±0.16	9.07

(b) **SVHN→MNIST**

Model	Target Acc	Acc Gain
Source-DTN	64.48	-
Source-BDTN	71.07	-
MMD [14]	61.1	-
GTA-Res152 [20]	77.1	-
CBST	81.82±4.87	17.34
CBST-BNN-1	89.23±4.64	18.16
CBST-BNN-∞	94.15±0.61	23.08

Table 2: Results on MNIST→USPS and SVHN→MNIST. CBST [27] uses DTN as the base model for self-training. CBST-BNN-∞ uses BDTN as the base model and optimizes $\mathcal{L}_\infty$, while CBST-BNN-1 optimizes $\mathcal{L}_1$.

Model	Target mean-Acc	Acc Gain
Source-Res101	48.02	-
Source-BRes101	46.03	-
CBST	76.81±3.41	28.79
CBST-BNN-∞	80.59±1.39	34.56

Table 3: Preliminary results on VisDA17 [19] classification benchmark.

References

[1] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015. 1, 2

[2] G. French, M. Mackiewicz, and M. Fisher. Self-ensembling for visual domain adaptation. *arXiv preprint arXiv:1706.05208*, 2017. 2

[3] Y. Gal and Z. Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015. 2

[4] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. 1, 2

[5] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005. 1, 2

[6] Y. Grandvalet and Y. Bengio. Entropy regularization. *Semi-supervised learning*, pages 151–168, 2006. 1, 2

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[8] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, volume 3, 2017. 2

[9] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017. 1, 2

[10] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[11] A. Kumar, P. Sattigeri, K. Wadhawan, L. Karlinsky, R. Feris, B. Freeman, and G. Wornell. Co-regularized alignment for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems*, pages 9345–9356, 2018. 1

[12] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3

[13] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013. 1, 2

[14] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015. 3

[15] C. Louizos and M. Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2218–2227. JMLR. org, 2017. 1

[16] T. Miyato, S.-i. Maeda, M. Koyama, K. Nakae, and S. Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015. 1, 2

[17] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012. 1

[18] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011. 3

[19] X. Peng, B. Usman, N. Kaushik, D. Wang, J. Hoffman, and K. Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2021–2026, 2018. 3

[20] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2018. 3

[21] R. Shu, H. H. Bui, H. Narui, and S. Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018. 1

[22] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 3

[23] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. *arXiv preprint arXiv:1811.12833*, 2018. 1

[24] Wikipedia contributors. Rnyi entropy — Wikipedia, the free encyclopedia, 2018. [Online; accessed 13-May-2019]. 2

[25] A. L. Yuille, P. Stolorz, and J. Utans. Statistical physics, mixtures of distributions, and the em algorithm. *Neural Computation*, 6(2):334–340, 1994. 2

[26] X. Zhang, F. X. Yu, S.-F. Chang, and S. Wang. Deep transfer network: Unsupervised domain adaptation. *arXiv preprint arXiv:1503.00591*, 2015. 3

[27] Y. Zou, Z. Yu, B. Vijaya Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 289–305, 2018. 1, 3