AutoAdapt: Automated Segmentation Network Search for Unsupervised Domain Adaptation

Xueqing Deng, Yuxin Tian, Shawn Newsam
UC Merced
xdeng7,ytian8,snewsam@ucmerced.edu

Yi Zhu
Amazon Web Services
yzaws@amazon.com

Abstract

Neural network-based semantic segmentation has achieved remarkable results when large amounts of annotated data are available, that is, in the supervised case. However, such data is expensive to collect and so methods have been developed to adapt models trained on related, often synthetic data for which labels are readily available. Current adaptation approaches do not consider the dependence of the generalization/transferability of these models on network architecture. In this paper, we perform neural architecture search (NAS) to provide architecture-level perspective and analysis for domain adaptation. We identify the optimization gap that exists when searching architectures for unsupervised domain adaptation which makes this NAS problem uniquely difficult. We propose bridging this gap by using maximum mean discrepancy and regional weighted entropy to estimate the accuracy metric. Experimental results on several widely adopted benchmarks show that our proposed AutoAdapt framework indeed discovers architectures that improve the performance of a number of existing adaptation techniques.

1. Introduction

Fully connected convolutional neural networks have shown to be effective for per-pixel classification (semantic segmentation) in images particularly in the supervised case [6, 5, 4, 17]. However, performance degrades when the networks are applied to domains which they were not trained on, that is, when there is a domain shift. Labeling data in the new domain is expensive so researchers have explored adapting the networks in an unsupervised manner, a technique known as unsupervised domain adaptation (UDA) for semantic segmentation.

UDA seeks to narrow the domain gap between a source dataset, for which we have images and labels, and a target dataset, for which we have only images. While good progress has been made in UDA for image classification [10], UDA for semantic segmentation remains a challenge since knowledge of both the image features and the image structure is needed in order to adapt.

Work has been proposed to address UDA at the feature-level [2], at the image-level [18] and at the output space-level [14, 11, 7]. However, we find some interesting observations at the architecture-level, in particular the lack of correlation between how well a model performs on a supervised problem and how well it performs for UDA. Our experimental results show model improvements in the supervised case do not necessarily translate to improvements for UDA. Model search must be performed separately for UDA but doing this manually is a significant undertaking especially since the optimal model can be dataset dependent. We therefore turn to the burgeoning field of neural architecture search (NAS). We design a search space over semantic segmentation models for UDA. We then develop a search framework that finds effective models that can be used with existing UDA approaches.

However, as we will point out, performing automated search over candidate model architectures for UDA including different training and adaptation configurations is a difficult problem. Simply combining existing NAS search methods with UDA approaches is not effective. Tighter integration between these two concepts is needed particularly regarding how to evaluate candidate architectures when labeled data is not available for the target domain. We study the optimization gap that results when using NAS for UDA and bridge this gap by simulating the target domain evaluation metric. We incorporate these contributions into a complete search framework.

We make the following contributions in this work. 1) We approach UDA from a novel perspective, namely *architecture*. We propose AutoAdapt, the first framework to our knowledge to perform automated network search for UDA. 2) We show that a tight integration between network search and domain adaptation is necessary, that it is not enough to simply combine existing NAS and UDA. 3) We propose a novel evaluation metric for model selection to guide the search controller so that it finds architectures close to those that would result if target domain labels were available.

2. AutoAdapt: Domain adaptation at the architecture level

2.1. Preliminaries and motivation

NAS searches the space of valid architectures guided by an objective function. If the space of architectures is parameterized by α and the network weights are parameterized by w then the joint architecture and weight search for a specific problem can be formulated as minimizing an objective function $\mathcal{J}(\alpha, w)$ as follows:

$$\begin{aligned} &\min_{\alpha} \mathcal{J}_{search}(\alpha, w^*(\alpha)) \\ &s.t.\ w^*(\alpha) = \operatorname*{argmin}_{w} \mathcal{L}_{train}(\alpha, w). \end{aligned} \tag{1}$$

