Дифференцируемый алгоритм поиска архитектуры модели с контролем её сложности

К. Д. Яковлев, О. С. Гребенькова, О. Ю. Бахтеев
iakovlev.kd@phystech.edu; grebenkova.os@phystech.edu; bakhteev@phystech.edu

В работе исследуется задача построения модели глубокого обучения. Предлагается метод поиска архитектуры модели, позволяющий контролировать её сложность с небольшими вычислительными затратами. Под сложностью модели понимается минимальная длина описания, минимальное количество информации, которое требуется для передачи информации о модели и выборке. В основе метода лежит дифференцируемый алгоритм поиска архитектуры модели (DARTS). Предлагается использовать гиперсеть в качестве функции релаксации. Под гиперсетью понимается модель, генерирующая параметры другой модели. Предложенный метод позволяет контролировать сложность модели в процессе поиска архитектуры. Для оценки качества предлагаемого алгоритма проводятся эксперименты на выборке MNIST.

Ключевые слова: *дифференцируемый алгоритм поиска архитектуры; глубокое обучение; гиперсети; нейронные сети; контроль сложности модели*

1 Введение

В данной работе рассматривается задача поиска архитектуры модели глубокого обучения с контролем её сложности. Под моделью понимается суперпозиция функций, решающая задачу классификации или регрессии [1]. Под поиском архитектуры модели понимается поиск архитектуры ячейки. В качестве базового алгоритма используется дифференцируемый алгоритм поиска архитектуры (DARTS) [2]. Данный метод решает задачу поиска архитектуры модели путем перевода пространства поиска из дискретного в непрерывное представление. В связи с этим появляется возможность использовать градиентные методы оптимизации, позволяющие использовать меньше вычислительных ресурсов, чем методы, работающие на дискретном множестве. Данный алгоритм работает как со сверточными, так и с рекуррентными нейронными сетями.

В работе [3] было указано, что DARTS нестабилен. Одним из источников нестабильности является этап получения фактической дискретной архитектуры из архитектуры непрерывной смеси. На этом этапе наблюдается снижение качества модели. Это связано с тем, что параметры архитектуры модели сходятся в узкий регион, поэтому небольшие возмущения архитектуры ведут к значительному понижению качества на валидационной выборке.

В работе [4] было замечено, что операция softmax обладает существенным недостатком. Некоторые компоненты вектора параметров архитектуры увеличиваются быстрее, чем остальные. В связи с этим предлагается использовать сигмоидную функцию потерь и отказаться от нормировки. Таким образом, все компоненты изменяются с одинаковой скоростью. В данной работе предлагается в качестве функции релаксации использовать гиперсеть [5]. Подход заключается в использовании небольшой сети для генерации параметров архитектуры другой сети. Предлагаются альтернативные подходы к решению задачи поиска архитектуры модели. В работе [6] формулируется задача обучения распределению. Параметры архитектуры подчинены распределению Дирихле, так как они определены на вероятностном симплексе. Таким образом, задача поиска архитектуры сводится к поиску параметров распределения Дирихле.

В работе [7] строится метод поиска нейронной архитектуры с ограниченным ресурсом (RC-DARTS). К базовому алгоритму DARTS добавляются ограничения, такие как число параметров модели. Для решения задачи условной оптимизации вводится алгоритм итерационной проекции, заключающийся в том, что через определенное число итераций градиентного спуска происходит проецирование на множество, задаваемое ограничениями. В данной работе для контроля сложности используется гиперсеть.

Вычислительный эксперимент проводится на выборке MNIST [8].

