Differential neural ensemble search with diversity control

A Preprint

P. Babkin, K. Yakovlev, K. Petrushina, O.Bakhteev,

Abstract

In our research, we investigate a novel method for sampling deep learning models using a hypernetwork. The hypernetwork is a neural network that controls the diversity of the models by translating a real number representing ensemble diversity into a sampled neural network architecture. Architectures are obtained in a one-shot manner by perturbing from the base architecture in terms of the Jenson-Shannon divergence (JSd). We evaluate the performance of the proposed algorithm by conducting experiments on the Fashion-MNIST and CIFAR-10 datasets, comparing the resulting ensembles with those sampled by other searching algorithms.

Keywords differential search · neural ensembles · hypernetwork · diversity control

1 Introduction

Nowadays methods of neural architecture search (NAS) are well-explored and proved to be an effective way of creating more effective and efficient neural networks [2, 10, 4]. Some of these methods use different ways to smooth out the architecture so optimum for it can be found by wide range of methods for smooth optimization problems. On the other hand, neural ensemble search (NES) is modern and not as well investigated problem as NAS, although it is known that ensembles of deep learning models show better results in different applied problems [3]. Our paper investigates an method of sampling deep learning models in a new way that gives compatible results and has its own sphere of implementations.

Our method takes the result of NAS as a base architecture than it samples architectures which are close to the optimal one in terms of Jensen-Shannon divergence (JSd). Architectures differ in terms of λ, idea of method is shown in Fig. 1. Basic architecture is gained with $\lambda = \lambda_1$. Blue ellipses are equidistant surfaces of architectures. Staring with diverse parameter $\lambda = \lambda_2$ resulting architecture performs unacceptable accuracy, so architectures beyond the surface are not included into ensemble. Method can control whether sampled architectures are close enough to the optimal one so they perform good accuracy on the original dataset and are diverse enough so every architecture makes its own contribution to the final answer.

To sample architectures we use hypernetwork [1]. This network generates parameters for another network, which is called target network. Previously hypernetwoks were intended to control different characteristics such as complexity of architecture [8] or parameters of the target model [7] in several modern papers. In our paper it controls diversity of the target models, so every sampled model differs from previously sampled ones in terms of JSd.

The hypernetwork uses JSd to measure difference between two architectures which is symmetric and finite in contrast to more popular Kullback–Leibler divergence. Our main idea of sampling different model is to use a regularizer, based on JSd as a source of diversity.

This way we sample deep learning models in one-shot unlike NES for Uncertainty Estimation [9], where every sampled models must be tested and least appropriate architectures are excluded from ensemble. In this way our method is similar to NES via Bayesian Sampling [6], but we sample architectures using different approach, posterior distribution for architectures is gained based on the optimal one. To sum up the scheme of our method: find a base architecture using DARTS, sample architectures in one-shot via differentiable algorithm. Inference answer is ensemble of the sampled deep learning models.