This implies a bilevel optimization problem. In order to optimize α, we first need to optimize w. For most supervised tasks, $\mathcal{J}_{search}$ and $\mathcal{L}_{train}$ can be easily minimized by using a training set to update w and a validation set to update α. This is because, the proxy datasets (training and validation) are derived from the same dataset. However, the situation is different for UDA where accuracy can be only computed for the source dataset ($\mathbf{x}^{\mathcal{S}} \in \mathcal{X}^{\mathcal{S}}, \mathbf{y}^{\mathcal{S}} \in \mathcal{Y}^{\mathcal{S}}$). Due to the lack of labels, the accuracy for target dataset ($\mathbf{x}^{\mathcal{T}} \in \mathcal{X}^{\mathcal{T}}$) cannot be computed and so there is no way to update α. Taking adversarial training as an example [16, 14, 11], adversarial loss is adapted for training. However, only an adversarial training loss is meaningless for evaluating the model performance on the target domain. The situation is thus different from NAS for supervised problem where $\mathcal{J}_{search}$ is equal to $\mathcal{L}_{val}$ which is the same loss as $\mathcal{L}_{train}$ (training set) but just with a different dataset (validation set). We identify this as an optimization gap when applying NAS to UDA.

2.2. Bridging the gap without labels

Then main challenge of the search problem is bridging this optimization gap that results when labels are not available for the target domain. The form of the objective used to guide the search is an open problem. Further, it is clear that the performance of a candidate model on the target dataset can only be approximated. We seek a joint metric to approximate this performance. Specifically, maximum mean discrepancy (MMD) is used for computing the distance between source and target domain and entropy is used for determining the model confidence based on output structure.

Regional Weighted Entropy (ReEnt) Entropy has been used in domain adaptation [14, 11] beyond feature and output spaces. As a confidence measurement, entropy indicates where a model is struggling. We therefore investigate using entropy to evaluate how candidate architectures perform on the target dataset. Specifically, given a softmax score map $\mathbf{p}$ for a target image $\mathbf{x}^{\mathcal{T}}$, denoted as $\mathbf{p}_{\mathbf{x}^{\mathcal{T}}}^{(h,w,c)}$ where $h, w, c \in \mathcal{H}, \mathcal{W}, \mathcal{C}$ indicate the indices along height, width and class dimensions of the last softmax layer of the model, we compute the entropy as

$$\text{Ent}_{\mathbf{x}^{\mathcal{T}}}^{(h,w)} = \frac{-1}{\log(\mathcal{C})} \sum_{c=1}^{\mathcal{C}} \mathbf{p}_{\mathbf{x}^{\mathcal{T}}}^{(h,w,c)} \log \mathbf{p}_{\mathbf{x}^{\mathcal{T}}}^{(h,w,c)}. \tag{2}$$

Here, $\mathcal{C}$ indicates the number of semantic classes. $\text{Ent}_{\mathbf{x}^{\mathcal{T}}}^{(h,w)}$ is a 2D matrix, and in order to produce a consistent metric, we average the entropy over all pixel locations as follows:

$$\text{Ent}_{\mathbf{x}^{\mathcal{T}}} = \frac{1}{\mathcal{H}\mathcal{W}} \sum_{h=1}^{\mathcal{H}} \sum_{w=1}^{\mathcal{W}} \text{Ent}_{\mathbf{x}^{\mathcal{T}}}^{(h,w)} \tag{3}$$

However, average entropy does not achieve our goal of evaluating output structure. We therefore propose an improved **Re**gional weighted **Ent**ropy **(ReEnt)** to measure adaptation performance regarding output structure. We partition the input image into regions $\mathcal{R}$ based on the appearance similarity computed using the low-level features. We then compute the entropy value for each region as follows:

$$\text{ReEnt}_{\mathbf{x}^{\mathcal{T}}}^{(r,c)} = \frac{1}{\log(A^r)} \sum_{h',w'}^{r} \mathbf{p}_{\mathbf{x}^{\mathcal{T}}}^{(h',w',c)} \log \mathbf{p}_{\mathbf{x}^{\mathcal{T}}}^{(h',w',c)} \tag{4}$$

where h', w' represent the location of the region r. A^r represents the total locations of the region.

