Neural Architecture Search for Domain Adaption

Chaoyue Xing

Research School of Computer Science, Australian National University, Canberra Australia
u6920870@anu.edu.au

Abstract. Neural architecture search (NAS) is a method to automatically search for a suitable network structure based on specific data sets and tasks. This method has achieved superior performance in many tasks, such as Re-Identification(Re-ID), image classification. However, few people have considered the performance of NAS in differently distributed training sets and test sets. In this paper, we provide two baselines for domain adapation using NAS, VehicleX to VeRi776 and PersonX to market1501. Both of them are Re-ID tasks. Their mAP of our model reached 5.44% and 16.48%, respectively, which were 15.85% and 4.84% lower than the mAP of using resnet50.

Keywords: Neural Architecture Search, Domain Adaption, Re-Identification

1 Introduction

To advance network designs for specific datasets, neural architecture search automates the net architecture engineering process by reinforcement supervision [13] or through neuro-evlolution [13] or formulating the task in a differentiable manner [6]. Conventional NAS models solves a bilevel optimization problem to derive neural architecture α along with the network parameters ω [6]:

$$\phi_{\alpha,\omega} = argmin_{\alpha}\mathcal{L}_{val}(\omega^*(\alpha), \alpha) \tag{1}$$

$$\omega^*(\alpha) = argmin_{\omega}\mathcal{L}_{train}(\omega, \alpha) \tag{2}$$

where $\mathcal{L}_{train}$ indicate the training loss and $\mathcal{L}_{val}$ indicate the validation loss. We can finally get specific deep convolutional networks suitable for specific data sets and specific tasks through NAS. Competitive performance on tasks such as image classification [4, 5] and re-identification [8] demonstrated by recent works.

However, designs of existing NAS algorithms typically do not consider that domain gap between the training and testing set, neglecting the scenario where two data domains or multiple feature distributions are of interest. Due to a phenomenon known as dataset bias or domain shift [2], deep convolutional networks trained on one large dataset do not generalize well to novel datasets and tasks[1]. Through experiments, we found that if the training set and validation set of the NAS come from the same domain, and the test set comes from different domain, the performance of the model searched from NAS is not well.

In our work, we provide two baselines for domain adaption of reID task using NAS. The first one is VechicleX [**?**] to VeRi776 [12], which means we search the neural architecture on the synthetic vehicle data set, VehicleX, and evaluate our model on the real-world vehicle data set, VeRi776. The second one is PersonX [9] to Market1501 [12], which is as same as VehicleX to VeRi776. PersonX data set is the synthetic person data set and Market1501 is the real-world person data set. Our model is based on the Auto-ReID [8], a NAS model designed for the person re-ID tasks. To introduce the domain gap in the model and let the model search for neural architectures suitable for the domain adaption, we change the bilevel optimization problem as follows:

$$\phi_{\alpha,\omega} = argmin_{\alpha}\mathcal{L}^t_{val}(\omega^*(\alpha), \alpha) \tag{3}$$

$$\omega^*(\alpha) = argmin_{\omega}\mathcal{L}^s_{train}(\omega, \alpha) \tag{4}$$

where $\mathcal{L}^t_{val}$ and $\mathcal{L}^s_{train}$ means training set is source domain and validation set is target domain. Our goal is minimising the validation loss to overcome the domain gap. Finally, the mean Average Precision (mAP) of VehicleX to VeRi776 is 16.48%, which is 6.04% higher than directly use Auto-ReID on the VehicleX. The mAP of PersonX to Market1501 is 5.44%, which is 1.62% higher than than directly use Auto-ReID on the PersonX.

2 Method

In this section we will introduce how to search a CNN with high performance for domain adaption of reID. We will introduce the background of NAS in section 2.1. Then we will introduce the search algorithm for domain adaption of reID in section 2.2. Besides, we will introduce the search space for domain adaption of reID in section 2.3.

In this section, we give the overview of the NAS as Fig 1.

