AM-LFS: AutoML for Loss Function Search

Chuming Li[1]*, Xin Yuan[1]*, Chen Lin[1]*, Minghao Guo[1], Wei Wu[1], Junjie Yan[1], Wanli Ouyang[2]
[1]SenseTime Group Limited
[2]The University of Sydney, SenseTime Computer Vision Research Group, Australia
{lichuming,yuanxin,linchen,guominghao,wuwei,yanjunjie}@sensetime.com; wanli.ouyang@sydney.edu.au

Abstract

Designing an effective loss function plays an important role in visual analysis. Most existing loss function designs rely on hand-crafted heuristics that require domain experts to explore the large design space, which is usually sub-optimal and time-consuming. In this paper, we propose AutoML for Loss Function Search (AM-LFS) which leverages REINFORCE to search loss functions during the training process. The key contribution of this work is the design of search space which can guarantee the generalization and transferability on different vision tasks by including a bunch of existing prevailing loss functions in a unified formulation. We also propose an efficient optimization framework which can dynamically optimize the parameters of loss function's distribution during training. Extensive experimental results on four benchmark datasets show that, without any tricks, our method outperforms existing hand-crafted loss functions in various computer vision tasks.

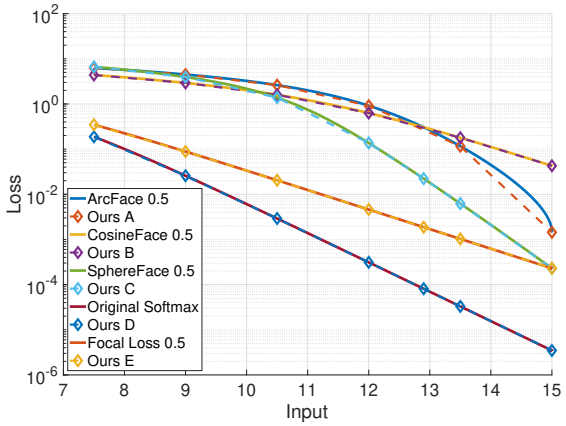

Figure 1. The motivation of the proposed loss function search space. The figure shows that the candidate loss functions in our search space (dotted lines) can well approximate the according existing loss functions (solid lines). The x-axis indicates the loss input and the y-axis indicates the output loss values in log-scale.

1. Introduction

Convolutional neural networks have significantly boosted the performance of a variety of visual analysis tasks, such as image classification [16, 33, 10], face recognition [22, 37, 5], person re-identification [24, 38, 34, 6] and object detection [8, 28] in recent years due to its high capacity in learning discriminative features. Aside from developing features from deeper networks to get better performance, better loss functions have also been proven to be effective on improving the performance of the computer vision frameworks in most recent works [22, 20].

Conventional CNN-based vision frameworks usually apply a widely-used softmax loss to high level features. L-softmax [23] is a variant of softmax loss which added multiplicative angular to each class to improve feature discrimination in classification and verification tasks. [22] introduced A-softmax by applying L-softmax [23] to face recognition task with weights normalization. [39, 37] moved the angular margin into cosine space to overcome the optimization difficulty of [22] and achieved state-of-the-art performance. [5] can obtain more discriminative deep features for face recognition by incorporating the additive angular margin. In addition to the above margin-based softmax loss functions, focal loss [20] is another variant of softmax loss which was proposed to adopt a re-weighting scheme to address the data imbalance problems in object detection. While these methods improve performance over the traditional softmax loss, they still come with some limitations: (1) Most existing methods rely on hand-crafted heuristics that require great efforts from domain experts to explore the large design space, which is usually sub-optimal and time-consuming. (2) These methods are usually task-specific, which may lack transferability when applied to other vision tasks. By utilizing AutoML methods for exploration in a well-designed loss function search space, a generic solution can be proposed to further improve the performance.

In this paper, we propose AutoML for Loss Function

*equal contribution

Search (AM-LFS) method from a hyper-parameter optimization perspective. Based on the analysis of existing modification on the loss functions, we design a novel and effective search space, which is illustrated in Figure 1 and formulate hyper-parameters of loss function as a parameterized probability distribution for sampling. The proposed search space include a bunch of popular loss designs, whose sampled candidate loss functions can adjust the gradients of examples at different difficulty levels and balance the significance of intra-class distance and inter-class distance during training. We further propose a bilevel framework which allows the parameterized distribution to be optimized simultaneously with the network parameters. In this bilevel setting, the inner objective is the minimization of sampled loss w.r.t network parameters, while the outer objective is the maximization of rewards (e.g. accuracy or mAP) w.r.t loss function distribution. After the AM-LFS training finishes, the network parameters can be directly deployed for evaluation and then get rid of the heavily re-training steps. Furthermore, since our method is based on the loss layer without modification on the network architecture of specific tasks, it can be easily applied to off-the-shelf modern classification and verification frameworks. We summarize the contributions of this work as follows:

(1) We provide an analysis based on the existing loss function design and propose a novel and effective search space which can guarantee the generalization and transferability on different vision tasks.

(2) We propose an efficient optimization framework which can dynamically optimize the distribution for sampling of the loss functions.

(3) The proposed approach advances the performance of the state-of-the-art methods on popular classification, face and person re-id databases including CIFAR-10, MegaFace, Market-1501 and DukeMTMC-reID.