This is different from standard entropy which is used to characterize the probability distribution along class dimensions. Our goal instead is to determine if a model produces smooth predictions within regions that have similar appearance. If the probability is evenly distributed (smooth prediction) within a region, then the entropy will be large entropy[13]. So that all the metrics are consistent (smaller is better), we take the opposite value. Finally we derive the regional weighted entropy for the whole image as:

$$\text{ReEnt}_{\mathbf{x}^{\mathcal{T}}} = \frac{1}{\mathcal{C}} \sum_{c=1}^{\mathcal{C}} \sum_{r}^{\mathcal{R}} \text{ReEnt}_{\mathbf{x}^{\mathcal{T}}}^{(r,c)} \tag{5}$$

Joint Metric for Model Selection Our metric for evaluating the performance of UDA which is also used for model selection consists of two parts, one to evaluate domain alignment (MMD [12]) and another to evaluate output structure. In order to also limit the computational cost of the searched segmentation models, we add a computation complexity (MACs) constraint as a regularization term in the objective function. Our search goal thus becomes minimizing the objective function as follows:

$$\begin{aligned} &\min_{\alpha} \text{MMD}(\mathcal{X}^{\mathcal{S}}, \mathcal{X}^{\mathcal{T}}) + \text{ReEnt}(w^*(\alpha), \alpha, \mathcal{X}^{\mathcal{T}}) + \lambda \text{MACs}(\alpha) \\ &\quad s.t.\ w^*(\alpha) = \operatorname*{argmin}_{w} \mathcal{L}_{DA}(\alpha, w, \mathcal{X}^{\mathcal{S}}, \mathcal{X}^{\mathcal{T}}). \end{aligned} \tag{6}$$

At last, we summarize the proposed AutoAdapt framework for unsupervised domain adaptation. AutoAdapt has two stages. **Stage I: Search and build model** Given a specified search space and a pretrained backbone (seed-net), a child model (super-net) will be created by the search controller (which could be based on reinforcement learning, evolutionary algorithms, gradient descent, etc.). **Stage II: Domain adaptation** Given the child model, our goal is to update the model weights by adapting from the source to the target, that is to compute $w^*(\alpha)$.

Segmentation Model	Backbone	Segmentation Head	Domain Adaptation (mIoU) AdvEnt	IntraDA	CCM	# Params	# MACs	Oracle
DeepLabV2[6]	Res101	ASPP	43.8	45.8	49.9	44.6M	380.5B	65.1
DANet [4]	Res101	Attention	39.8	42.7	47.5	57.7M	490.1B	77.6
DeepLabV3+[5]	Res101	ASPP and Low-level	42.5	45.4	47.1	68.6M	627.0B	77.8
DeeplabV2-S*	SENet[3]	ASPP	41.3	43.9	45.7	49.3M	381.6B	-
Deeplabv2-D[9]	Searched	ASPP	29.8	32.1	36.8	5.0M	6.2B	-
AutoDeeplab[8]	Searched	Searched	37.6	39.4	42.9	10.2M	33.2B	79.7
Random search	NAS	NAS	37.8	38.5	39.4	45.5M	381.2B	-
DARTS-1*	NAS	ASPP	24.8	27.1	28.8	5.0M	6.2B	-
DARTS-2*	NAS	ASPP	28.4	33.9	39.8	5.0M	6.2B	-
PC-DARTS*	NAS	ASPP	35.7	39.8	43.1	8.0M	8.7B	-
AutoDeeplab*	NAS	NAS	32.8	35.1	37.5	10.2M	33.2B	-
AutoAdapt (MMD)	NAS	NAS	44.2	45.9	49.8	44.7M	380.1B	-
AutoAdapt (Ent.)	NAS	NAS	44.3	46.0	49.9	44.8M	380.5B	-
AutoAdapt (ReEnt)	NAS	NAS	44.6	46.5	49.9	44.8M	380.5B	-
AutoAdapt (Joint)	NAS	NAS	**45.3**	**47.2**	**50.2**	44.9M	380.5B	-

Table 1: Comparisons between automatic search and hand-crafted architectures for UDA. mIoU is reported as the evaluation metric. DeepLabV2-S denotes our implementation with backbone SENet, and DeeplabV2-D denotes our implementation with backbone DARTS which is searched on CIFAR and then trained on ImageNet. * denotes our implementation. The difference between searched and NAS is : Searched: searched by other application and train for UDA; NAS: perform searching for UDA and train for UDA.