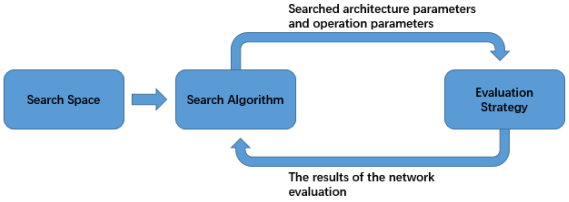

Fig. 1. The overview of the NAS. For a given search space, we use the seach algorithm to find the appropriate architecture parameters and operation parameters each time, and then pass the searched network into the evaluation strategy for evaluation, and perform the next search based on the evaluation results. In particular, NAS for domain adaption, architecture parameters and operation parameters will be searched on different distributed datasets.

2.1 Preliminary

Generally, NAS approaches will search for the architecture of a neural cell and stack multiple copies of the neural cell to construct a CNN model [6, 14]. A neural cell is composed of different layers, and each layer has many candidate possibilities. By searching the most likely candidate for each layer, we can get the architecture of the neural cell. And each neural cell uses the output of the previous neural cell as its own input, and then generates a new output as the input of the next neural cell. We will follow the previous NAS approaches [6, 14] and AutoReID [8] to search for the architecture of the neural cell..

Specifically, a neural cell is a directed acyclic graph (DAG) with an ordered sequence of N nodes [6]. Each node $x^{(i)}$ is a latent representation and each directed edge (i, j) is associated with some operation that transforms $x^{(i)}$. We assume the cell to have two input nodes and a single output node, we will apply two operations on these tow input respectively and sum these two inputs to output. And we have an operation candidate set $\mathcal{O}$ to search the applied operation. Following Auto-ReID [8], we use the following operations in our $\mathcal{O}$: (1) 3×3 max pooling, (2) 3×3 average pooling, (3) 3×3 depth-wise separable convolution, (4) 3×3 dilated convolution, (5) zero operation (none), (6) identity mapping. The i-th node in the c-th neural cell can be represented as a 4-tuple, i.e., (I^c_{i1}, I^c_{i2}, $\mathcal{O}^c_{i1}$, $\mathcal{O}^c_{i2}$). Besides, the output tensor of the i-th node in the c-th neural cell is:

$$I^c_i = \mathcal{O}^c_{i1}(I^c_{i1}) + \mathcal{O}^c_{i2}(I^c_{i2}) \tag{5}$$

where $\mathcal{O}^c_{i1}$ and $\mathcal{O}^c_{i2}$ are selected operations from operations $\mathcal{O}$. I^c_{i1} and I^c_{i2} are selected from the candidate input tensors $\mathcal{I}^c_i$, which consists of output tensors from the last two neural cells(I^{c-1} and I^{c-2}) and output tensors from the previous block in the current cell.

To search for the best choicess of $\mathcal{O}^c_{i1}$ and $mathcal I^c_{i1}$ in 5, we relax the categorical choice of a particular operation as a softmax over all possible operations following [6]:

$$\mathcal{O}^c_{i1}(I^c_{i1}) = \sum_{H \in \mathcal{I}_i}^{c} \sum_{o \in \mathcal{O}} \frac{exp(\alpha_o^{(H,i)})}{\sum_{o' \in \mathcal{O}} exp(\alpha_{o'}^{(H,i)})} o(H) \tag{6}$$

where $\alpha = \{\alpha_o^{(H,i)}\}$, **architecture parameters**, represents the topology structure for a neural cell. Denote the parameters of all operations in $\mathcal{O}$ as ω, named as **operation parameters**, a typical differentiable NAS approach [6] jointly trains ω on the training set and α on the validation set. After training, the strength of H to I^c_i is defined as $max_{o \in O, o \neq none} \frac{exp(\alpha_o^{(H,i)})}{\sum_{o' \in \mathcal{O}} exp(\alpha_{o'}^{(H,i)})}$. The H $\in \mathcal{I}^c_i$ with the maximum strength is selected as $^c_{i1}$, and the operation with the maximum weight for $\mathcal{I}^c_{i1}$ is selected as $\mathcal{O}^c_{i1}$. This paradigm is designed for the classification problem. Auto-ReID modified the search algorithm and search space on this basis.

2.2 Domain Adaption Search Algorithm

Previous NAS approaches have focused on finding a well-performing architecture suitable for classification, in which the softmax with cross-entropy loss is applied to optimizing both α and [6, 14]. Different from classification, reID tasks aim to learn a discriminative feature extractor during training, then extract the feature to retrieve images of same identity during evaluation. Simply inheriting the cross-entropy loss can not guarantee a good retrieve performance, so Auto-ReID introduce the triplet loss to the NAS. Based on Auto-ReID, We introduced knowledge of domain gap in the model.