2. Related Work

2.1. Loss Function

Loss function plays an important role in deep feature learning of various computer vision tasks. Softmax loss is a widely-used loss for CNN-based vision frameworks. A large margin Softmax (L-Softmax) [23] modified softmax loss by adding multiplicative angular constraints to each identity to improve feature discrimination in classification and verification tasks. SphereFace [22] applies L-Softmax to deep face recognition with weights normalization. CosineFace [39, 37] and ArcFace [5] can achieve the state-of-the-art performance on the MegaFace with more discriminative deep features by incorporating the cosine margin and additive angular margin, respectively. SphereReID [6] adopted the sphere softmax and trained the model end-to-end to achieve the state-of-the-art results on the challenging person reid datasets. For object detection, focal loss [20] and gradient harmonized detector [17] adopt a re-weighting scheme to address the class imbalance problem. However, noisy labels can lead to misleading gradient, which may be amplified by gradient re-weighting schemes and cause a training failure.

2.2. AutoML

AutoML was proposed as an AI-based solution to the challenging tasks by offering faster solutions creation and models outperforming those designed by hand. Recent works of automatically searching neural network architectures (NAS) has greatly improve the performance of neural networks. NAS utilizes reinforcement learning [48, 47, 46] and genetic algorithms [27, 42, 31] to search the transferable network blocks whose performance surpasses many manually designed architectures. However, all the above methods require massive computation during the search, particularly thousands of GPU days. Recent efforts such as [21, 25, 26], utilized several techniques trying to reduce the search cost. [21] is a differential based method which utilized a bilevel optimization procedure for the jointly training of the real-valued architecture parameters and the model parameters. Several methods attempt to automatically search architectures with fast inference speed by explicitly taking the inference latency as a constrain [36, 1] or implicitly encoding the topology information [9]. In addition to the network architecture search, [12] utilized AutoML techniques for effective model compression, where the pruning rate was automatically decided by reinforcement learning. [41] utilized a teacher model inspired by [7] to guide the student model training with dynamic loss function. However, how to design a generic search space to various domains of tasks and an efficient optimization framework remain an open problem.

3. Approach

In this section, we first revisit several loss function designs from a novel perspective and then analyze their influence on the training procedure and reformulating them in a unified expression. We hence propose a novel search space on the basis of the unified expression to include the good properties of existing popular loss function designs. We also propose an optimization framework by utilizing AutoML methods for efficient loss function search during the whole training process, which is illustrated in Figure 2.

3.1. Revisiting Loss Function Design

Softmax Loss: The most widely used softmax loss can be written as

$$L = \frac{1}{N}\sum_i L_i = \frac{1}{N}\sum_i -log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right), \qquad (1)$$

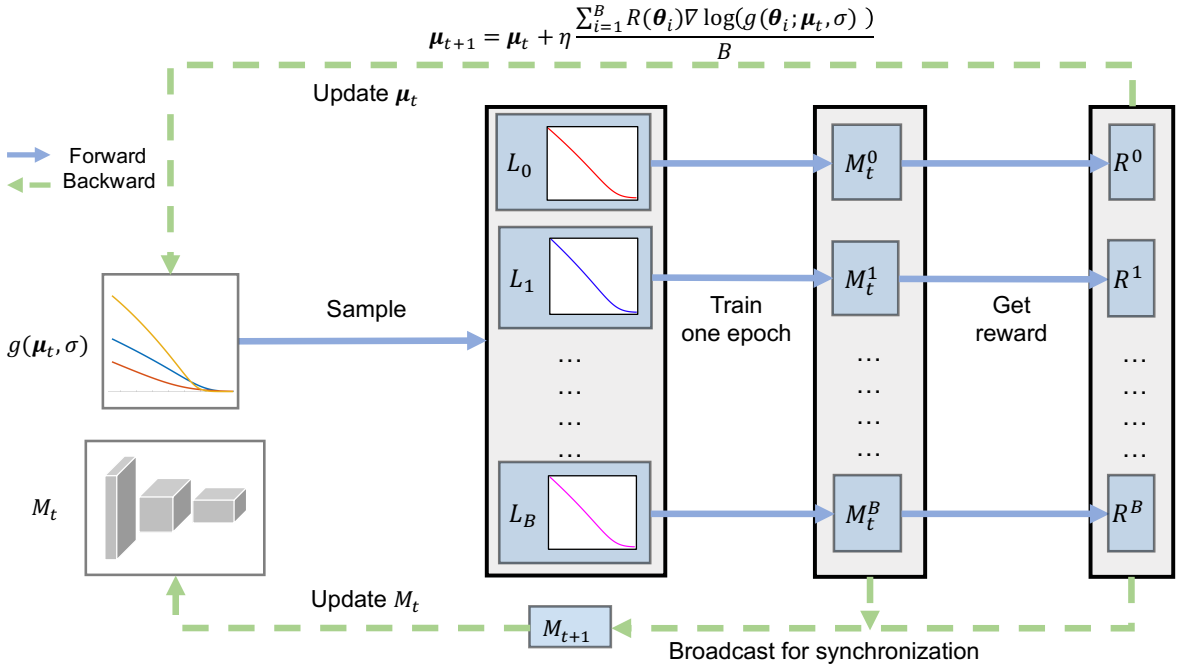

Figure 2. The bilevel optimization framework of our proposed AM-LFS approach. In this bilevel setting, the inner objective is the minimization of sampled loss w.r.t network parameters, while the outer objective is the maximization of rewards (e.g. accuracy or mAP) w.r.t loss function distribution. After each train epoch, we broadcast the model parameters with the highest reward to each sample for synchronization.

