Adversarial Branch Architecture Search for Unsupervised Domain Adaptation

Luca Robbiano[1], Muhammad Rameez Ur Rahman[2], Fabio Galasso[2], Barbara Caputo[1], Fabio Maria Carlucci[3]
Politecnico di Torino[1], Sapienza University of Rome[2], Huawei Noah's Ark Lab[3],
luca.robbiano@polito.it

Abstract

Training a neural network to perform a real-world task is an expensive process. A large amount of labelled data is needed, and often the trained model fails to generalize to visual domains other than the training one. Unsupervised Domain Adaptation methods can help to solve both issues: provided that enough unlabelled data is available from the domain of interest, a network can be trained by using only labelled samples from a -possibly synthetic- other domain. Several proposed approaches for UDA rely on a multi-branch architecture, but they need human experts to manually adjust the adaptation branch for a specific backbone architecture (e.g. ResNet). This process can be automated by using Neural Architecture Search, but a method to rank architectures without using target labels is needed. We tackle this issue by proposing EMS, an ensemble model selection method that combines several metrics to define a target accuracy proxy, which we use to optimize the secondary branch.

1 Introduction

Unsupervised Domain Adaptation (UDA) enables the transfer of domain knowledge from a source domain to a target domain, for which only unlabelled data is available. Several UDA methods employ a dual-branch architecture. The main branch performs the required task, and the secondary branch performs a task that leads the network to generalize to the target domain. For example, adversarial methods train the main branch supervisedly on source data, and add a discriminator branch, which reduces the distributional domain gap by making the domains indistinguishable. The architecture of the adaptation branch is typically manually designed by experts, with little or no guarantee that it would be optimal for any backbone or data. Neural Architecture Search could be employed to optimize the architecture of the auxiliary branch. However, the lack of target labels prevents the network to be evaluated directly on target, making the optimization against target accuracy challenging.

We propose Adversarial Branch Architecture Search for Unsupervised Domain Adaptation (ABAS) [Robbiano *et al.*, 2022] to address this limitation. More specifically, we employ an Ensemble Model Selection metric to define a proxy for target accuracy without the need for target labels. Then, we use a multi-fidelity Bayesian Optimization algorithm (BOHB) [Falkner *et al.*, 2018] to search for both architecture and hyperparameters of the adversarial branch. We conduct an in-depth performance evaluation of two adversarial methods, DANN [Ganin *et al.*, 2016] and ALDA [Chen *et al.*, 2020], and show how ABAS can consistently improve their performance on three datasets.

2 Method

As previously mentioned, a number of UDA methods employ a dual-branch architecture. While in our work we explicitly apply ABAS to domain adversarial learning, it would be trivial to extend the idea to other multi-branch methods.

2.1 Adversarial branches for Domain Adaptation

Let us define a dataset $\boldsymbol{X}_s = \{\boldsymbol{x}_s^i, y_s^i\}_{i=0}^{N_s}$ drawn from a labeled source domain $\mathcal{S}$, and a dataset $\boldsymbol{X}_t = \{\boldsymbol{x}_t^j\}_{j=0}^{N_t}$ from a different unlabeled target domain $\mathcal{T}$, sharing the same set of categories. Our goal is to maximize the classification accuracy on $\boldsymbol{X}_t$ while training on $\boldsymbol{X}_s$. The overall architecture consists of 3 components: the feature extractor G, the labeled classifier C and the adversarial branch D. The basic intuition behind discriminative adversarial approaches for domain adaptation is that a domain classifier is trained to distinguish between source and target samples; as its gradient is reversed, the features in G are trained so that source and target have similar representations. The model is trained by optimizing the following objective function:

$$\min_{G,C,D} \mathcal{L}_{CE}(C(G(x_s)), y_s) + \lambda \mathcal{L}_{Adv}(D(G(x_s \cup x_t)) \quad (1)$$

Where $\mathcal{L}_{CE}$ is the standard cross-entropy loss, computed on source samples only, and $\mathcal{L}_{Adv}$ is the adversarial loss trained to reduce the domain gap. How $\mathcal{L}_{Adv}$ is defined depends on the specific adversarial method which is being implemented (refer to the referenced papers for further details).