Network Structure For the reID backbone, the macro structure of ResNet [3] is used, where each residual layer is replaced by a neural cell. We search the topology structure of neural cells. Use f as the feature extracted from the backbone, we use one embedding layer to transfer the feature f into g following [10], and the feature g is mapped into the logits h with the output dimension of C by another linear transformation layer, where where C denotes the number of training identities. Two dropout layers are added between f & g and g& h, respectively.

Algorithm 1 Domain Ddaption Search Algorithm

Require: the architecture parameter α and the operation parameter ω; the training set $\mathcal{D}_T$ from synthetic data set and the evaluation set $\mathcal{D}_E$ from real data set; a class-balance data sample;
Split $\mathcal{D}_E$ into the evaluation set $\mathcal{D}_{train}$ and the search validation set $\mathcal{D}_{val}$
while not converged **do**
Use the sampler to get batch data from$\mathcal{D}_{train}$
Update ω via the retrieval loss in Eq.(9)
Use the sampler to get batch data from$\mathcal{D}_{val}$
Update α via the retrieval loss in Eq.(9)
end while
Obtain the final CNN from α following the strategy in [6].
Evaluate the trained CNN on the evaluation set $\mathcal{D}_E$

Loss Function Auto-ReID applies both softmax with cross-entropy loss and triplet loss as follows:

$$L_s = \sum_{i=1}^{N} -log\frac{exp(h_i[c])}{\sum_{c'=1}^{C} exp(h_i[c'])} \tag{7}$$

where H_i indicates the feature h of the i-th sample, and $h_i[c]$ indicates the c-th element in h_i. N is the number of samples during training. The reID model usually applies the triplet loss as:

$$L_t = \sum_{i=1}^{N} max(margin, ||f_i - f_i^p|| - ||f_i - f_i^n||) \tag{8}$$

where f_i indicates the feature f of the i-th sample. f_i^p indicates the hardest positive feature of f_i. The margin term indicates the margin of triplet loss. Since the triplet loss is very sensitive to batch data, we should carefully sample the training data in each batch. We use a class-balanced data sampler to sample the batch data of triple loss. The sampler first uniformly samples certain identities, and then randomly samples the same number of images for each identity.

The final loss is the weighted combination of all tasks:

$$L_{ret} = \lambda L_s + (1 - \lambda)L_t \tag{9}$$

where $\lambda \in [0, 1]$ is a weight balancing of L_s and L_t.

We show out overall algorithm in Alg. 1, which solves a bi-level optimization problem. We search for a robust reID model by alternatively optimizing α with L_t. We will introduce the domain gap into the model by use whole synthetic data set as training set and a small part of real-world data set as validation set. After we find a robust CNN for the domain adaption of reID task, we evaluate this CNN in the standard way [11].

2.3 Domain Adaption Search space

For NAS, it is very important that the search space can contain all possible candidate CNN networks. Generally, we can use "NASNet search space" [14] as the search space. "NASNet search space" contains many different types of layers, but none of these structures can handle person information well. So Auto-ReID proposes a new search space suitable for re-ID.

The search space for reID search space is $\mathcal{O}_{reid}$: (1) part-aware module, (2) 3×3 max pooling, (3) 3×3 average pooling, (4) 3×3 depth-wise separable convolution, (5) 3×3 dilated convolution, (6) zero operation, and (7) identity mapping. Among them, the part-aware module is a different component from other search spaces. Through the part-aware module, we can divide the person feature vector into multiple parts and perform self-attention learning separately, so that we can incorporate global information into each part vectors to enhance its body structure cues. Finally, we merge these parts with the original features. In this way, we can integrate the captured body structure information into the input features.

3 Experiments

We empirically evaluate the proposed method in this section. We will introduce the datasets and preprocessing for the datasets in section 3.1, evaluation metrics in section 3.2, the implementation details in section 3.3, the valuation of the two baseline in section 3.4 and 3.5.

3.1 Datasets and Preprocessing

Overall, we have explored four large-scale datasets. Two of these are real datasets and the other two are synthetic datasets.