2 Постановка задачи

2.1 Дифференцируемый алгоритм поиска архитектуры ячейки

Поставим задачу поиска архитектуры ячейки. Ячейка представляет собой N занумерованных узлов, пердставленных в виде ориентированного ациклического графа. Каждому ребру (i, j) поставлено в соответствие отображение (операция) $o^{(i,j)} \in \mathcal{O}$, где $\mathcal{O}$ – семейство отображений(множество операций). Значения в каждом из промежуточных узлов $x^{(j)}$ определяются через значения в узлах с меньшим номером:

$$x^{(j)} = \sum_{i<j} o^{(i,j)}(x^{(i)}) \tag{1}$$

Таким образом, задача поиска архитектуры заключается в выборе операций между узлами ячейки. Для того, чтобы свести задачу дискретной оптимизации к задаче непрерывной оптимизации, введем смешанную операцию для каждого ребра (i, j):

$$\hat{o}^{(i,j)}(x) = \sum_{o\in\mathcal{O}} \frac{\exp(\alpha_o^{(i,j)})}{\sum_{o'\in\mathcal{O}} \exp(\alpha_{o'}^{(i,j)})} o(x), \tag{2}$$

где $\alpha_o^{(i,j)}$ обозначает соответствующий вес операции o на ребре (i, j). Таким образом, каждому ребру (i, j) ставится в соответствие вектор $\alpha^{(i,j)}$ размерности $|\mathcal{O}|$. Пусть $\alpha = [\alpha^{(i,j)}]$. Сформулируем двухуровневую задачу оптимизации:

$$\begin{gathered} \min_{\alpha} \mathcal{L}_{\text{val}}(\mathbf{w}^*(\alpha), \alpha), \\ \text{s.t.} \quad \mathbf{w}^* = \arg\min_{\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha) \end{gathered} \tag{3}$$

Здесь $\mathcal{L}_{\text{val}}$ и $\mathcal{L}_{\text{train}}$ функции потерь модели на валидации и на обучении соответственно.

2.2 Линейная гиперсеть

Пусть Λ – множество параметров, контролирующие сложность модели. Гиперсеть – это следующее отображение:

$$\mathbf{G} : \Lambda \times \mathbb{U} \to \mathbb{A}, \tag{4}$$

где $\mathbb{A}$ – пространство параметров архитектуры $\alpha_o^{(i,j)}$, а $\mathbb{U}$ – множество параметров гиперсети. В данной работе для получения весов операций используется линейная гиперсеть:

$$\mathbf{G}_{\text{linear}}(\lambda) = \lambda\mathbf{b}_1 + \mathbf{b}_2, \tag{5}$$

где $\mathbf{b}_1, \mathbf{b}_2$ настраиваются согласно оптимизационной задаче 3.

3 Описание алгоритма

В качестве базового алгоритма используется алгоритм, описанный в работе [2]. Идея состоит в приближении истинного градиента:

$$\nabla_\alpha \mathcal{L}_{\text{val}}(\mathbf{w}^*(\alpha), \alpha) \approx \nabla_\alpha \mathcal{L}_{\text{val}}(\mathbf{w} - \xi \nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha), \alpha)$$

Пусть $\mathbf{w}'(\alpha) = \mathbf{w} - \xi \nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha)$. Тогда получаем:

$$\nabla_\alpha \mathcal{L}_{\text{val}}(\mathbf{w}'(\alpha), \alpha) = \nabla_\alpha \mathcal{L}_{\text{val}}(\mathbf{w}, \alpha)\big|_{\mathbf{w}=\mathbf{w}'(\alpha)} - \xi \nabla^2_{\alpha,\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha) \nabla_{\mathbf{w}'} \mathcal{L}_{\text{val}}(\mathbf{w}'(\alpha), \alpha) \tag{6}$$

Обозначим $\mathbf{w}^\pm = \mathbf{w} \pm \varepsilon \nabla_{\mathbf{w}'} \mathcal{L}_{\text{val}}(\mathbf{w}'(\alpha), \alpha)$, где $\varepsilon = 0.01 / \|\nabla_{\mathbf{w}'} \mathcal{L}_{\text{val}}(\mathbf{w}'(\alpha), \alpha)\|_2$. Тогда

$$\nabla^2_{\alpha,\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha) \nabla_{\mathbf{w}'} \mathcal{L}_{\text{val}}(\mathbf{w}'(\alpha), \alpha) \approx \frac{\nabla_\alpha \mathcal{L}_{\text{train}}(\mathbf{w}^+, \alpha) - \nabla_\alpha \mathcal{L}_{\text{train}}(\mathbf{w}^-, \alpha)}{2\varepsilon} \tag{7}$$

