Оптимизация архитектуры нейросети с контролем эксплуатационных характеристик на целевом устройстве

Фирсов Сергей

Научный руководитель: к.ф.-м.н. О.Ю. Бахтеев

Московский физико-технический институт

2025

Цель исследования

- **Neural Architecture Search:** Задача автоматизированного поиска оптимальной архитектуры нейросети.
- **Цель:** Получать архитектуры решающие поставленную ML задачу, при этом удовлетворяя вычислительным или ресурсным ограничениям.
- **Проблемы:**
  - Обширное пространство для поиска
  - Баланс точности и сложности
  - Получение семейства решений

Постановка задачи

- Многоклассовая классификация. Выборка $\mathfrak{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{\mathcal{N}}, \ \boldsymbol{x_i} \in \mathsf{X}, \ y_i \in \mathsf{Y}$.

- Модель — нейронная сеть из повторяющихся блоков, каждый из которых представляется в виде DAG[1]. Задача выбирать рёбра-операции $\boldsymbol{o}^{(m)}$ для каждой пары вершин в блоке.

- Для релаксации дискретной задачи выбора архитектуры в непрерывную, вводится *смешанная операция*

$$\widehat{\boldsymbol{g}}^{(i,j)}(\boldsymbol{x}) \ = \ \sum_{m=1}^{k} \gamma_m^{(i,j)} \, \boldsymbol{o}^{(m)}(\boldsymbol{x}), \ \ \gamma^{(i,j)} \sim \ \mathrm{GumbelSoftmax}\big(\boldsymbol{\alpha}^{(i,j)}, \, t\big).$$

[1] Liu, Simonyan и Yang, 2019: «DARTS: Differentiable architecture search»

Регуляризация сложности

Вводится *векторный параметр сложности*[2] $\boldsymbol{s}$, компоненты которого — коэффициенты регуляризации по соответствующим операциям.

$$\boldsymbol{s} \in \Delta^{k-1}, \quad \Delta^{k-1} \;=\; \Big\{ \boldsymbol{s} \in \boldsymbol{R}^k \mid \sum_{m=1}^{k} s_m = 1,\ s_m \geq 0 \Big\},$$

где $k = |\mathcal{O}|$, Δ^{k-1} *симплекс* размерности $k-1$.

Определим функцию затрат Cost архитектуры γ на основе $\boldsymbol{s}$:

$$Cost\big(\gamma;\, \boldsymbol{s}\big) \;=\; \sum_{(i,j)\in E} \sum_{m=1}^{k} \gamma_m^{(i,j)}\, s_m.$$

[2]Yakovlev и др., 2022: «Neural architecture search with structure complexity control»

Теорема

Theorem (Фирсов, 2025)

Пусть для каждого набора весов $\boldsymbol{w} \in \mathbb{R}^n$ *и любого входа* $\boldsymbol{x} \in \mathcal{X}$ *функция*

$$f(\boldsymbol{\gamma}) \;=\; \mathcal{L}_{\text{task}}\big(\boldsymbol{w}, \boldsymbol{\gamma}\big) \;+\; \kappa \,\langle \boldsymbol{\gamma}, \boldsymbol{s} \rangle, \qquad \boldsymbol{s} \in \Delta^{k-1},$$

непрерывна и ограничена на симплексе $\Delta^{k-1} = \big\{\boldsymbol{\gamma} \in \mathbb{R}^k \mid \sum_{i=1}^{k} \gamma_i = 1,\ \gamma_i \geq 0\big\}$. *Положим*

$$\boldsymbol{\gamma}(t) \sim \mathrm{GumbelSoftmax}(\boldsymbol{\alpha}, t), \qquad \bar{\boldsymbol{\gamma}}(t) = \mathrm{Softmax}\big(\boldsymbol{\alpha}/t\big) = \mathbb{E}\big[\boldsymbol{\gamma}(t)\big].$$

