1 Введение

Проблему выбора структуры модели глубокого обучения можно сформулировать следующим образом: решается задача классификации или регрессии на заданной выборке. Требуется выбрать структуру нейронной сети, доставляющей минимум ошибки на этой функции и максимум качества на некотором внешнем критерии. Под моделью глубокого обучения понимается суперпозиция дифференцируемых нелинейный функций. Под структурой модели понимается значения структурных параметров модели, т.е. параметров модели, характеризующий вид итоговой суперпозиции.

Смежной задачей к задаче выбора структуры модели является задача корректного представления структуры сети или параметризация сети глубокого обучения. Одним из возможных представлений структуры модели является графовое представление, в котором в качестве ребер графа выступают нелинейные функции, а в качестве вершин графа — представление выборки под действием соответствующих нелинейных функций. Данный подход к описанию модели является достаточно общим и коррелирует с походом, описанным в [?], а также в библиотеках типа TensorFlow, Caffe, Teano, Torch, в которых модель рассматривается как граф, ребрами которого выступают математические операции, а вершинами — результат их действия на выборку. В то же время, существуют и другие способы представления модели. В то же время, в ряде работ, посвященных байесовской оптимизации [1, 2, 3], модель рассматривается как "черный ящик", имеющий ограниченный набор операций типа "произвести оптимизацию параметров" и "предсказать значение зависимой переменной по независимой переменной". Подход, описанный в данных работах, также коррелирует с библиотеками машинного обучения, такими как Weka, RapidMiner или sklearn, в которых модель машинного обучения рассматривается как "черный ящик".

Заметим, что частным случаем выбора структуры глубокой сети является выбор обобщенно-линейных моделей, т.к. отдельные слои нейросети можно рассматривать как обобщенно-линейные модели. Задачу выбора обобщенно-линейной модели можно рассматривать как задачу выбора признаков, методы решения которой делятся на три группы [4]:

1. Фильтрационные методы. Основной особенностью данных методов является то, что такие методы не используют какой-либо информации о модели, а отсекают признаки только на основе статистических показателей.

2. Оберточные методы — методы, анализирующие подмножества признаков. Такие методы выбирают не признаки, а подмножества признаков, что позволяет учесть корреляция признаков.

3. Методы погружения проводят оптимизацию моделей и выбор признаков в единой процедуре, являясь комбинацией предыдущих типов отбора признаков.

В данном обзоре методы порождения и выбора обобщенно-линейных моделей не рассматривается в силу общности рассматриваемой задачи.

2 Постановка задачи

Задана выборка

$$\mathfrak{D} = \{(\mathbf{x}_i, y_i)\}, i = 1, \ldots, m, \tag{1}$$

состоящая из множества пар «объект-метка»,

$$\mathbf{x}_i \in \mathbf{X} \subset \mathbb{R}^n, \quad y_i \in \mathbf{y} \subset \mathbb{Y}.$$

Метка y объекта $\mathbf{x}$ принадлежит либо множеству: $y \in \mathbb{Y} = \{1, \ldots, Z\}$ в случае задачи классификации, где Z — число классов, либо некоторому подмножеству вещественных чисел $y \in \mathbb{Y} \subseteq \mathbb{R}$ в случае задачи регрессии.

Моделью глубокого обучения $\mathbf{f}$ назовем суперпозицию функций

$$\mathbf{f}(\mathbf{w}, \mathbf{X}) = \mathbf{f}_1(\mathbf{f}_2(\ldots \mathbf{f}_K(\mathbf{X}))) : \mathbb{R}^{m \times n} \to \mathbb{Y}^m, \tag{2}$$

где $\mathbf{f}_k$ — подмодели, параметрическое семейство дважды дифференцируемых по параметрам вектор-функций, $k \in \{1, \ldots, K\}$; $\mathbf{w} \in \mathbb{R}^u$ — вектор параметров моделей.

Для каждой модели определена функция правдоподобия $p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \mathbf{f})$, где $\mathbf{x}$ — строка матрицы $\mathbf{X}$, $\mathbf{y}$ — вектор меток зависимой переменной y.

Будем полагать, что множество рассматриваемых моделей задается некоторой функцией $\mathfrak{F}(\beta)$. Для каждой модели $\mathbf{f}$ из конечного множества моделей $\mathfrak{F}(\beta)$ задано априорное непрерывное распределение параметров $p(\mathbf{w}|\alpha)$.