Следовательно, градиент 6 вычисляется за $O(\dim \mathbf{w} + \dim \alpha)$. Заметим, что без оценки 7 приходилось бы осуществлять $O(\dim \mathbf{w} \dim \alpha)$ операций. Приведем псевдокод алгоритма:

Алгоритм 1 DARTS – Differentiable Architecture Search

1: Для каждого узла создадим смешанную операцию $\hat{o}^{(i,j)}$, параметризованную $\alpha^{(i,j)}$
2: **пока** алгоритм не сошелся
3: обновить α, сделав градиентный шаг вдоль $\nabla_\alpha \mathcal{L}_{\text{val}}\big(\mathbf{w} - \xi \nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha), \alpha\big)$
4: обновить веса $\mathbf{w}$, сделав градиентный шаг вдоль $\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w}, \alpha)$
5: получить окончательную архитектуру из полученного α

Дискретная архитектура получается следующим образом:

$$o^{(i,j)} = \arg\max_{o \in \mathcal{O}} \alpha_o^{(i,j)}$$

В предлагаемом алгоритме смешанная операция определяется как:

$$\hat{o}^{(i,j)}(x) = \sum_{o \in \mathcal{O}} \alpha_o^{(i,j)} o(x)$$

Вектор параметров архитектуры модели определяется линейной гиперсетью:

$$\alpha = \lambda \mathbf{b}_1 + \mathbf{b}_2$$

Одним из свойств предлагаемого решения является то, что изменение сложности модели происходит заменой параметра λ гиперсети без дополнительного обучения.

4 Вычислительный эксперимент

4.1 Базовый эксперимент

Целью базового эксперимента является получение зависимости качества работы алгоритма DARTS с регуляризатором от количества эпох при разных значениях параметра регуляризации.

Предлагается использовать алгоритм DARTS с регуляризатором $\lambda \sum_{i=1}^{\dim \alpha} \mathbf{v}_i |\alpha_i|$ в качестве второго слагаемого в функции потерь, где $\mathbf{v}$ – вектор весов опреаций. Чем сложнее

операция, тем больше соответствующий вес. Вычислительный эксперимент проводится на выборке MNIST [8], которая представляет собой набор рукописных цифр.

Эксперимент запускался 5 раз при значениях $\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. На рисунке 1 представлен график зависимости точности (precision) модели от числа эпох.

График показывает, что при большом значении параметра регуляризации качество сильно падает. Это связано с тем, что в этом случае не выгодно брать операции с большим весом. Таким образом, модель становится переупрощенной.

Также из графика видно, что при $\lambda \in \{10^{-4}, 10^{-3}\}$ качество модели значительно возрастает, что связано с тем, что в архитектуре появляются операции, имеющие большой вес.