Тогда

$$\lim_{t \to 0^+} \mathbb{E}\big[f\left(\boldsymbol{\gamma}(t)\right)\big] \quad = \quad f\big(\bar{\boldsymbol{\gamma}}(0)\big).$$

Оптимизационная задача

Для генерации архитектуры вводится гиперсеть $u_{\boldsymbol{a}} \colon \Delta^{k-1} \to \Gamma$ с параметрами $\boldsymbol{a}$, которая по заданному вектору сложности $\boldsymbol{s} \in \Delta^{k-1}$ порождает логиты $\boldsymbol{\alpha}$, тем самым определяя архитектуру модели с учётом эксплуатационных характеристик.

Оптимизационная задача:

$$\boxed{\mathbb{E}_{\boldsymbol{s} \sim U(\Delta^{k-1})} \left[\mathbb{E}_{\gamma \sim \mathrm{GS}(\boldsymbol{\alpha}(\boldsymbol{s}), t)} \mathcal{L}_{\text{task}}(\boldsymbol{w}, \gamma) + \kappa\, Cost\left(\bar{\gamma}(\boldsymbol{s}); \boldsymbol{s}\right) \right] \to \min_{\boldsymbol{w}, \boldsymbol{a}}}$$

Иллюстрация

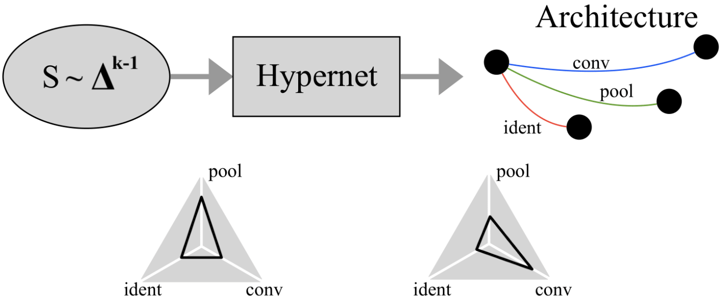


Иллюстрация предлагаемого метода.

- Предлагается использовать векторный параметр сложности $\boldsymbol{s}$, компоненты которого — коэффициенты регуляризации по соответствующим операциям.
- Гиперсеть на основе $\boldsymbol{s}$ генерирует архитектурные параметры для нейросети.

Постановка эксперимента

- Эксперименты поставлены для подтверждения работоспособности метода и иллюстрации его возможностей.
- Представленный эксперимент проводится на выборке Fashion-MNIST.
- Рассматриваются модели из 3 блоков по 4 вершины. Доступные операции 3×3 convolution, 3×3 max-pooling и identity (передача данных без изменений).
- Параметры $\boldsymbol{w}$ и параметры гиперсети $\boldsymbol{a}$ тренировались с помощью оптимизаторов Adam.
- Температура Gumbel–Softmax линейно уменьшалась от t=1.0 до t=0.2 по эпохам.

Получение моделей

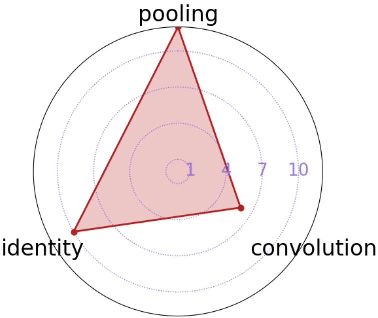
Получено **семейство** нейронных сетей, из которого для анализа выделяется четыре архитектуры, соответствующие приведённым ниже векторам сложности $\boldsymbol{s} = (pool, ident, conv)$. Компоненты вектора являются штрафами за соответствующие операции.

$$\text{(A)}\ (0.33, 0.33, 0.33), \quad \text{(B)}\ (0.15, 0.15, 0.70),$$

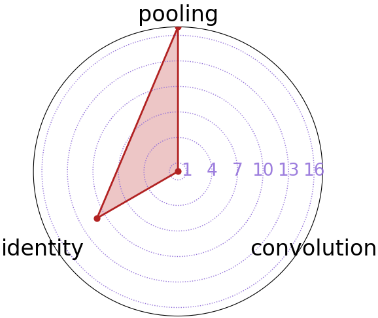
$$\text{(C)}\ (0.70, 0.15, 0.15), \quad \text{(D)}\ (0.15, 0.70, 0.15).$$

Представленные архитектуры отражают равномерный штраф за операции (A), увеличенный за свёртки (B), увеличенный за пулинг (C) и увеличенный за пропуск (D).