VeRi776 [7] contains 49,357 images of 776 vehicles from 20 cameras. The dataset is collected in the real traffic scenario, which is close to the setting of CityFlow. There are 11,579 gallery images, 1,678 query images of 200 identities, 37,778 training images of 576 identities.

Task	Method	Data	mAP	Rank-1	Rank-5
VehicleX to VeRi776	NAS	S	3.82	8.67	19.74
	NAS	S+R	5.44	12.28	20.08
	Resnet50	S	21.29	51.25	67.70
PersonX to Market1501	NAS	S	10.44	22.00	39.79
	NAS	S+R	16.48	34.98	56.00
	Resnet50	S	14.19	26.53	–
	Resnet50(SPGAN)	S+R	21.35	41.11	–

Table 1. We analyze the performance for Vehiclex to VeRi776 and PersonX to Market1501. For method and data, S means the model is only trained on synthetic dataset, S+R means the model is trained using synthetic dataset and real-world dataset. We provide the performance of our model and the baseline using resnet50 in this table. All performances are tested on real-world datasets.

VehicleX [11] VehicleX is a large-scale synthetic dataset. Created in Unity, it contains 1,362 vehicles of various 3D models with fully editable attributes. In our paper, we used VehicleX dataset, which is performed domain adaption form itself to VeRi776. It contains 75,516 images of 1362 identities.

Market1501 [12] is a real-world person reID dataset. It contains 19,372 gallery images, 3,368 query images and 12,396 training images collected from six cameras. There are 751 identities in training set and 750 identities in the test set and they have no overlap. Every identity in the training set has 17.2 images on average.

PersonX [9] PersonX is a large-scale data synthesis engine. PersonX contains 1,266 manually designed identities and editable visual variables. We use the PersonX dataset for The Visual Domain Adaptation Challenge 2020, which contains 30,816 gallery images, 5,136 query images, 9,840 training images. Query images are of 410 identities, training images are of 1181 identities.

For VehicleX to VeRi776, we use the whole VehicleX dataset as the training set and we select a picture from each identity of the training set of the veri776 data set as the validation set. For each image of training set, we will resize the height and width of the image to [256,256] first, then we will perfrom randomhorizontalFlip, padding, randomcrop, normalize, randomerasing on it. We just resize the each image to [256,256] and normalize for validation set. We can enhance the data and increase the robustness of the model through such preprocessing method of the data.

For PersonX to Market1501, the training set of PersonX is used for the training set of the NAS model, and we also use the same method to form the validation set through training set of Market1501. Training set and validation set are also transformed as VehicleX to VeRi776, the only difference is that the length and height become [256,128]. This is because the bounding box of a vehicle is generally a square, while the bounding box of a person is generally a rectangle.

3.2 Evaluation Metrics

To evaluate the performance of our Auto-ReID and compare with other reID methods, we use top-1 accuracy, top-5 accuracy and mAP. To calculate mAP, we calculate the area under the Precision-Recall curve for each query, which is also called average precision (AP), the mean value of APs of all queries is mAP. The map considers both accuracy and recall, thus providing a more comprehensive evaluation.

3.3 Implementation Details

According to Auto-ReID [8], We choose the ResNet macro structure to construct the overall network. This network has a 3x3 convolutional head and four blocks sequentially, where each block has several neural cells. We set the number of cells in each block to 2. The channel of the first convolutional layer is 32, and each block will double the number of channels. The first cell in the 2-th, 3-th, and 4- th block is a reduction cell, and other cells are the normal cell.

During searching, batch size is 64, the total epoch is 50. We use momentum SGD to optimize ω with the learning rate of 0.001, momentum of 0.9. We use Adam to optimize α with the learning rate of 0.005. The weight decay for both SGD and Adam is set as 0.0005. The margin is set as 0.3 when using the retrieval objective. The λ is 0.5.

Searching for PersonX to Market1501 costs about 2 days to finish one searching procedure, and VehicleX to VeRi costs about 3 days. They are respectively trained on a singe GTX3090.