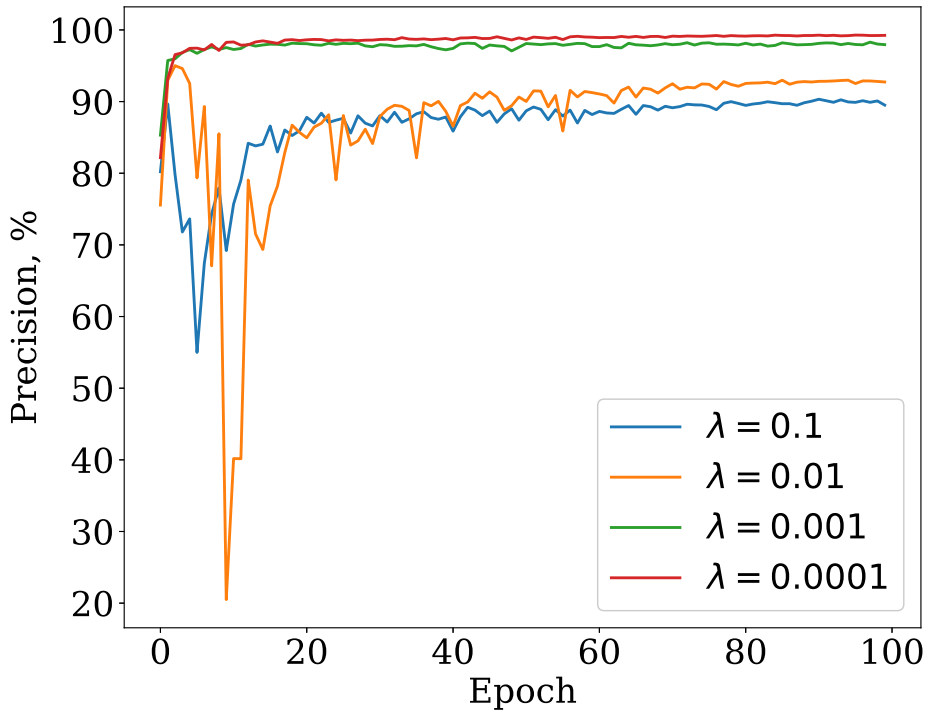

Рис. 1 Зависимость точности модели от числа прошедших эпох для разынх параметров регуляризации

4.2 Основной эксперимент

Целью основного эксперимента является получение зависимости качества работы предложенного метода от параметра гиперсети $\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. Эксперимент проводится на выборке MNIST [8]. Рассматривался поиск архитектуры сверночной нейронной сети с одним слоем и четырьмя узлами. Контроль сложности производился линейной гиперсетью 5. Обучение проводилось на протяжении 100 эпох. В процессе обучения параметр λ выбирался равномерно из отрезка $[10^{-5}, 10^{-1}]$. Как видно из графика 2, для $\lambda = 0.1$ качество модели заметно хуже, чем для других значениях λ. Это связано с тем, что штраф за сложность модели становится большим, поэтому происходит переупрощение модели. Также для каждой эпохи и для каждого $\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}\}$ качество модели практически

не меняется. Это означает, что качество модели заметно не ухудшается при уменьшении сложности.