Пусть задана дифференцируемая функция Q, определяющая качество модели.

Определение Модель классификации $\hat{\mathbf{f}}$ назовем оптимальной среди моделей $\mathfrak{F}$, если достигается максимум качества Q:

$$\hat{\mathbf{f}} = \arg\max_{\mathbf{f} \in \mathfrak{F}} Q(\mathbf{f}).$$

Определение Назовем оператором оптимизации алгоритм T выбора вектора параметров $\mathbf{w}'$ по предыдущему значению параметров модели $\mathbf{w}$:

$$\mathbf{w}' = T(\mathbf{w}, \gamma),$$

где γ — вектор параметров оптимизации.

Требуется найти оптимальную модель $\mathbf{f}$ среди заданного множества моделей $\mathfrak{F}$, а также значения ее параметров $\mathbf{w}$, доставляющие максимум функции Q:

$$\mathbf{f} = \arg\min_{\mathbf{f}' \in \mathfrak{F}} \min_{\gamma, \alpha} Q, \tag{3}$$

$$\mathbf{w} = T(\mathbf{w}, \gamma).$$

В дальнейшем будем использовать следующие наименования для групп параметров, участвующих в задаче оптимизации (3):

1. $\mathbf{w}$ — множество параметров модели.
2. α — множество гиперпараметров модели.
3. β — множество структурных параметров.
4. γ — метапараметры.

3 Метаоптимизация

Задача выбора структуры модели тесно связана с раздел машинного обучения под названием *метаобучение*. Под метаобучением понимаются алгоритмы машинного оубчения [5], которые:

1. могут оценивать и сравнивать методы оптимизации моделей
2. оценивать возможные декомпозиции процесса оптимизации моделей
3. на основе полученных оценок предлагать оптимальные стратегии оптимизации моделей и отвергать неоптимальные стратегии.

3.1 Теоретические основания метаобучения

В работе [6] рассматривается задача построения порождающих моделей, предлагается критерий для послойного обучения порождающих моделей:

$$\mathcal{U}_{\mathcal{D}}(\theta_I) := \max_{Q} \mathbb{E}_{\mathbf{x}\sim P_{\mathcal{D}}} \left[\log \sum_{\mathbf{h}} P_{\theta_I}(\mathbf{x}|\mathbf{h}) Q(\mathbf{h}) \right]$$

В работе [7] рассматриваются подходы к сэмплированию моделей глубокого обучения. Под *сэмплированием* понимается порождение нескольких экземпляров модели из заданного распределения для дальнейшего выбора наилучшей модели. Предлагается формализация пространства поиска и формальное описание элементов пространства моделей:

```
(Concat
    (Conv2D [32, 64] [3, 5] [1])
    (MaybeSwap BatchNormalization ReLU)
    (Optional (Dropout [0.5, 0.9]))
    (Affine [10]))
```

Figure 1. A simple search space with 24 different models. See Figure 2 for a path through the search space.

3.2 Метаоптимизация: learning to learn

В работе [8] предлагается подход к адаптивному изменению структуры сети, основанный на обучении с подкреплением. Предлагается параметризация модели нейросети, включающая в себя модифицирующие и анализирующие выходы, позволяющие модифицировать параметры модели:

$$net_{y_k}(1) = 0, \quad \forall t \geq 1: \quad x_k(t) \leftarrow environment,$$

$$y_k(t) = f_{y_k}(net_{y_k}(t)),$$

$$\forall t > 1: \quad net_{y_k}(t) = \sum_l w_{y_k l}(t-1) l(t-1), \qquad (7)$$

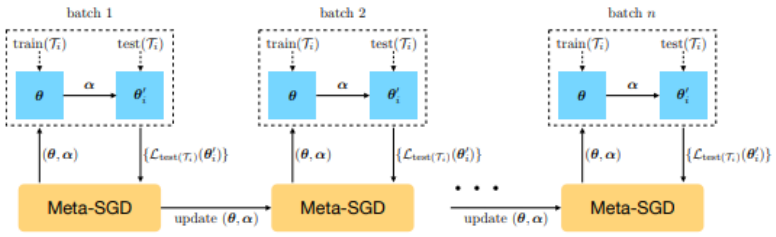
$$\forall t \geq 1: \quad w_{ij}(t+1) = w_{ij}(t) + \triangle(t)\, g[\, \|adr(w_{ij}) - mod(t)\|^2\,] \qquad (8)$$