where $\boldsymbol{x}_i$ and y_i is the i-th input feature and the label, respectively. f_j denotes the j-th element ($j \in [1, C]$, where C is the number of classes) of the vector of scores $\boldsymbol{f}$, N is the length of training set. $\boldsymbol{f}$ is usually the activation of a fully connected layer $\boldsymbol{W}$. We further denote f_j as $\boldsymbol{W}_j^T \boldsymbol{x}_i$, where $\boldsymbol{W}_j$ is the j-th column of $\boldsymbol{W}$. Hence f_j can be formulated as:

$$f_j = \|\boldsymbol{W}_j\| \, \|\boldsymbol{x}_i\| \, cos\left(\theta_j\right), \tag{2}$$

where $\theta_j (0 \leq \theta_j \leq \pi)$ is the angle between the vector $\boldsymbol{W}_j$ and $\boldsymbol{x}_i$, $\|\boldsymbol{W}_j\|$ and $\|\boldsymbol{x}_i\|$ are L_2 norm of $\boldsymbol{W}_j$ and $\boldsymbol{x}_i$. Note that $\|\boldsymbol{W}_j\| = 1$ or $\|\boldsymbol{x}_i\| = 1$ if $\boldsymbol{W}_j$ or $\boldsymbol{x}_i$ is normalized. Thus the original softmax loss can be rewritten as:

$$L_i = -log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\| \|\boldsymbol{x}_i\| cos\left(\theta_{y_i}\right)}}{\sum_j e^{\|\boldsymbol{W}_j\| \|\boldsymbol{x}_i\| cos\left(\theta_j\right)}}\right) \tag{3}$$

We can easily obtain several variants of the original softmax loss such as margin-based softmax loss and focal loss by inserting transforms into Eq. 3.

Margin-based Softmax Loss: The family of margin-based loss functions can be obtained by inserting a continuously differentiable transform function $t\left(\right)$ between the the norm $\|\boldsymbol{W}_{y_i}\| \, \|\boldsymbol{x}_i\|$ and $cos\left(\theta_{y_i}\right)$, which can be written as:

$$L_i^t = -log\left(\frac{e^{\|\boldsymbol{W}_{y_i}\| \|\boldsymbol{x}_i\| t\left(cos\left(\theta_{y_i}\right)\right)}}{e^{\|\boldsymbol{W}_{y_i}\| \|\boldsymbol{x}_i\| t\left(cos\left(\theta_{y_i}\right)\right)} + \sum_{j \neq y_i} e^{\|\boldsymbol{W}_j\| \|\boldsymbol{x}_i\| cos\left(\theta_j\right)}}\right) \tag{4}$$

Table 1. The expressions of some existing transforms including L-softmax, A-softmax and ArcFace.

transform	expression
L-softmax [23]	$t(x) = cos(m \cdot arccos(x))$
A-softmax [22]	$t(x) = x + m$
ArcFace [5]	$t(x) = cos(arccos(x) + m)$

where t is used to distinguish margin-based softmax loss functions with different transforms. We also list several transforms including L-softmax, A-softmax, ArcFace and their corresponding expressions in Table. 3.1,

Focal Loss: This is also a variant which can be derived from softmax loss by adding transform function at another location, which can be described as

$$L_i^t = -\tau(log(p_{y_i})) \tag{5}$$

$$\tau(x) = x(1 - e^x)^\alpha. \tag{6}$$

3.2. Analysis of Loss Function

In this section, we first discuss the impacts of margin-based softmax loss functions on the training process from a new perspective based on the analysis of the relative significance of intra-class distance and inter-class distance. We

define the inter-class distance d_j as the distance between the feature $\boldsymbol{x}_i$ and the class center $\boldsymbol{W}_j$. The intra-class distance d_{y_i} can be defined in a similar way. For simplification, we assume that $\boldsymbol{W}_j$ and $\boldsymbol{x}_i$ are normalized, which means $\|\boldsymbol{W}_j\| = \|\boldsymbol{x}_i\| = 1$ and $f_j = cos(\theta_j)$. Under this assumption, the relationship between f_j and d_j can be described as follows:

$$d_j^2 = (\boldsymbol{x}_i - \boldsymbol{W}_j)^2 = 2 - 2f_j. \tag{7}$$

As a result, we can analyze the impact of margin-based softmax loss on f_{y_i} and f_j instead. The impact can be calculated as the norm of gradient passed to the activation layer $\boldsymbol{f}$. Specifically, the gradients of the loss layer with respect to f_{y_i} and f_j are:

$$||\frac{\partial L_i^t}{\partial f_{y_i}}|| = (1 - p_{y_i}^t)t'(f_{y_i}), \tag{8}$$

$$||\frac{\partial L_i^t}{\partial f_j}|| = p_j^t, \tag{9}$$

where

$$p_{y_i}^t = \frac{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|t\left(cos\left(\theta_{y_i}\right)\right)}}{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|t\left(cos\left(\theta_{y_i}\right)\right)} + \sum_{j\neq y_i} e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|cos(\theta_j)}}, \tag{10}$$

$$p_j^t = \frac{e^{\|\boldsymbol{W}_j\|\|\boldsymbol{x}_i\|cos(\theta_j)}}{e^{\|\boldsymbol{W}_{y_i}\|\|\boldsymbol{x}_i\|t\left(cos\left(\theta_{y_i}\right)\right)} + \sum_{j\neq y_i} e^{\|W_j\|\|\boldsymbol{x}_i\|cos(\theta_j)}} \tag{11}$$

We further define **relative significance** of intra-class distance to inter-class distance as the **ratio** of norms of the gradients of f_{y_i} and f_j with respect to the margin-based softmax loss, which is described as follows:

$$r_i^t = \frac{||\frac{\partial L_i^t}{\partial f_{y_i}}||}{||\frac{\partial L_i^t}{\partial f_j}||} = \frac{(1 - p_{y_i}^t)}{p_j^t}t'(f_{y_i}), \tag{12}$$

while with respect to the original softmax loss, this significance ratio is

$$r_i^o = \frac{||\frac{\partial L_i^o}{\partial f_{y_i}}||}{||\frac{\partial L_i^o}{\partial f_j}||} = \frac{(1 - p_{y_i}^o)}{p_j^o}. \tag{13}$$