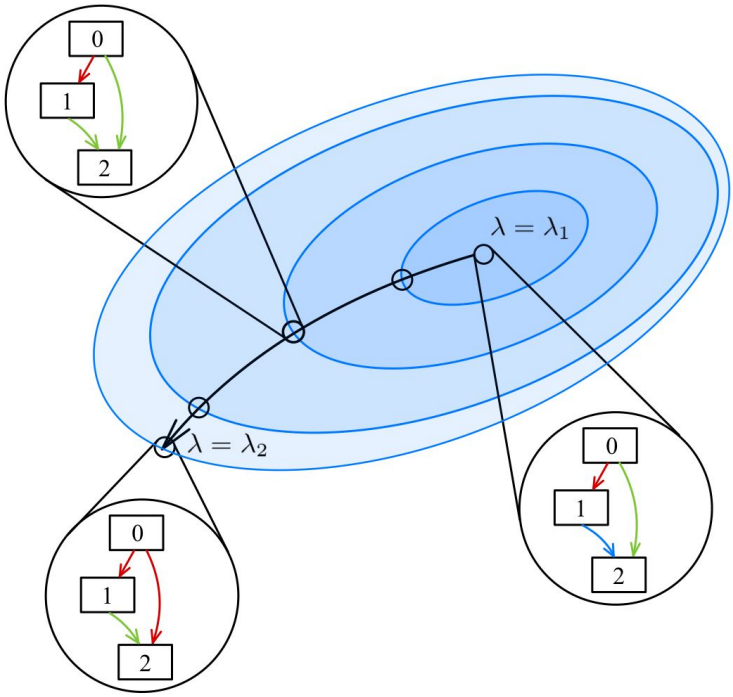

Рис. 1: Architecture space with JSd metrics. Blue ellipses are equidistant surfaces. The space is continuous but has varying preferred operations across different architectural levels. Architectures with different operations are pictured separately on the picture.

We conduct experiments on CIFAR and MNIST datasets to evaluate performance of the proposed method in terms of accuracy and time. Also we compare the performance with state-of-art NES algorithms [9, 6].

2 Problem statement

In our paper, we address the problem of classification. We assume that the dataset $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ is given, where $\mathbf{X}$ represents the feature matrix and $\mathbf{y}$) represents the target vector. The dataset is divided into training and validation subsets, for which the loss functions $\mathcal{L}_{train}$ and $\mathcal{L}_{val}$ are specified, respectively. In general, these functions depend on the feature matrix, target vector, and the predicted outputs of the training model. However, in our paper, we will not explicitly state their dependence on the dataset, assuming it implicitly. Therefore, we will state that the loss functions depend only on the predicted outputs of the model.

Contrary to the selection of one single architecture in conventional NAS algorithm [2, 5, 11], this paper focuses on the problem of selecting a well-performing neural network ensemble with diverse architectures from the NAS search space, i.e., neural ensemble search (NES).

In order to facilitate our study, several key terms are defined. The architecture of a model, i.e. a set of node operations, is denoted by $\boldsymbol{\alpha}$. For a fixed architecture, optimal parameters are denoted by $\boldsymbol{w}^*_{\boldsymbol{\alpha}}$. The optimal architecture resulting from neural architecture search (NAS) is denoted as $\boldsymbol{\alpha}^*$. To quantify architectural diversity, we define λ, a real number ranging from 0 to Λ. The corresponding architecture for a given λ is denoted by $\boldsymbol{\alpha}(\lambda)$, which can be obtained by solving a specific problem. The output of an architecture $\boldsymbol{\alpha}$, given its corresponding model parameters $\boldsymbol{w}_{\boldsymbol{\alpha}}$, is denoted by $f(\boldsymbol{w}_{\boldsymbol{\alpha}}, \boldsymbol{\alpha})$. Finally, the set of architectures included in the ensemble is denoted by $\mathcal{S}$.

Given the ensemble scheme, NES can be formally framed as

$$\min_{S} \mathcal{L}_{val}\left(\frac{1}{|S|}\sum_{\boldsymbol{\alpha}\in S} f(\boldsymbol{w}^*_{\boldsymbol{\alpha}}, \boldsymbol{\alpha})\right)$$
$$s.t.\ \forall \boldsymbol{\alpha} \in S\ \boldsymbol{w}^*_{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{w}} \mathcal{L}_{train}(f(\boldsymbol{w}^*_{\boldsymbol{\alpha}}, \boldsymbol{\alpha}))$$

3 Method

Below we briefly described our method for solving problem of classification with dataset $\mathcal{D} = (\mathbf{X}, \mathbf{y})$. For each object $\mathbf{x} \in \mathbf{X}$ there is a label $y \in \mathbf{y}$. We solve the problem via NES, sampling architectures according to novel methodology which is formally described below.