$$val(1) = 0, \quad \forall t \geq 1: \quad val(t+1) =$$

$$= \sum_{i,j} g[\|ana(t) - adr(w_{ij})\|^2] w_{ij}(t). \qquad (9)$$

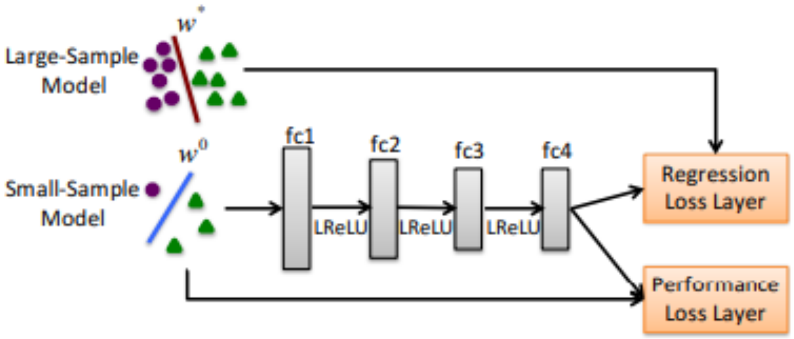
Предлагается продолжение подхода, позволяющая рекуррентно продолжать анализ модели и порождать мета-мета-...-анализ.

В работе [9] рассматривается оптимизация метапараметров (шага градиентного спуска и начального распределения параметров) с использованием обучения с подкреплением. На каждой итерации сэмплируется подвыборка, по которой проводится оптимизация данных метапараметров:

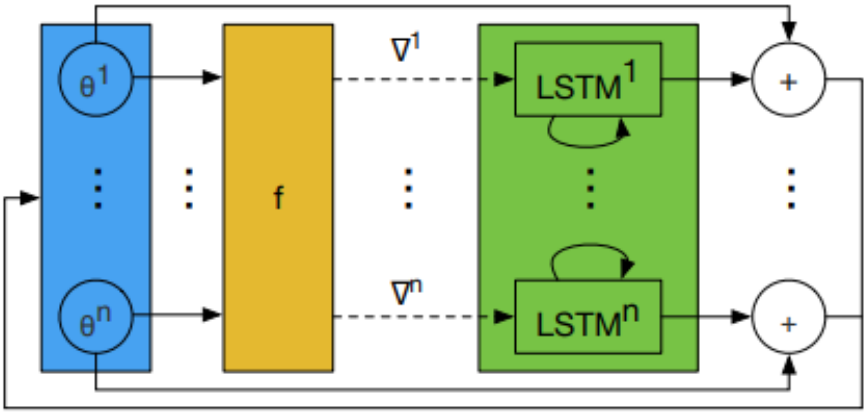


В работе [10] рассматривается задача восстановления параметров модели по параметрам слабо обученной модели:

$$L(\Theta) = \sum_{j=1}^{J} \left\{ \frac{1}{2} \left\| \mathbf{w}_j^* - T\left(\mathbf{w}_j^0, \Theta\right) \right\|_2^2 + \lambda \sum_{i=1}^{M+N} \left[1 - y_i^j \left(T\left(\mathbf{w}_j^0, \Theta\right)^T \mathbf{x}_i^j \right) \right]_+ \right\}. \quad (1)$$



В работе [11] рассматривается оптимизация метапараметров оптимизации с помощью LSTM, которая выступает альтернативе аналитических алгоритмов, таких как Adam или AdaGrad. LSTM имеет (сравнительно) небольшое количество параметров, т.к. для каждого метапараметра используется своя копия модели LSTM с одинаковыми параметрами для каждой копии:



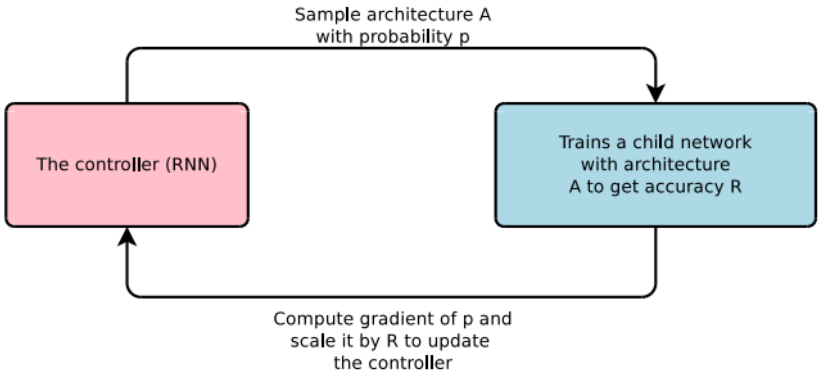
3.3 Перебор структур

В работе [12] рассматривается задача порождения сверточных нейронных сетей. Предлагается проводить поиск оптимальной структуры сети по восходящему по сложности порядку: начиная от сетей с одни блоком и наращивая блоки. В силу высокой вычислительной сложности данного подхода, вместо построения модели,предлагается обучить рекуррентную нейросеть,которая предсказывает качество модели по заданным блокам.

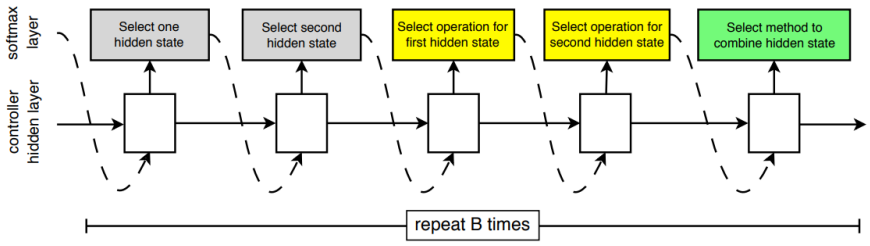
В работе [13] рассматривается задача выбора архитектуры с помощью большого количества параллельных запусков обучения моделей, предлагаются критерии ранней остановки оптимизации обучения моделей.

3.4 Обучение с подкреплением

В работе [14] представлена схема выбора архитектуры сверточной нейросети с использованием обучения с подкреплением. В качестве актора (контроллера) выступает рекуррентная нейронная сеть.

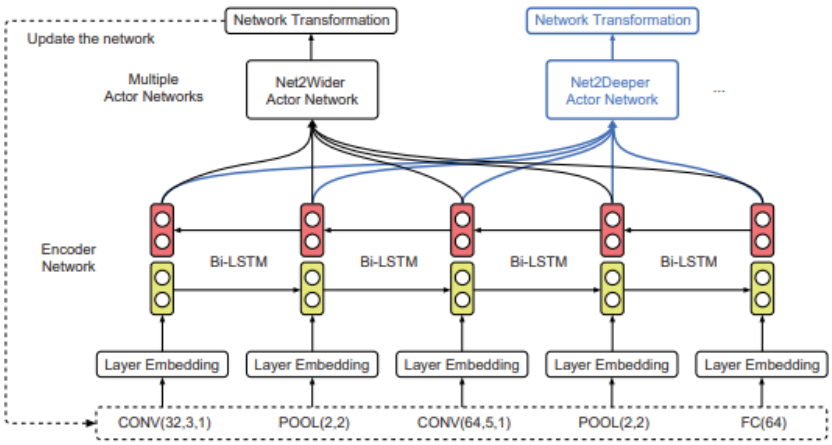


В работе [15] предлагается построение регрессионной модели для оценки финального качества модели и ранней остановки оптимизации моделей. Данный подход позволил существенно ускорить поиск моделей, представленный в работе [14]. В работе [16] рассматривается задача переноса архитектуры нейросети, обученной на более простой выборки, на более сложную. Также предлагается параметризация пространства поиска, более делатьное, чем в [14]:



В отличие от предыдущих работ, в работе [17] предлагается подход к инкрементальному обучению нейросети, основанном на модификации модели, полученной на предыдущем шаге. Рассматривается две операции над нейросетью:

- Расширение сети
- Углубление сети



4 Адаптивное изменение структуры

В данном разделе собраны методы изменения структуры существующей модели.

Алгоритмы наращивания и прореживания параметров модели В работе [18] предлагается удалять неинформативные параметры модели, где в качестве показателя информативности выступает следующий фнуционал:

$$\delta E = \sum_i g_i \delta u_i + \frac{1}{2} \sum_i h_{ii} \delta u_i^2 + \frac{1}{2} \sum_{i \neq j} h_{ij} \delta u_i \delta u_j + O(||\vec{R}^-||^3) \qquad (1)$$

В работе [19] было предложено развитие данного метода. В данной работе, в отличие от [18] не вводится предположений о дигональности Гессиана функции ошибок, поэтому удаление неинформативных параметров модели производится точнее.