3.4 Comparison for VehicleX to VeRi776

As shown in the table 1, we do an ablation study. NAS&S is our NAS model trained totally on the synthetic datasets. NAS&R+S is the model we provided in this paper, training set comes from source domain, validation set comes from target domain. The mAP and of NAS&R+S is 5.44%, which is 1.62% highed than the model only trained in the source domain and the top-1 accuracy, top-5 accuracy of NAS&R+S are also higher. This proves that our model learned the domain gap of VehicleX to VeRi776 when searching for CNN. However, the mAP, top-1 accuracy, and top-5 accuracy of NAS&R+S and NAS&S are far lower than the baseline trained on

synthetic datasets using resnet50 [11]. This may be because Auto-ReID is designed for the person Re-ID task, and its generalization ability on the vehicle dataset is not good enough.

3.5 Comparison for PersonX to Market1501

We also do an ablation study for PersonX to Market1501. From table 1, the mAP, top-1 accuracy, top-5 accuracy of NAS&R+S are 16.48%, 34.98%, 56.00%, which are 6.04%, 12.98%, 16.21% higher than NAS&S respectively. This also proves that our model learned the domain gap of PersonX to Market776 when searching for CNN. By comparing with the baseline using resnet50, whether the data is S or R+S, the effect of the NAS model is worse than resnet50. This may be caused by the following reasons. Because the training time is too long, the epoch of our NAS is only set 50 times, so the performance of the searched network is lower than the performance of resnet50. Secondly, the method of using SPGAN to fuse real data and synthetic data is better than our method for domain adaption. And we can also find that the performance of PersonX to Market1501 is more close to the baseline using resnet50, compared with VehicleX to VeRi776, which can also explain to a certain extent that the reason for the poor performance of the vehicle may be the insufficient generalization ability of Auto-ReID.

4 Conclusion

In this paper, we have modified the training method of NAS. By setting the training set and validation set of NAS to source domain and target domain respectively, our goal is to let NAS design a neural network structure adapted to the domain gap for the specific dataset. And we provide two NAS-based domain adaption baselines for this purpose, PersonX to Market1501 and VehicleX to VeRi. These two tasks are both Re-ID tasks. Through experiments, we found that our NAS model can better adapt to the domain gap compared to the NAS model trained only in the source domain. However, the performance of our model is lower than that of resnet50. We hope that our work can make more people pay attention to NAS for domain adaption, and pave the way for improving the performance of NAS for domain adation in the future. Exploring how to generate a suitable network structure for domain adaption is an interesting direction.

5 Future work

In this paper, we only discussed the case of NAS for domain adaption for Re-ID tasks. However, NAS is also widely used in image calssification and other fields, and can get a competitive result. Therefore, in future work, we will join the discussion on the performance of NAS for domain adaption under different tasks. And through experiments, although our results are improved compared to the NAS trained only in the source domain, the performance is still not as good as resnet50. Our future work will also focus on improving the performance of NAS for domain adaption.

References

1. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: International conference on machine learning. pp. 647–655. PMLR (2014)
2. Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schölkopf, B.: Covariate shift by kernel mean matching. Dataset shift in machine learning **3**(4), 5 (2009)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
4. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.J., Fei-Fei, L., Yuille, A., Huang, J., Murphy, K.: Progressive neural architecture search. In: Proceedings of the European conference on computer vision (ECCV). pp. 19–34 (2018)
5. Liu, H., Simonyan, K., Vinyals, O., Fernando, C., Kavukcuoglu, K.: Hierarchical representations for efficient architecture search. arXiv preprint arXiv:1711.00436 (2017)
6. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055 (2018)
7. Liu, X., Liu, W., Mei, T., Ma, H.: A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In: European conference on computer vision. pp. 869–884. Springer (2016)
8. Quan, R., Dong, X., Wu, Y., Zhu, L., Yang, Y.: Auto-reid: Searching for a part-aware convnet for person re-identification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3750–3759 (2019)
9. Sun, X., Zheng, L.: Dissecting person re-identification from the viewpoint of viewpoint. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 608–617 (2019)
10. Sun, Y., Zheng, L., Yang, Y., Tian, Q., Wang, S.: Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In: Proceedings of the European conference on computer vision (ECCV). pp. 480–496 (2018)
11. Yao, Y., Zheng, L., Yang, X., Naphade, M., Gedeon, T.: Simulating content consistent vehicle datasets with attribute descent. arXiv preprint arXiv:1912.08855 (2019)
12. Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., Tian, Q.: Scalable person re-identification: A benchmark. In: Proceedings of the IEEE international conference on computer vision. pp. 1116–1124 (2015)
13. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578 (2016)
14. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8697–8710 (2018)