Распределение операций



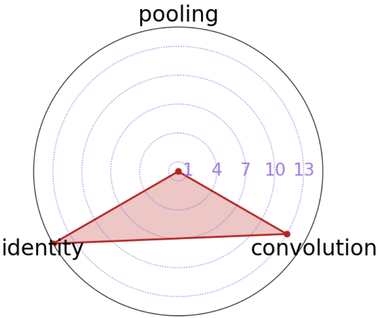
(A) равномерный штраф



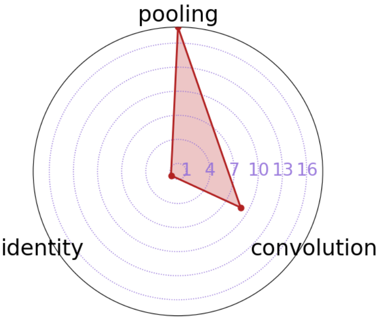
(B) увеличенный за свёртки

Статистика по операциям для архитектур эксперимента 1.

Распределение операций



(C) увеличенный за пулинг



(D) увеличенный за пропуск

Статистика по операциям для архитектур эксперимента 1.

Результаты

Модели	A	B	C	D
Accuracy	82.5%	79.2%	**85.0%**	81.4
Количество параметров	38 304	**5 120**	**143 488**	12 320
Количество pooling	12	17	**0**	18
Количество convolution	6	**0**	13	9
Количество identity	10	11	15	**1**

Сравнение параметров полученных архитектур. **(A)** равномерный штраф, **(B)** увеличенный за свёртки, **(C)** увеличенный за пулинг, **(D)** увеличенный за пропуск.

Результаты подтверждают корректность работы предлагаемого метода. При увеличении штрафа за операцию — уменьшается её использование в модели. Регуляризация позволяет учитывать различные эксплуатационные ограничения без необходимости дообучения модели на них.

Выносится на защиту

- Предложен метод позволяющий получать семейство моделей с возможностью контроля сложности обучения и аппаратных ограничения.
- Метод обладает возможностью получать архитектуры моделей за счёт изменения вектора параметра сложности без необходимости дополнительного дообучения.
- Вычислительные эксперименты подтверждают работоспособность метода и демонстрируют заявленную гибкость получаемого решения.

Результаты работы доложены на конференции МФТИ и в университете Сириус.
В рамках дальнейшего развития метода планируется добавление многокритериального учета сложности и проведение полноформатного сопоставления результатов с классическими NAS-подходами.

Список литературы

[1] H. Liu, K. Simonyan и Y. Yang. «DARTS: Differentiable architecture search». В: *International Conference on Learning Representations*. 2019.

[2] K. Yakovlev и др. «Neural architecture search with structure complexity control». В: *Recent Trends in Analysis of Images, Social Networks and Texts*. 2022, с. 207—219.

[3] J. Dong, Y. Yang и … «BigNAS: Scaling Up Neural Architecture Search with Big Single Stage Models». В: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, с. 3173—3182.

[4] T. Elsken, J. H. Metzen и F. Hutter. «Neural Architecture Search: A Survey». В: *Journal of Machine Learning Research* 20.55 (2019), с. 1—21.

[5] H. Cai, L. Zhu и S. Han. *ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware*. 2019. arXiv: 1812.00332 [cs.LG].