Where o is the identity transform. The impact of margin-based loss on the relative significance of intra-class distance to inter-class distance can be calculated as the ratio of r_i^t and r_i:

$$\begin{aligned}\frac{r_i^t}{r_i^o} = \frac{\frac{(1-p_{y_i}^t)}{p_j^t}}{\frac{(1-p_{y_i}^o)}{p_j^o}}t'(f_{y_i}) &= \frac{\frac{\sum_{t\neq y_i} e^{\|W_t\|\|x_i\|cos(\theta_t)}}{e^{\|W_j\|\|x_i\|cos(\theta_j)}}}{\frac{\sum_{t\neq y_i} e^{\|W_t\|\|x_i\|cos(\theta_t)}}{e^{\|W_j\|\|x_i\|cos(\theta_j)}}}t'(f_{y_i}) \\ &= t'(f_{y_i})\end{aligned} \tag{14}$$

Here we conclude that margin-based softmax loss layer mainly functions as a controller to change the relative significance of intra-class distance to inter-class distance by its derivative $t'(f_{y_i})$ during the training process. In addition to the margin-based softmax loss, we also analyze the impact of the focal loss. The gradient of focal loss with respect to the activation $\boldsymbol{f}$ equals to that of original softmax loss multiply $\tau'(log(p_{y_i})$. This gradient leads to a totally different yet very effective impacts on the training process, which monotonically decreases with the log-likehood and help balance samples at different levels of difficulty.

3.3. Search Space

Based on the analysis above, we can insert two transforms τ and t into the original softmax loss to generate loss functions with a unified formulation, which have both capabilities of balancing (1) intra-class distance and inter-class distance, (2) samples at different levels of difficulty. The unified formulation can be written as:

$$L_i^{\tau,t} = -\tau(log\left(p_{y_i}^t\right)), \tag{15}$$

where τ and t are any function with positive gradient. To ensure τ has a bounded definition domain $[0, 1]$ hence reduce the complexity of searching in this space, we exchange τ and log,

$$L_i^{\tau,t} = -log(\tau\left(p_{y_i}^t\right)). \tag{16}$$

We prove that these two search space defined in Eq. 15 and Eq. 16 are equivalent: for any $\tau_1(x)$ in Eq. 15, we can get the same loss function by simply setting $\tau_2(x) = e^{\tau_1(log(x))}$ in Eq. 16.

Our search space is formulated by the choices of τ and t, whose impacts on the training procedure are decided by derivatives τ' and t' according to the analysis above. As a result, we simply set the candidate set as piecewise linear functions that evenly divide the definition domain, which ensures independent slopes and bias within each interval. Take the function t as an example:

$$t(x) = a_i^t x + b_i^t, x \in [\zeta_i^t, \zeta_{i+1}^t], \tag{17}$$

where $\boldsymbol{\zeta}^t = [\zeta_0, ...\zeta_M]$($M$ is the number of intervals) are the end points of the intervals, and $\zeta_{i+1}^t - \zeta_i^t = (\zeta_M^t - \zeta_0^t)/M$ for $i \in [0, M-1]$.

We also analyze the effectiveness of the components in our search space. Samples in different intervals are at different levels of difficulty. For example, larger-value intervals contain easier samples with smaller intra-class distance d_{y_i}. Since t' denotes the relative significance between intra-class and inter-class distance, we assign each interval with independent slope $t' = a_i^t$ to ensure the loss function can independently balance the significance of intra-class and inter-class distance at different levels of difficulty. Similarly, τ's candidate set is the described by a_i^τ, b_i^τ and ζ_i^τ, which balances the significance of samples at different difficulty levels. The biases b_i^t and b_i^τ guarantees the independence of

each interval from previous intervals. We set $\boldsymbol{\zeta}^t$ and $\boldsymbol{\zeta}^\tau$ as constant values which evenly divide definition domains into M intervals. We define $\boldsymbol{\theta} = [\boldsymbol{a}^{t^T}, \boldsymbol{b}^{t^T}, \boldsymbol{a}^{\tau^T}, \boldsymbol{b}^{\tau^T}]^T$. Given $\boldsymbol{\zeta}^t, \boldsymbol{\zeta}^\tau$, the loss function $L^{\boldsymbol{\theta}} = L^{t,\tau}$ can decided merely by $\boldsymbol{\theta}$. Thus, our search space can be parameterized by $\mathcal{L} = \{L^{\boldsymbol{\theta}}\}$.

3.4. Optimization

Suppose we have a network model $M_{\boldsymbol{\omega}}$ parameterized by $\boldsymbol{\omega}$, train set $\mathcal{D}_t = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and validation set $\mathcal{D}_v$, our target of loss function search is to maximize the model $M_{\boldsymbol{\omega}}$'s rewards $r(M_{\boldsymbol{\omega}}; \mathcal{D}_v)$ (e.g. accuracy or mAP) on validation set D_v with respect to $\boldsymbol{\theta} = [\boldsymbol{a}^{t^T}, \boldsymbol{b}^{t^T}, \boldsymbol{a}^{\tau^T}, \boldsymbol{b}^{\tau^T}]^T$, and the model $M_{\boldsymbol{\omega}}$ is obtained by minimizing the search loss:

$$\begin{aligned} &\max_{\boldsymbol{\theta}} R(\boldsymbol{\theta}) = r(M_{\boldsymbol{\omega}^*(\boldsymbol{\theta})}, \mathcal{D}_v) \\ &\text{s.t.} \quad \boldsymbol{\omega}^*(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\omega}} \sum_{(x,y)\in\mathcal{D}_t} L^{\boldsymbol{\theta}}(M_{\boldsymbol{\omega}}(x), y), \end{aligned} \tag{18}$$

This refers to a standard bilevel optimization problem [3] where loss function parameters $\boldsymbol{\theta}$ are regarded as hyper-parameters. We trained model parameters $\boldsymbol{\omega}$ which minimize the training loss $L^{\boldsymbol{\theta}}$ at the inner level, while seeking a good loss function hyper-parameters $\boldsymbol{\theta}$ which results in a model parameter $\boldsymbol{\omega}^*$ that maximizes the reward on the validation set $\mathcal{D}_v$ at the outer level.