2.2 Bayesian Optimization for AutoML

For efficiency, we adopt BOHB, a multi-fidelity combination of BO and Hyperband [Li *et al.*, 2017]. While single fidelity BO evaluates all samples with full budget, BOHB resorts to

Algorithm 1 BO auxiliary branch optimization

```
1: Input: Domain adaptation method $Q$, Performance estimator $E$, BO surrogate model $p(f|\mathbf{\Theta}, D)$ and acquisition function $\alpha(\mathbf{\Theta}|D)$
2: for $t = 1$ to $T$ do
3:     Recommend $\{\mathbf{\Theta}_t^j\}_{j=1}^B = \arg\max \alpha_{t-1}(\mathbf{\Theta}|D)$
4:     for $j = 1$ to $B$ in parallel do
5:         Build the auxiliary branch and evaluate $E(Q(\mathbf{\Theta}_t^j))$ to obtain its corresponding performance metric $f_t^j$
6:     end for
7:     Update $D$ and thus $p(f|\mathbf{\Theta}, D)$ with $\{\mathbf{\Theta}_t^j, f_t^j\}_{j=1}^B$
8: end for
9: return The best performing model according to $E(Q(\mathbf{\Theta}^*))$
```

partial evaluations with smaller-than-full budget, excluding bad configurations early in the search process and reserving computational resources for the most promising configurations. So, given the same time budget, it evaluates many more configurations and it achieves faster optimization than competing methods.

2.3 Ensemble-based Model Selection (EMS)

Model selection in the context of UDA is extremely challenging, as target labels are not available. To overcome this, we propose using an ensemble of weakly correlating predictors. Experimentally, we show how a linear regressor over these metrics, trained on a single dataset, can capture the ranking between models on a *different* unseen dataset, thus highlighting the generalization power of this approach.

In our implementation, we adopt six weakly-correlating metrics: target entropy, diversity of class prediction, Silhouette and Calinski-Harabasz scores to measure how much features are well-clustered, accuracy on the source domain, and consistency of pseudolabels during training. Note how all of these metrics, with the exception of the source accuracy, are computed on the target domain.

2.4 Auxiliary branch optimization

A full overview can be found in Fig. 1 and Algorithm 1: ABAS combines BOHB [Falkner *et al.*, 2018] and an adversarial method of choice, with our model selection strategy. Given a fixed budget, ABAS performs a number of rounds, alternating between sampling and evaluating. At each step, we sample the Bayesian acquisition function $\alpha(\mathbf{\Theta}|D)$ for B different configurations $\{\mathbf{\Theta}_t^j\}_{j=1}^B$. The configurations are used to build auxiliary branches for method Q and the resulting network is trained on the target setting. After training, supervised (from the source) and unsupervised (from the target) features are collated for our model selection module. The ensemble predictor finally gives feedback to the BO process. This procedure is repeated for a given number of rounds and in the end the best performing model is returned.

3 Potential impact in industrial settings

The ability to generalize to different domains is a major need in many computer vision applications. Since manual data annotation is way more expensive than unlabelled data collection, it is often desirable to train a model for use on a new domain without repeating the data annotation process. Moreover, this allows generating synthetic data (which is labelled for free by definition) and using it to train models able to perform tasks on real-world data. An example of an application is autonomous driving: labelled images can be gathered from a simulated environment and used for real-world models [Dosovitskiy *et al.*, 2017; Zolfaghari Bengar *et al.*, 2019]. Another field that can benefit from the improvement of UDA methods is robotics. Machines that make use of computer vision often happen to reduce their performance dramatically due to environmental changes: for example, different lighting conditions could be enough to make a robot useless. On the other side, the performance could also decrease within the same environment, if the robot needs to manipulate new objects. Without domain adaptation and generalization methods, launching a new product or even changing the texture of an existing one would require the collection of a new handcrafted dataset.

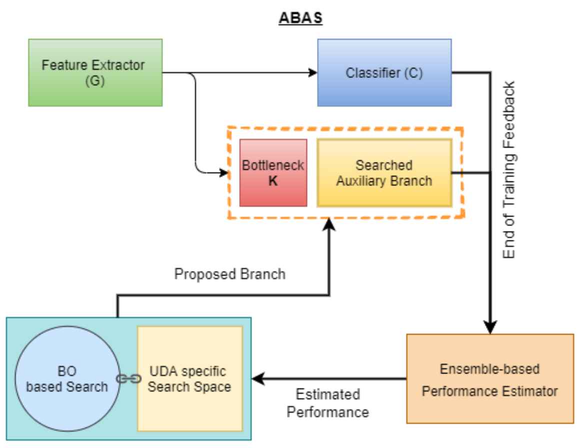

Figure 1: At each iteration, the Bayesian Optimization (BO) acquisition function proposes a number of candidate branch architectures. After training, we extract a number of supervised (source) and unsupervised (target) features which are fed into our ensemble-based performance estimator. Its feedback is sent to BO, thus closing the loop.