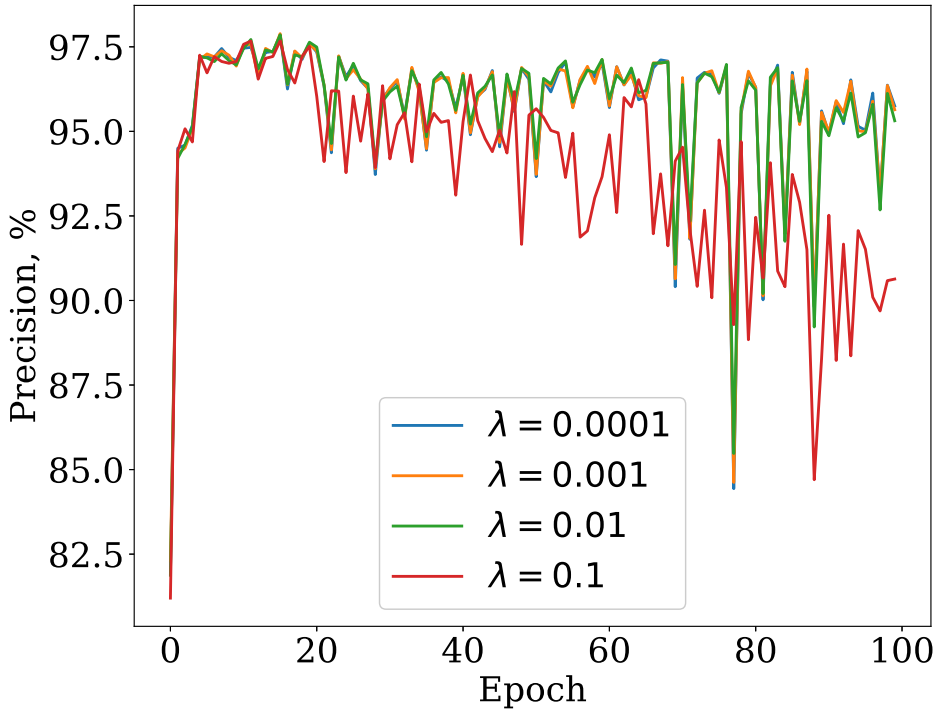

Рис. 2 Зависимость качества модели от числа прошедших эпох для разных параметров λ гиперсети.

4.3 Анализ ошибки

Модель	Precision, %			
	эпоха 30	эпоха 50	эпоха 70	эпоха 100
Hypernet, $\lambda = 0.1$	96.3500	95.4800	94.1200	90.6333
Hypernet, $\lambda = 0.01$	95.8900	96.7167	91.0633	95.3133
Hypernet, $\lambda = 0.001$	95.9133	96.6200	90.6400	95.6400
Hypernet, $\lambda = 0.0001$	95.9733	96.5367	90.4067	95.7533
DARTS, $\lambda = 0.1$	86.6000	87.3967	89.3433	89.5067
DARTS, $\lambda = 0.01$	84.1333	90.6333	91.9100	92.7333
DARTS, $\lambda = 0.001$	97.6533	97.5800	98.0833	97.9467
DARTS, $\lambda = 0.0001$	**98.5800**	**98.8867**	**98.9467**	**99.2400**

Таблица 1 Результаты базового и основного экспериментов. Приведены значения качества моделей на валидации.

Из тыблицы 1 видно, что для модели Hypernet свойственно уменьшение качества на валидации с ростом номера эпохи. Это связано с тем, что происходит выбор более про-

стой архитектуры, а именно, становится больше пропусков соединений. Данное свойство DARTS исследовалось в работе [4].