To solve this problem, we propose an hyper-parameter optimization method which samples B hyper-parameters $\{\boldsymbol{\theta_1}, ...\boldsymbol{\theta_B}\}$ from a distribution at each training epoch and use them to train the current model. In our AM-LFS, we model these hyper-parameters $\boldsymbol{\theta}$ as independent Gaussian distributions, described by

$$\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma I). \tag{19}$$

After training for one epoch, B models are generated and the rewards of these models are used to update the distribution of hyper-parameters by REINFORCE [40] as follows,

$$\boldsymbol{\mu_{t+1}} = \boldsymbol{\mu_t} + \eta \frac{1}{B} \sum_{i=1}^{B} R(\boldsymbol{\theta_i}) \nabla_{\boldsymbol{\theta}} log(g(\boldsymbol{\theta_i}; \boldsymbol{\mu_t}, \boldsymbol{\sigma})) \tag{20}$$

, where $g(\boldsymbol{\theta}; \boldsymbol{\mu}, \sigma)$ is PDF of Gaussian distribution. The model with the highest score is used in next epoch. At last, when the training converges, we direct take the model with the highest score $r(M_{\boldsymbol{\omega}^*(\boldsymbol{\theta})}, \mathcal{D}_v)$ as the final model without any retraining. To simplify the problem, we fix σ as constant and optimize over $\boldsymbol{\mu}$. The training procedure of our AM-LFS is summarized in Algorithm 1.

Algorithm 1 :AM-LFS

Input: Initialized model $M_{\boldsymbol{\omega}_0}$, initialized distribution $\boldsymbol{\mu}_0$, total training epochs T, distribution learning rate η

Output: Final model $M_{\boldsymbol{\omega}_T}$
for $t = 1$ **to** T **do**
  Sample B hyper-parameters $\boldsymbol{\theta}_1, ...\boldsymbol{\theta}_B$ via distribution $\mathcal{N}(\boldsymbol{\mu}_t, \sigma I)$;
  Train the model $M_{\boldsymbol{\omega}_t}$ for one epoch separately with the sampled hyper-parameters and get $M_{\boldsymbol{\omega}_t^1}, ...M_{\boldsymbol{\omega}_t^B}$;
  Calculate the score $R(\boldsymbol{\theta}_1), ...R(\boldsymbol{\theta}_B)$
  Decide the index of model with highest score $i = \arg\max_j R(\boldsymbol{\theta}_j)$;
  Update $\boldsymbol{\mu}_{t+1}$ using Eq. (20)
  Update $M_{\boldsymbol{\omega}_{t+1}} = M_{\boldsymbol{\omega}_t^i}$
end for
return $M_{\boldsymbol{\omega}_T}$

4. Experiments

We conducted experiments on four benchmarking datasets including CIFAR-10 [15], MegaFace [14], Market-1501 [44] and DukeMTMC-reID [29, 45] to show the effectiveness of our method on classification, face recognition and person re-identification tasks.

4.1. Implementation Details

Since our AM-LFS utilizes a bilevel optimization framework, our implementation settings can be divided into inner level and outer level. In the inner level, for fair comparison, we kept all experimental settings such as warmup stage, learning rate, mini-batch size and learning rate decay consistent with the corresponding original baseline methods on the specific tasks. Note that for the baseline methods with multi-branch loss, we only replace the softmax loss branch with AL-LFS while ignoring others. For example, in MGN [38], we only apply AL-LFS to 8 softmax loss branches while keeping triplet loss unchanged. In the outer level, we optimized the distribution of loss functions in the search space, the gradients of distribution parameters were computed by REINFORCE algorithm with rewards from a fixed number of samples. The rewards are top-1 accuracy, rank 1 value and mAP for classification, face recognition and person re-identification, perspectively. We normalized the rewards returned by each sample to zero mean and unit variance, which is set as the reward of each sample. For all the datasets, the numbers of samples B and intervals M are set to 32 and 6, respectively. Note that we also conducted the investigation on these numbers in Section 4.6. For the distribution parameters, we used Adam optimizer with a learning rate of 0.05 and set $\sigma = 0.2$. Having finished the update of the distribution parameters, we broadcast the model parameters with the highest mAP to each sample for synchronization. For all the datasets, each sampled model is trained with 2 Nvidia 1080TI GPUs, so a total of 64 GPUs are required.

4.2. Datasets

Classification: The CIFAR-10 dataset consist of natural images with resolution 32×32. CIFAR-10 consists of 60,000 images in 10 classes, with 6,000 images per class. The train and test sets contain 50,000 and 10,000 images respectively. On CIFAR datasets, we adopted a standard data augmentation scheme (shifting/mirroring) following [19, 13], and normalized the input data with channel means and standard deviations.

Face Recognition: For face recognition, we set the CASIA-Webface [43] as the training dataset and MegaFace as the testing dataset. CASIA-Webface contains 494,414 training images from 10,575 identities. MegaFace datasets are released as the largest public available testing benchmark, which aims at evaluating the performance of face recognition algorithms at the million scale of distractors. To perform open-set evaluations, we carefully remove the overlapped identities between training dataset and the testing dataset.