Примечание. Клетка

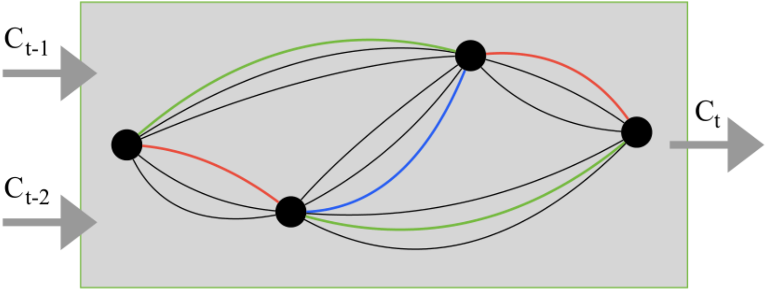


Иллюстрация клетки.

Примечание. Гиперсеть

Введём *гиперсеть* $\boldsymbol{u_a}$ с параметрами $\boldsymbol{a}$, которая на основе $\boldsymbol{s}$ строит логиты $\boldsymbol{\alpha}^{(i,j)}$ для каждого ребра (i,j) как взвешенную комбинацию заранее выученных логитов опорных архитектур $\boldsymbol{\alpha}_{\mathrm{ref},r}^{(i,j)}$, полученных в процессе оптимизации гиперсети. По вектору сложности $\boldsymbol{s}$ вычисляются веса интерполяции

$$w_r(\boldsymbol{s}) = \frac{\exp\left(-\|\boldsymbol{s} - \boldsymbol{s}_{\mathrm{ref},r}\|^2\right)}{\sum_{l=1}^{m} \exp\left(-\|\boldsymbol{s} - \boldsymbol{s}_{\mathrm{ref},l}\|^2\right)}, \quad r = 1, \ldots, m,$$

где $\boldsymbol{s}_{\mathrm{ref},r} \in \Delta^{k-1}$ — r-я опорная точка, а m — их общее число. Затем итоговые логиты для ребра (i,j) вычисляются как

$$\boldsymbol{\alpha}^{(i,j)}(\boldsymbol{s}) \; = \; \sum_{r=1}^{m} w_r(\boldsymbol{s}) \, \boldsymbol{\alpha}_{\mathrm{ref},r}^{(i,j)}.$$

Примечание. Оптимизационная задача

$$\mathbb{E}_{\boldsymbol{s}\sim U(\Delta^{k-1})}\left[\mathbb{E}_{\gamma\sim\mathrm{GS}(\boldsymbol{\alpha}(\boldsymbol{s}),t)}\mathcal{L}_{\mathsf{task}}(\boldsymbol{w},\gamma)+\kappa\,\mathit{Cost}\big(\bar{\gamma}(\boldsymbol{s});\boldsymbol{s}\big)\right]\to\min_{\boldsymbol{w},\boldsymbol{a}},$$

где $\boldsymbol{w}$ — все параметры внутри примитивных операций $\mathrm{o}^{(m)}$; $\boldsymbol{a}$ — параметры гиперсети $\boldsymbol{u}_{\boldsymbol{a}}$, которая генерирует логиты $\boldsymbol{\alpha}^{(i,j)}$ для всех $(i,j)\in E$ при подстановке $\boldsymbol{s}$;
$\boldsymbol{s}=(\boldsymbol{s}\_1,\dots,\boldsymbol{s}\_k)\in\Delta^{k-1}$ — вектор сложности, порождённый из равномерного распределения на симплексе Δ; $\mathcal{L}_{\mathsf{task}}$ — кросс-энтропия по обучающей выборке; $\gamma^{(i,j)}(\boldsymbol{s})$ — веса операций на ребре (i,j), получаемые через Gumbel-Softmax от логитов $\boldsymbol{u}_{\boldsymbol{a}}^{(i,j)}(\boldsymbol{s})$; γ — вектор состоящий из $\gamma^{(i,j)}$; $\bar{\gamma}=\mathbb{E}[\gamma(t)]$; *Cost* — штраф за использование дорогих операций; κ — гиперпараметр, управляющий важностью штрафа.