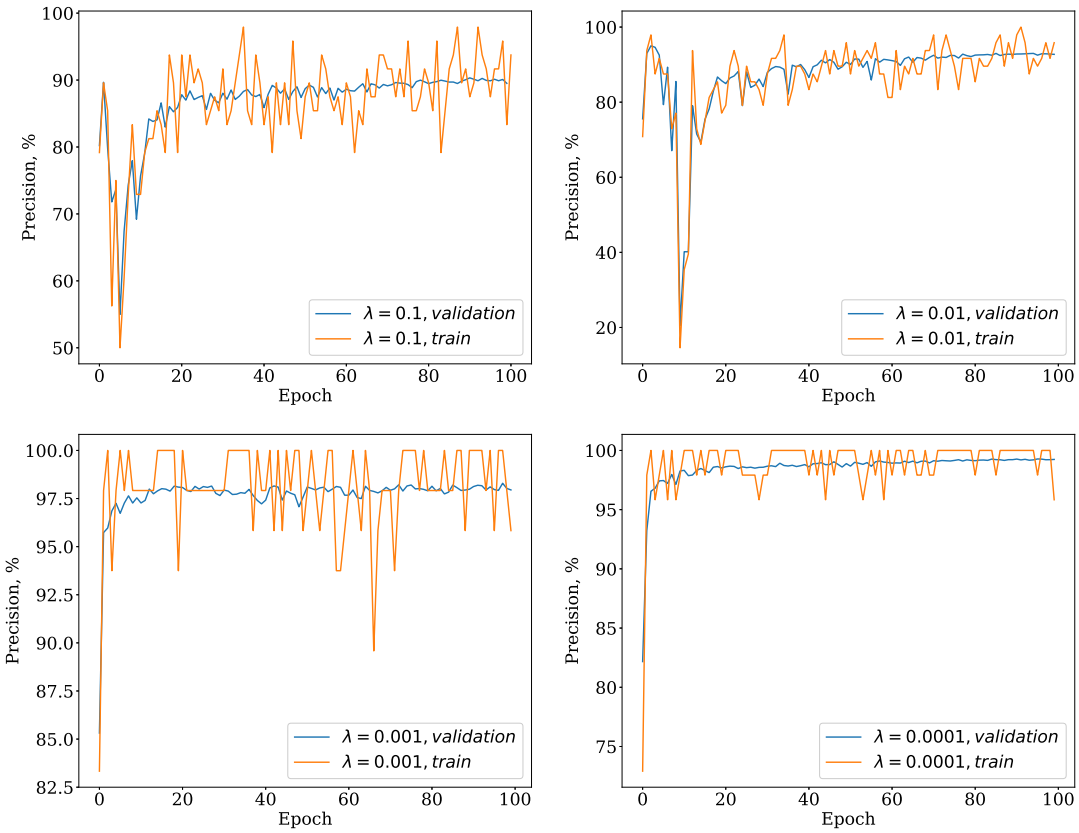

Рис. 3 Зависимость качества на обучающей и тестовой выборках от числа прошедших эпох для разных значений параметров λ регуляризатора в базовом эксперименте.

Из графика 3 видно, что уже после 20 эпохикачество на обучающей и на тестовой выборках пререстаёт сильно меняться. Кроме того, данный график показывает отсутствие переобучения для всех λ.

5 Заключение

В данной работе рассматривалась задача поиска архитектуры модели с контролем сложности. Предложен метод, позволяющий контролировать сложность модели в процессе поиска архитектуры. Метод обладает тем свойством, что изменение сложности итоговой модели происходит заменой параметра λ гиперсети без дополнительного обучения. Также результаты показывают, что данный метод сопоставим по качеству на валидационной выборке с DARTS.

Литература

[1] *Bakhteev Oleg Yu., Strijov Vadim V.* Deep learning model selection of suboptimal complexity // Autom. Remote. Control, 2018. Vol. 79. No. 8. P. 1474–1488.

[2] *Liu Hanxiao, Simonyan Karen, Yang Yiming.* Darts: Differentiable architecture search // CoRR, 2018. Vol. abs/1806.09055. URL: http://arxiv.org/abs/1806.09055.

[3] *Chen Xiangning, Hsieh Cho-Jui.* Stabilizing differentiable architecture search via perturbation-based regularization // CoRR, 2020. Vol. abs/2002.05283.

[4] *Chu Xiangxiang, Zhou Tianbao, 0046 Bo Zhang, Li Jixiang.* Fair darts: Eliminating unfair advantages in differentiable architecture search // CoRR, 2019. Vol. abs/1911.12126. URL: http://arxiv.org/abs/1911.12126.

[5] *Ha David, Dai Andrew M., Le Quoc V.* Hypernetworks // CoRR, 2016. Vol. abs/1609.09106. URL: http://arxiv.org/abs/1609.09106.

[6] *Chen Xiangning, Wang Ruochen, Cheng Minhao, Tang Xiaocheng, Hsieh Cho-Jui.* Drnas: Dirichlet neural architecture search // CoRR, 2020. Vol. abs/2006.10355.

[7] *Jin Xiaojie, Wang Jiang, Slocum Joshua, 0001 Ming-Hsuan Yang, Dai Shengyang et al.* Rc-darts: Resource constrained differentiable architecture search // CoRR, 2019. Vol. abs/1912.12814. URL: http://arxiv.org/abs/1912.12814.

[8] *LeCun Yann, Cortes Corinna.* MNIST handwritten digit database, 2010. URL: http://yann.lecun.com/exdb/mnist/.

Received February 25, 2021