Person ReID: For person re-identification, we used Market-1501 and DukeMTMC-reID to evaluate our proposed AM-LFS method. Market-1501 includes images of 1,501 persons captured from 6 different cameras. The pedestrians are cropped with bounding-boxes predicted by DPM detector. The whole dataset is divided into training set with 12,936 images of 751 persons and testing set with 3,368 query images and 19,732 gallery images of 750 persons. In our experiments, we choose the single-query mode, where features are extracted from only 1 query image. DukeMTMC-reID is a subset of Duke-MTMC for person re-identification which contains 36,411 annotated bounding box images of 1,812 different identities captured by eight high-resolution cameras. A total of 1,404 identities are observed by at least two cameras, and the remaining 408 identities are distractors. The training set contains 16,522 images of 702 identities and the testing set contains the other 702 identities.

4.3. Results on CIFAR-10

We demonstrate our method with ResNet-20 [11] on the CIFAR-10 dataset [15]. We trained the model using the standard cross-entropy to obtain the original top-1 test error 8.75%. Table 2 shows the classification results compared with standard cross-entropy (CE) [4] and the dynamic loss by learning to teach (L2T-DLF) [41] methods. As can be observed, among all three loss function methods, our AM-LFS helps ResNet-20 achieve the best performance of 6.92% top-1 error rate. We also see that the recently proposed L2T-DLF reduces the error rate of softmax loss by 1.12% because L2T-DLF introduced the dynamic loss functions outputted via teacher model which can help to cultivate better student model. Note that AM-LFS can further reduce the top-1 error rate of L2T-DLF by 0.71%, which attributes to more effective loss function search space and optimization strategy design in our method.

Table 2. Results on the dataset CIFAR-10 using ResNet-20, showing the ratio of noise label, the top-1 test error rate (%) with standard cross entropy, L2T-DLF and AM-LFS.

Noise ratio(%)	CE [4]	L2T-DLF [41]	AM-LFS
0	8.75	7.63	**6.92**
10	12.05	—	**10.05**
20	15.05	—	**12.73**

In addition to the conventional experiments, we also conducted the CIFAR-10 noisy label experiments, where labeled classes can randomly flip to any other label by a given noise ratio, to demonstrate AM-LFS has the property of data re-weighting during training. As shown in Table 2, AM-LFS consistently outperforms the baseline softmax loss by 2.00% and 2.32% under the noise ratio of 10% and 20% respectively.

4.4. Results on MegaFace

We compare the proposed AM-LFS with three state-of-the-art loss function methods, including SphereFace, CosineFace and ArcFace. We used a modified ResNet with 20 layers that is adapted to face recognition and followed the same experimental settings with [37] when training the model parameters at the inner level. Table 3 shows the MegaFace rank1@1e6 performance with various loss functions. For SphereFace and CosineFace, we directly reported the results from the original paper. For ArcFace, we report their results by running the source codes provided by the authors to train the models by ourselves. We can observe that our proposed AM-LFS outperforms all compared methods by 6.1%, 1.0% and 1.1%, respectively. The main reason is that the candidates sampled from our proposed search space can well approximate all these compared loss functions, which means their good properties can be sufficiently explored and utilized during the training phase. Meanwhile, our optimization strategy enables that the dynamic loss can guide the model training of different epochs, which helps further boost the discrimination power.

4.5. Results on Market-1501 and DukeMTMC-reID

We demonstrate the effectiveness of our AM-LFS by applying it to some existing competitors including SphereReID, SFT and MGN. We compare with current state-of-the-art methods on both datasets to show our performance advantage over the existing baseline. We report the mean average precision (mAP) and the cumulative matching characteristics (CMC) at rank-1 and rank-5 on all the candidate datasets. On Market-1501 dataset, we only conduct experiments both in single-query mode. The results on Market-1501 dataset and DukeMTMC-reID dataset are

Table 3. Comparison with state-of-the-art loss functions on the MegaFace dataset using ResNet-20. For SphereFace and CosineFace, we directly reported the results from the original paper. For ArcFace, we report their results by running the source codes provided by their respective authors to train the models by ourselves following the same setting with CosineFace.

Method	MegaFace rank1@1e6
SphereFace [22]	67.4
CosineFace [37]	72.5
ArcFace [5]	72.4
AM-LFS	**73.5**

Table 4. Comparison with state-of-the-art methods on the Market-1501 dataset using ResNet 50 showing mAP, rank 1 and rank 5. RK refers to implementing re-ranking operation.

Methods	mAP	rank1	rank5
MLFN [2]	74.3	90.0	—
HA-CNN [18]	75.7	91.2	—
DuATM [32]	76.6	91.4	97.1
Part-aligned [34]	79.6	91.7	96.9
PCB [35]	77.4	92.3	97.2
SphereReID [6]	83.6	94.4	98.0
SFT [24]	82.7	93.4	—
MGN [38]	86.9	95.7	—
MGN(RK) [38]	94.2	**96.6**	—
SphereReID+ours	84.4	95.0	98.1
SFT+ours	83.2	93.6	97.9
MGN+ours	88.1	95.8	**98.4**
MGN(RK)+ours	**94.6**	96.1	**98.4**

shown in Table 4 and Table 5, respectively. We divide the results into two groups according to whether our AL-LFS is applied or not. For Market-1501, MGN(RK)+AM-LFS outperforms best competitor MGN(RK) by 0.4% in mAP. We observe that MGN rank 1 exhibits a degradation (0.5%) after adopting AM-LFS. The mainly reason is that AM-LFS utilized mAP-related reward for guidance, which may not always be consistent with the rank 1 value. For DukeMTMC-reID, MGN(RK)+AM-LFS surpass all compared methods in terms of mAP and rank 1. We conclude that although the baseline models SphereReID, SFT and MGN already achieved very high results on both Market-1501 and DukeMTMC-ReID, applying AM-LFS to them can still help cultivate better model hence boost the performance consistently.