3. Experimental results

3.1. Effectiveness of AutoAdapt

In this section, we perform comparisons to answer the question "Is automated model search really necessary and effective for UDA?" That is, how does automated model search for UDA compare with selecting hand-crafted architectures and then applying UDA. Details are shown in Tab. 1. Adaptation is conducted from GTA5→Cityscapes and mIoU on the Cityscape validation set is reported.

1) Comparisons to hand-crafted methods we replace the commonly used semantic segmentation network DeeplabV2 [6] with the state-of-the-art networks DeeplabV3+ [5] and DANet [4] in a UDA framework. For fair comparison, the backbone remains the same for all three segmentation models. Rows 1-3 show the results of using advanced segmentation networks with various UDA methods. We can see DANet and DeeplabV3 outperform DeeplabV2 in the non-domain adaptation case (oracle=training using the target dataset) by large margins of over 10%. But, when there is domain adaptation, they perform worse than DeeplabV2, often with over 2% lower mIoU. We also investigate how state-of-the-art backbone affects UDA. We replace the backbone of DeeplabV2 which is a ResNet-101 network with SENet [3] and call this DeeplavV2-S. We realize there are other advanced backbones like Xception [1] that could be considered. But, for fair comparison, we only consider SE-Net[3] as it is included in our search space. Row 4 shows the results for DeeplabV2-S. We observe that an advanced backbone does not improve performance as it achieves only 42.3%, 43.9% and 45.7% mIoU when used with AdvEnt, IntraDA and CCM which is lower than the DeeplabV2-ResNet101 results. We conclude that manually designing segmentation backbone or head for non-domain adaptation scenarios does not necessarily lead to improved UDA results.

2) Comparisons to searched methods We select the NAS discovered backbone DARTS [9] with an ASPP (DeeplabV2-D) in DeeplabV2 and the NAS discovered segmentation network AutoDeeplab [8] to separately replace the segmentation networks for different UDA methods. Rows 5 and 6 show these architectures perform poorly for UDA. We conclude that replacing the network with architectures discovered using existing NAS techniques is not effective for UDA. This is because these architectures are discovered in supervised manner which does not take into account the transferability/generalization of the models.

3) Comparisons to NAS methods We modify the search loss function with a UDA loss in DARTS [9], PC-DARTS [15] and AutoDeeplab [8] to see if existing NAS methods can find effective architectures for UDA. Rows 8-11 show the performance of the modified NAS methods with a UDA loss. Replacing the search loss function does not provide a reasonable search space for UDA due to the optimization gap. The difference between DARTS-1 and DARTS-2 is the input size which results in different network depths due to GPU memory constraints. PC-DARTS is seen to perform better because it allows us to set a larger number of network depths. Finally, we see from the last row in Tab. 1 that our proposed AutoAdapt approach which per-

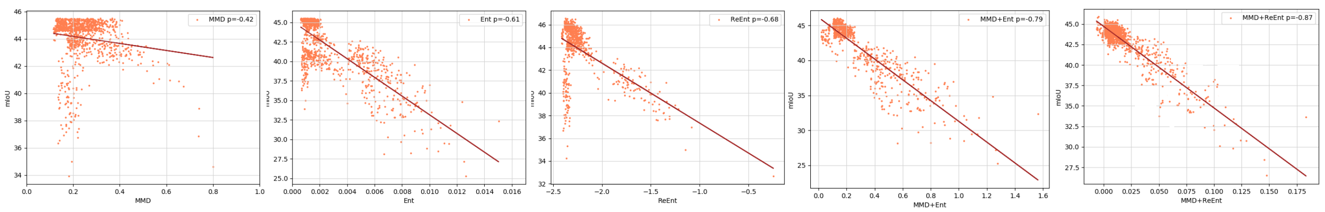

Figure 1: Correlation between entropy and mIoU. Entropy is highly (inversely) correlated with mIoU as indicated by Spearman's rank correlation (p) for all metrics used in AutoAdapt. All experiments are performed on GTA5→Cityscapes.

forms architecture search specifically for UDA provides the best performance for all three domain adaptation methods.

