
Differentiable algorithm for searching ensembles of deep learning models with diversity control

A Preprint

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

Abstract

This paper is developed to introduce a new method of creating ensembles of deep learning models. Many modern researches were focused on creating effective and efficient algorithms of differentiable architecture search, missing opportunity to create ensembles of deep learning models. This approach gives impressive results as it was shown in few modern papers. In our research we investigate an algorithm of sampling deep learning models using hypernetwork, which controls diversity of the models. This method allows us to sample deep learning models in one-shot, without any additional calculations losses, this independence means that algorithm is able to sample as many models as we want. To evaluate the performance of the proposed algorithm, we conducted experiments on the Fashion-MNIST and CIFAR-10 datasets and compare the resulting ensembles with ones sampled by other searching algorithms.

Keywords differential search · neural ensembles · hypernetwork · diversity control

1 Introduction

Nowadays algorithms of neural architecture search (NAS) are well-explored and proved to be an effective way of creating more effective and efficient neural networks, [5, 9, 6]. Such algorithms use different ways to make problem differentiable so it can be solved 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.

Our paper investigates an algorithm of sampling deep learning models in a new way. Despite the fact that similar algorithms were described and investigated, our scheme has unique set of techniques that gives compatible results and has its own sphere of implementations.

First of all, we use DARTS [5] as our basic NAS algorithm. It is proved to be an effective architecture search algorithm. Some modern investigations have shown that the algorithm can be upgraded [2, 1, 3], but the modernisation do not make tangible difference and are mainly made for some specific cases. So in our paper we are focusing on ensemble sampling, so slightly different base model does not change anything in resulting ensemble.

Second of all, we use hypernetwork in our sampling algorithm. Hypernetwork is small network so it does not consume too much computational capacity [4]. This network contains information about another network, which is called target network. This approach is not new: hypernetworks and supernetworks were intended to control different characteristics such as complexity of architecture [8] or parameters of the target model [7] in several modern investigations. In our algorithm it controls diversity of the target models, so every sampled model differs from previously sampled ones.

The hypernetwork uses Jensen-Shannon divergence (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 are able to sample deep learning models in one-shot, without and additional computational losses. To sum up the scheme of our method:

1. find a base architectire using DARTS
2. sample architectures in one-shot via differentiable algorithm
3. inferencing answer is ensemble of the sampled deep learning models

2 Method

See Section 2.

2.1 Problem statement

Contrary to the selection of one single architecture in conventional NAS algorithm, 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). Let $f(w_\alpha, \alpha)$ denote the output of an architecture α and model parameter w_α , S be a set of architectures and \mathcal{L}_{train} and \mathcal{L}_{val} denote the training and validation losses, respectively. Given the ensemble scheme with an ensemble size of $|S|$, NES can be formally framed as

$$\begin{aligned} \min_S \mathcal{L}_{val} \left(\frac{1}{|S|} \sum_{\alpha \in S} f(w_\alpha^*, \alpha) \right) \\ s.t. \forall \alpha \in S \ w_\alpha^* = \arg \min_w \mathcal{L}_{train}(f(w_\alpha, \alpha)) \end{aligned}$$

We slightly rearranged the problem: in our case w^* is general for all architectures. Also architectures differ in terms of λ so resulting functions can be calculated in terms of expected values.

$$\begin{aligned} \min_{\lambda \in \Lambda} \mathbb{E}_\lambda[\mathcal{L}_{val}(w^*, \alpha(\lambda))] \\ s.t. \ w^* = \arg \min_w \mathbb{E}_\lambda[\mathcal{L}_{train}(w, \alpha(\lambda))] \end{aligned}$$

2.1.1 Headings: third level

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Paragraph Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

3 Examples of citations, figures, tables, references

3.1 Citations

Citations use `natbib`. The documentation may be found at

<http://mirrors.ctan.org/macros/latex/contrib/natbib/natnotes.pdf>

Рис. 1: Sample figure caption.

Таблица 1: Sample table title

Part		
Name	Description	Size (μm)
Dendrite	Input terminal	~ 100
Axon	Output terminal	~ 10
Soma	Cell body	up to 10^6

3.2 Figures

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetur odio sem sed wisi. See Figure 1. Here is how you add footnotes. ¹ Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetur eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

3.3 Tables

See awesome Table 1.

The documentation for `booktabs` (‘Publication quality tables in LaTeX’) is available from:

<https://www.ctan.org/pkg/booktabs>

3.4 Lists

- Lorem ipsum dolor sit amet
- consectetur adipiscing elit.
- Aliquam dignissim blandit est, in dictum tortor gravida eget. In ac rutrum magna.

Список литературы

- [1] X. Chen and C. Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In Proc. ICML, pages 1554–1565, 2020.
- [2] X. Chen, L. Xie, J. Wu, and Q. Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In Proc. ICCV, pages 1294–1303, 2019.
- [3] X. Chu, X. Wang, B. Zhang, S. Lu, X. Wei, and J. Yan. DARTS-: Robustly stepping out of performance collapse without indicators. arXiv:2009.01027, 2020.
- [4] D. Ha, A. Dai, and Q. V. Le. Hypernetworks. 2016.
- [5] H. Liu, K. Simonyan, and Y. Yang. DARTS: Differentiable architecture search. In Proc. ICLR, 2019.
- [6] 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.
- [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.

¹Sample of the first footnote.

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