4.6. Ablation Study

Effectiveness of the components: We showed the importance of both components **search space** and **search strategy** by demonstrating (1) the proposed design of the loss function search space itself can lead to AM-LFS's strong empirical performance and (2) the AM-LFS's capability of dynamically learning the distributions of better loss functions hence boost the performance during the training process. We trained the SphereReID model on Market-1501 by sampling candidates from the initial distribution while not updating this distribution. At the convergence, the model has the mAP of 84.0%, which outperforms the original baseline by 0.4%. We conducted this study for the proposed search space, which can guarantee a performance gain over the baseline model even under a guided random search setting. We further enabled the optimization of the distribution and obtained an additional performance gain of 0.4%. We therefore conclude that both the design of the loss function search space and the appropriate optimization procedures are crucial for good performance.

Table 5. Comparison with state-of-the-art methods on the DukeMTMC-ReID dataset using ResNet 50 showing mAP, rank 1 and rank 5. RK refers to implementing re-ranking operation.

Methods	mAP	rank1	rank5
PSE [30]	62.0	79.8	89.7
MLFN [2]	62.8	81.0	—
HA-CNN [18]	63.8	80.5	—
DuATM [32]	64.6	81.8	90.2
Part-aligned [34]	69.3	84.4	92.2
PCB+RPP [35]	69.2	83.3	—
ShpereReID [6]	68.5	83.9	90.6
SFT [24]	73.2	86.9	93.9
MGN [38]	78.4	88.7	—
MGN(RK) [38]	88.6	90.9	—
ShpereReID+ours	69.8	84.3	92.0
SFT+ours	73.8	87.0	95.1
MGN+ours	80.0	89.9	95.2
MGN(RK)+ours	**90.1**	**92.4**	**95.7**

Table 6. Effects of the number of samples by setting B as 4, 8, 16, 32 in terms of mAP on the Market-1501 dataset and DukeMTMC-reID dataset using our AM-LFS based on the SphereReID baseline model.

Method	B=4	B=8	B=16	B=32
Market-1501	83.6	83.8	84.2	84.4
DukeMTMC-reID	68.4	68.9	69.7	69.8

Investigation on samples: We study the effects of the number of samples in the optimization procedure by changing the parameter B in AM-LFS. Note that it costs more computation resources (GPUs) to train a minibatch of data as B increases. We report the performance results of different B values selected from $\{4, 8, 16, 32\}$ in Table. 6 in terms of mAP on the Market-1501 and DukeMTMC-reID based

Table 7. Effects of the number of intervals by setting M as 3, 6, 10 in terms of mAP on the Market-1501 dataset and DukeMTMC-reID dataset using our AM-LFS based on the SphereReID baseline model.

Dataset	M=3	M=6	M=10
Market-1501	83.8	84.4	84.2
DukeMTMC-reID	68.6	69.8	69.5

on SphereReID. The results show that when B is small, the performance degrades because efficient gradients cannot be obtained without enough samples. We also observe that the performance exhibits saturation when we keep enlarging B. For a tradeoff of the performance and the training efficiency, we choose to fix B as 32 during training.

Investigation on intervals: We study the effects of the the number of intervals in the search space by changing the interval parameter M in AM-LFS. According to our design of the search space, samples at different levels of difficulty are assigned to specific intervals, which enables a dynamic tradeoff between the intra-class distance and inter-class distance. Table. 7 shows mAP performance results with respect to M on the Market-1501 dataset and DukeMTMC-reID dataset. When we set the interval number as a small number ($M = 3$), the mAP exhibits a low value because the intervals are not enough to handle all levels of hard examples during training. The network is hard to train and suffers from an accuracy degradation when we set a large interval number ($M = 10$) because the excessive distribution parameters are hard to optimize. We also observe that the best performance is achieved at a moderate value of $M = 6$.

Investigation on convergency: To evaluate the training process of the our AM-LFS, we need to conduct investigation on the training convergency especially at the outer level. However, it's hard to simply track the loss convergency since our AM-LFS learns dynamic loss function distributions during the training process. Tracking the average reward is also not a good idea because this signal is very noisy which would create the illusion of instability of the training progress. The main reason is that small updates to loss function distributions at outer level may lead to large changes to the network parameters at the inner level. As a result, we choose to track a more intuitive metric, the distribution parameters to study the convergency of our method. From the Figure 3, we see that the distribution parameters tends to converge to specific values as the epochs increase, which indicates AM-LFS can be trained in a stable manner.

Visualization of gradient distribution: We visualize the gradient distribution of intra-class distance term f_{y_i} in Figure 4 to demonstrate AM-LFS that has more discrimination power than the baseline Sphere softmax loss function in SphereReID on Market-1501 dataset. When the activation value (x-axis) increases, the intra distance will decrease, where data samples are at a relatively easy level. On the contrary, when the activation value (x-axis) decreases, data samples are at a hard level. As can be observed from Figure 4, the gradients of AM-LFS with regard to hard examples are lower than those of baseline sphere softmax, which leads to a focus on the inter-class distance. We conclude that AM-LFS can dynamic tradeoff the significance of intra-class distance and inter-class distance hence boost the model's discrimination power.

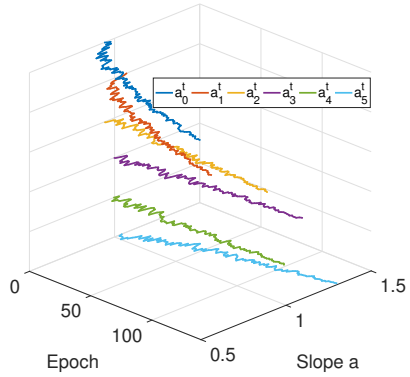

Figure 3. Convergency analysis of AM-LFS.