4 Conclusions

The shift from hand-crafted to automated architecture search is ongoing, as it is witnessed by the flourishing NAS research and techniques. ABAS fills in two important requirements of NAS for UDA: it provides a data-driven strategy EMS for model selection, circumventing the lack of target labels; and it focuses on searching the architecture of auxiliary branches attached to a pre-trained backbone, essential practise for state-of-the-art performance. Our experiments show how ABAS consistently improves the performance of the baselined adversarial UDA methods (Tab. 1, 2, 3), confirming the importance of the adversarial branch architecture. Future work might evaluate ABAS on a broader set of UDA methods, possibly including self-supervised approaches.

	A-W	A-D	D-W	D-A	W-D	W-A	AVG
DANN	85.0±0.5	82.5±0.5	96.7±0.2	63.9±1.1	99.2±0.3	64.7±0.7	82.0
ABAS-DANN	**89.4**	**87.6**	**98.4**	**64.1**	**99.8**	**69.3**	**84.8**
ALDA	94.8±0.6	91.6±0.9	98.3±0.4	69.6±0.9	99.9±0.0	70.8±0.7	87.5
ABAS-ALDA	**96.1**	**95.0**	98.5	**75.9**	**100.0**	70.7	**89.4**

Table 1: Results of ABAS on Office31 with a ResNet-50 backbone for all the source-target combinations of the domains *Amazon*, *Webcam* and *DSLR*.

	Ar-Cl	Ar-Pr	Ar-Rw	Cl-Ar	Cl-Pr	Cl-Rw	Pr-Ar	Pr-Cl	Pr-Rw	Rw-Ar	Rw-Cl	Rw-Pr	AVG
DANN	39.8	58.0	68.1	48.6	57.0	59.9	46.9	38.4	68.8	63.2	47.7	75.9	56.0
ABAS-DANN	**44.9**	**61.1**	**71.2**	**52.7**	**60.4**	**62.5**	**50.1**	**43.1**	**70.0**	**65.4**	**50.9**	**77.1**	**59.1**
ALDA	46.4	68.6	74.6	57.6	67.0	69.4	57.2	46.3	75.6	69.2	53.3	80.8	63.8
ABAS-ALDA	**51.5**	**71.7**	**75.5**	**59.8**	**69.4**	69.5	**59.8**	47.1	**77.7**	**70.6**	**55.2**	80.2	**65.7**

Table 2: Results of ABAS on Office-Home, using a ResNet-50 backbone for all the source-target combinations of the domains *Art*, *Clipart*, *Product* and *Real World*.

	P-A	C-A	S-A	A-P	C-P	S-P	A-C	S-C	P-C	A-S	C-S	P-S	AVG
ALDA	89.3	91.9	69.9	98.3	97.3	63.4	85.1	75.2	**74.3**	79.2	70.6	60.7	79.6
ALDA + EMS	90.2	**92.0**	72.3	98.4	**97.8**	69.5	86.0	82.1	72.1	**80.7**	**75.1**	**66.1**	81.9
ABAS-ALDA	**93.1**	91.8	**78.1**	**98.7**	**97.8**	**70.8**	**88.7**	**84.9**	69.7	79.8	69.5	64.9	**82.3**

Table 3: Results of ABAS on PACS, using a ResNet-50 backbone, + EMS: experiments run with our model selection strategy.

References

[Chen *et al.*, 2020] Minghao Chen, Shuai Zhao, Haifeng Liu, and Deng Cai. Adversarial-learned loss for domain adaptation. In *AAAI*, pages 3521–3528, 2020.

[Dosovitskiy *et al.*, 2017] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[Falkner *et al.*, 2018] Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *International Conference on Machine Learning*, pages 1437–1446. PMLR, 2018.

[Ganin *et al.*, 2016] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[Li *et al.*, 2017] Lisha Li, Kevin G Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: Bandit-based configuration evaluation for hyperparameter optimization. In *ICLR (Poster)*, 2017.

[Robbiano *et al.*, 2022] Luca Robbiano, Muhammad Rameez Ur Rahman, Fabio Galasso, Barbara Caputo, and Fabio Maria Carlucci. Adversarial branch architecture search for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2918–2928, 2022.

[Zolfaghari Bengar *et al.*, 2019] Javad Zolfaghari Bengar, Abel Gonzalez-Garcia, Gabriel Villalonga, Bogdan Raducanu, Hamed H Aghdam, Mikhail Mozerov, Antonio M Lopez, and Joost van de Weijer. Temporal coherence for active learning in videos. *arXiv preprint arXiv:1908.11757*, 2019.