3.2. Ablation studies

We expect that an effective surrogate evaluation metric should be correlated with the standard metric if it could be computed (if labels were available for the target domain). Fig. 1 shows the (negative) correlation graphically as well as through Spearman's rank correlation coefficient ρ between variations of our novel evaluation metric (MMD, Entropy (Ent), and Regional Weighted Entropy (ReEnt) and the combination of them) and mIoU for GTA5→Cityscapes adapted using AdvEnt. We stress that we only use the Cityscape labels to compute mIoU for this correlation analysis–we do not use the labels in our AutoAdapt framework. Fig. 1 shows that MMD which only measures the distribution distance between source and target domains performs the worst. Combining MMD and the output structure metric Ent or ReEnt improves the correlation significantly. The proposed ReEnt by itself shows higher correlation than standard entropy. The proposed joint metric which consists of MMD and ReEnt achieves the best correlation with $\rho = -0.87$.

4. Conclusion

We approach UDA from a novel perspective, namely architecture search. We propose AutoAdapt, the first framework to our knowledge to perform automated network search for UDA. Even though we propose several strategies to accelerate our evolutionary-based search, it is still slower than gradient-based methods. In the future, we plan to improve AutoAdapt by developing a fully gradient-based framework. A current limitation of our work is that we still rely on hand-crafted domain adaptation methods, like AdvEnt and CCM. We hope to overcome this by automatically adapting not only the model architecture but also the model weights in a unified framework.

References

[1] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 3

[2] J. Hoffman, D. Wang, F. Yu, and T. Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016. 1

[3] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 3

[4] H. Tian Y.Li Y. Bao Z. Fang H. Lu J. Fu, J. Liu. Dual Attention Network for Scene Segmentation. In *CVPR*, 2019. 1, 3

[5] G. Papandreou F. Schroff H. Adam L. Chen, Y. Zhu. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *ECCV*, 2018. 1, 3

[6] I. Kokkinos K. Murphy A. Yuille L. Chen, G. Papandreou. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *TPAMI*, 2017. 1, 3

[7] G. Li, G.and Kang, W. Liu, Y. Wei, and Y. Yang. Content-consistent matching for domain adaptive semantic segmentation. In *ECCV*, 2020. 1

[8] C. Liu, L. Chen, F. Schroff, H. Adam, W. Hua, A. Yuille, and F. Li. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019. 3

[9] H. Liu, K. Simonyan, and Y. Yang. DARTS: Differentiable architecture search. In *Proceedings of the IEEE International Conference on Computer Vision(ICCV)*, 2019. 3

[10] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 1

[11] F. Pan, I. Shin, F. Rameau, S. Lee, and I. Kweon. Unsupervised intra-domain adaptation for semantic segmentation through self-supervision. In *CVPR*, 2020. 1, 2

[12] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum Classifier Discrepancy for Unsupervised Domain Adaptation. In *CVPR*, 2018. 2

[13] C Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 1948. 2

[14] T. Vu, H. Jain, M. Bucher, Ma. Cord, and P. Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *CVPR*, 2019. 1, 2

[15] Y. Xu, L. Xie, X. Zhang, X. Chen, G. Qi, Q. Tian, and H. Xiong. PC-DART: Partial channel connections for memory-efficient architecture search. In *ICLR*, 2020. 3

[16] S. Schulter M. Chandraker Y. Tsai, K. Sohn. Domain Adaptation for Structured Output via Discriminative Representations. In *ICCV*, 2019. 2

[17] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1

[18] Y Zou, Z. Yu, B. Vijaya Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 1