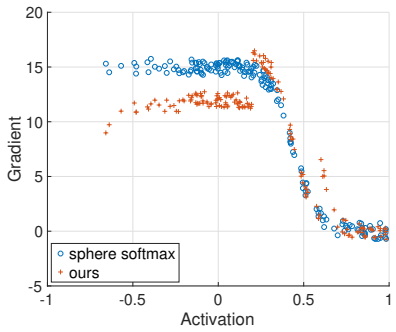

Figure 4. Visualization of gradient distribution of Sphere SoftmaxLoss and our AM-LFS.

5. Conclusion

In this paper, we have proposed AutoML for Loss Function Search (AM-LFS) which leverages REINFORCE to search loss functions during the training process. We carefully design an effective and task independent search space and bilevel optimization framework, which guarantees the generalization and transferability on different vision tasks. While this paper only demonstrates the effectiveness of AM-LFS applying to classification, face recognition and person re-id datasets, it can also be easily applied to other off-the-shelf modern computer vision frameworks for various tasks, which is an interesting future work.

References

[1] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *CoRR*, abs/1812.00332, 2018.

[2] Xiaobin Chang, Timothy M. Hospedales, and Tao Xiang. Multi-level factorisation net for person re-identification. In *CVPR*, pages 2109–2118, 2018.

[3] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals OR*, 153(1):235–256, 2007.

[4] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals OR*, 134(1):19–67, 2005.

[5] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *CoRR*, abs/1801.07698, 2018.

[6] Xing Fan, Wei Jiang, Hao Luo, and Mengjuan Fei. Spherereid: Deep hypersphere manifold embedding for person re-identification. *CoRR*, abs/1807.00537, 2018.

[7] Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Learning to teach. In *ICLR*, 2018.

[8] Ross B. Girshick. Fast R-CNN. In *ICCV*, pages 1440–1448, 2015.

[9] Minghao Guo, Zhao Zhong, Wei Wu, Dahua Lin, and Junjie Yan. Irlas: Inverse reinforcement learning for architecture search. *arXiv preprint arXiv:1812.05285*, 2018.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[12] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. AMC: automl for model compression and acceleration on mobile devices. In *ECCV*, pages 815–832, 2018.

[13] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *ECCV*, pages 646–661, 2016.

[14] Ira Kemelmacher-Shlizerman, Steven M. Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, pages 4873–4882, 2016.

[15] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[17] Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. *CoRR*, abs/1811.05181, 2018.

[18] Wei Li, Xiatian Zhu, and Shaogang Gong. Harmonious attention network for person re-identification. In *CVPR*, pages 2285–2294, 2018.

[19] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2013.

[20] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2999–3007, 2017.

[21] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[22] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, pages 6738–6746, 2017.

[23] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, pages 507–516, 2016.

[24] Chuanchen Luo, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Spectral feature transformation for person re-identification. *CoRR*, abs/1811.11405, 2018.

[25] Renqian Luo, Fei Tian, Tao Qin, Enhong Chen, and Tie-Yan Liu. Neural architecture optimization. In *Advances in Neural Information Processing Systems*, pages 7827–7838, 2018.

[26] Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*, 2018.

[27] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. *arXiv preprint arXiv:1802.01548*, 2018.

[28] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NeurIPS*, pages 91–99, 2015.

[29] Ergys Ristani, Francesco Solera, Roger S. Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV*, pages 17–35, 2016.

[30] M. Saquib Sarfraz, Arne Schumann, Andreas Eberle, and Rainer Stiefelhagen. A pose-sensitive embedding for person re-identification with expanded cross neighborhood reranking. In *CVPR*, pages 420–429, 2018.

[31] Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In *NIPS*, pages 4053–4061, 2016.

[32] Jianlou Si, Honggang Zhang, Chun-Guang Li, Jason Kuen, Xiangfei Kong, Alex C. Kot, and Gang Wang. Dual attention matching network for context-aware feature sequence based person re-identification. In *CVPR*, pages 5363–5372, 2018.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[34] Yumin Suh, Jingdong Wang, Siyu Tang, Tao Mei, and Kyoung Mu Lee. Part-aligned bilinear representations for person re-identification. In *ECCV*, pages 418–437, 2018.

[35] Yifan Sun, Liang Zheng, Yi Yang, Qi Tian, and Shengjin Wang. Beyond part models: Person retrieval with refined part pooling (and A strong convolutional baseline). In *ECCV*, pages 501–518, 2018.

[36] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. *arXiv preprint arXiv:1807.11626*, 2018.

[37] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Process. Lett.*, 25(7):926–930, 2018.

[38] Guanshuo Wang, Yufeng Yuan, Xiong Chen, Jiwei Li, and Xi Zhou. Learning discriminative features with multiple granularities for person re-identification. In *2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, October 22-26, 2018*, pages 274–282, 2018.

[39] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, pages 5265–5274, 2018.

[40] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

[41] Lijun Wu, Fei Tian, Yingce Xia, Yang Fan, Tao Qin, Jian-Huang Lai, and Tie-Yan Liu. Learning to teach with dynamic loss functions. In *NeurIPS*, pages 6467–6478, 2018.

[42] Lingxi Xie and Alan L Yuille. Genetic cnn. In *ICCV*, pages 1388–1397, 2017.

[43] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Learning face representation from scratch. *CoRR*, abs/1411.7923, 2014.

[44] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *ICCV*, pages 1116–1124, 2015.

[45] Zhedong Zheng, Liang Zheng, and Yi Yang. Unlabeled samples generated by GAN improve the person re-identification baseline in vitro. In *ICCV*, pages 3774–3782, 2017.

[46] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *CVPR*, 2018.

[47] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[48] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. *arXiv preprint arXiv:1707.07012*, 2(6), 2017.