3.1 Architecture

Let $\mathcal{V} = \{1, \dots, N\}$ be a set of vertices, where N is a number of vertices, and $\mathcal{E} = \{(i, j) \in V \times V \mid i < j\}$ a set of edges between them. A set of possible operations $\mathcal{O}$ usually contains pooling, convolutions, etc. For each edge there is an operation $o \in \mathcal{O}$ that transits information from one node to another.

NAS algorithms search for optimal architecture. As it was mentioned below, architecture of neural network is a set of operations between nodes. In NAS methods architecture is a vector constructed by following rules. For each edge $(i, j) \in \mathcal{E}$, $\boldsymbol{\alpha}^{(i,j)}$ is a vector, which assigns impact of each operation. $\boldsymbol{\alpha}$ is a concatenation of all structural parameters vectors $\boldsymbol{\alpha}^{(i,j)}$.

3.2 Reguralizer and heuristic

Source of diversity in out method is a regularizer with diversity parameter λ. By subtracting it from a loss function we promote method for finding an architecture which differs from the optimal one.

We use Jensen-Shannon divergence to measure diversity of two architectures. NES algorithms give discrete architecture as a result of their work, i.e. initially optimal architecture $\boldsymbol{\alpha}^*_{\boldsymbol{init}}$ is discrete, so JSd cannot be calculated with $\boldsymbol{\alpha}^*_{\boldsymbol{init}}$ as an argument. We smooth the architecture using smooth parameter τ to solve this problem.

$$\boldsymbol{\alpha}^* = (1 - \tau)\boldsymbol{\alpha}^*_{\boldsymbol{init}} + \tau\frac{1}{|\mathcal{O}|}$$

Assuming τ close to one we obtain a smoothed architecture that contains the same information about architecture as a initial one, but we also can use it in JSd.

3.3 Diversity control via hypernetwork

In order to control diversity we employ the concept of hypernetwork. A hypernetwork is a parametric mapping from $[0, \Lambda]$ to the set of model structural parameters [8].

$$\boldsymbol{\alpha} : [0, \Lambda], \mathbb{R}^u \to \mathbb{R}^s$$

Where $\mathbb{R}^u$ is a hypernetwork parametric space and $\mathbb{R}^s$ is space of model structural parameters. In this terms $\boldsymbol{\alpha}$ can be redefined using hypernetwork

$$\boldsymbol{\alpha} = \boldsymbol{\alpha}(\lambda, \boldsymbol{a}) \text{ or } \boldsymbol{\alpha}^{(i,j)} = \boldsymbol{\alpha}^{(i,j)}(\lambda, \boldsymbol{a}^{(i,j)})$$

In this paper each function $\boldsymbol{\alpha}^{(i,j)}$ is a piecewise linear function

$$\boldsymbol{\alpha}^{(i,j)}(\lambda, \boldsymbol{a}^{(i,j)}) = \sum_{k=0}^{N-1}\left(\frac{\lambda - r_k}{r_{k+1} - r_k}\boldsymbol{a}^{(i,j)}_{\boldsymbol{k}} + \left(1 - \frac{\lambda - r_k}{r_{k+1} - r_k}\right)\boldsymbol{a}^{(i,j)}_{\boldsymbol{k+1}}\right) I[\lambda \in [r_k, r_{k+1}]]$$

In our method λ is sampled from predefined distribution $p(\lambda) = U(0, \lambda)$ and new architecture is obtained from hypernetwork.

3.4 Final statement

Upon concluding the method description, a reformulated problem can be postulated: specifically, architectures are obtained via hypernetwork, and the regularizer is subtracted from the loss function. A varied parameter λ, ranging from 0 to Λ, is distributed according to a uniform distribution.

$$\min_{\boldsymbol{\alpha}} \mathbb{E}_{\lambda \sim U(0,\Lambda)}[\mathcal{L}_{val}(\boldsymbol{w}^*, \boldsymbol{\alpha}(\lambda)) - \lambda JS(\boldsymbol{\alpha}^*, \boldsymbol{\alpha}(\lambda))]$$
$$s.t.\ \boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \mathbb{E}_{\lambda \sim U(0,\Lambda)}[\mathcal{L}_{train}(\boldsymbol{w}, \boldsymbol{\alpha}(\lambda))]$$

Following problem resolution, a new architecture is acquired and subsequently included into the ensemble. It is assured that said architecture is diverse enough from its counterparts and performs an acceptable level of accuracy.

4 Computational experiment

Main goal of basic experiment is to investigate dependence of resulting architecture's performance on λ and also to estimate suitable Λ.

We conducted experiments on fashionMNIST dataset for $\lambda = 1, \ldots, 40$ and looked at resulting architecture and accuracy. Then we searched for appropriate Λ by the following algorithm: we chose three random values of λ from 0 to Λ, ensembled corresponding architectures and looked and ensemble performance in terms of accuracy on the dataset.

4.1 Preliminary report

Obtained results of architectures investigation are depicted in the table below Tab. 1. Maximal accuracy of sampled architectures is written down. Also we recorded amount of matched operations between nodes i.e. number of edges between particular nodes that contain the same operation in obtained architecture and the basic one. Graphical visualization can be found in Fig. 2.

Although the graphic is unstable, deterioration of accuracy with growth of architecture diversity parameter λ can be clearly seen.

This results allow us to choose $\Lambda = 32$. However amount of matching edges does not really coincide with our expectations. Architectures almost do not intersect. It can be explained by the fact that the optimized function is not convex, so algorithm fall into different minimums, however we can get more matching edges by using negative λ.

Таблица 1: preliminary results

λ	accuracy, %	matched operations
0 (optimum)	91.24	-
1/2	91.24	4
2	91.26	3
32	89.21	2
64	87.09	1

5 Conclusion

In this paper, we proposed a novel method for sampling ensembles of deep learning models with diversity control. Our method utilizes a hypernetwork to generate diverse architectures by perturbing a base architecture in terms of Jensen-Shannon divergence. The diversity of the ensemble is controlled by a penalty term added to the loss function, which encourages the ensemble members to be diverse. We conducted experiments on the Fashion-MNIST dataset and demonstrated that our method performs compatible results in terms of accuracy and architectural diversity. Overall, our proposed method shows potential for practical applications in deep learning ensembling.

Список литературы

[1] D. Ha, A. Dai, and Q. V. Le. Hypernetworks. 2016.

[2] H. Liu, K. Simonyan, and Y. Yang. DARTS: Differentiable architecture search. In Proc. ICLR, 2019.

[3] A. R. Narayanan, A. Zela, T. Saikia, T. Brox, and F. Hutter. Multi-headed neural ensemble search. 2021.

[4] N. Nayman, A. Noy, T. Ridnik, I. Friedman, R. Jin, and L. Zelnik-Manor. XNAS: Neural architecture search with expert advice. In Proc. NeurIPS, pages 1975–1985, 2019.

[5] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean. Efficient neural architecture search via parameter sharing. In Proc. ICML, pages 4092–4101, 2018.

[6] Y. Shu, Y. Chen, Z. Dai, B. Kian, and H. Low. Neural ensemble search via bayesian sampling. 2022.

[7] J. von Oswald, C. Henning, B. F. Grewe, and J. Sacramento. Continual learning with hypernetworks. 2022.

[8] K. D. Yakovlev, O. S. Grebenkova, O. Y. Bakhteev, and V. V. Strijov. Neural architecture search with structure complexity control. 2022.

[9] S. Zaidi, A. Zela, T. Elsken, C. C. Holmes, F. Hutter, and Y. W. Teh. Neural ensemble search for uncertainty estimation and dataset shift. In Proc. NeurIPS, 2022.

[10] A. Zela, T. Elsken, T. Saikia, Y. Marrakchi, T. Brox, and F. Hutter. Understanding and robustifying differentiable architecture search. In Proc. ICLR, 2020.

[11] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. In Proc. ICLR, 2017.