В работе [20] был предложен метод, основанный на получении вариационной нижней оценки правдоподобия модели. В качестве критерия информативности параметра выступало отношение вероятности нахождения параметра в пределах априорного распределения к вероятности равенста параметра нулю:

$$\exp\left(-\frac{\mu_i^2}{2\sigma_i^2}\right) > \gamma \implies \left|\frac{\mu_i}{\sigma_i}\right| < \lambda$$

Идея данного метода была развита в [21], где также используются вариационные методы. В отличие от предыдущей работы, в данной работе рассматривается ряд априорных распределений параметров, позволяющих прореживать модели более эффективно:

- Нормальное распределение с лог-равномерным распределением дисперсии, независимой для каждого нейрона:

$$p(\mathbf{W}, \mathbf{z}) \propto \prod_i^A \frac{1}{|z_i|} \prod_{ij}^{A,B} \mathcal{N}(w_{ij}|0, z_i^2),$$

- Произвденеие двух половинных распределений Коши: одно ответственно за отдельный параметр, другое — за общее распределение параметров:

$$s \sim \mathcal{C}^+(0, \tau_0); \qquad \tilde{z}_i \sim \mathcal{C}^+(0, 1); \qquad \tilde{w}_{ij} \sim \mathcal{N}(0, 1); \qquad w_{ij} = \tilde{w}_{ij}\tilde{z}_i s,$$

Смежной темой к прореижванию моделей выступает компрессия нейросетей. Основным отличием задачи прореживания и компрессии выступает эксплуатационное требование: если прореживание используется для получения оптимальной и наиболее устойчивой модели, то компрессия часто производится для сохранения памяти и основных эксплуатационных характеристик исходной модели (?). В работе [22] предлагается итеративное использование регуляризации типа DropOut [23] для прореживания модели. В работах [24, 25] используются методы снижения вычислительной точности представления парамеров модели на основе кластеризации весов. В работе [25] предлагается метод компрессии, основанный на кластеризации значений параметров модели и представлении их в сжатом виде на основе кодов Хаффмана.

В работах [26, ?] предлагается наращивание моделей, основанное на бустинге. В работе рассматривается задача построения нейросетевых моделей

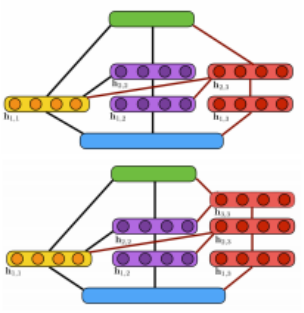
специального типа:

$$f_{t+1} = \sigma(f_t) + f_t,$$

приводится параметризация модели, позволяющая рассматривать декомпозировать модель на слабые классификаторы. В работе [?] на каждом шаге построения выбирается одно из двух расширений модели, каждое из которых рассматрвается как слабый классификатор: 1. Сделать модель шире 2. Сделать модель глубже

(a)

(b)



Построение модели заканчивается при условии снижении радемахереовской сложности:

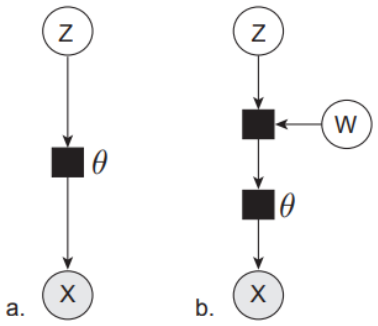
$$\widehat{\mathfrak{R}}_S(\mathcal{G}) = \frac{1}{m}\,\mathbb{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in\mathcal{G}}\sum_{i=1}^{m}\sigma_i h(x_i)\right],$$

5 Байесовские методы порождения и выбора моделей

5.1 Автоматическое определение релевантности параметров

В работе [27] рассматривается задача оптимизации гиперпараметров. Авторы предлагают оптимизировать константы l_2-регуляризации отдельно для каждого параметра модели, проводится параллель с методами автоматического определения релевантности параметров (ARD) [28].

В работе [29] рассматривается метод ARD для снижения размерности скрытого пространства вариационных порождающих моделей: скрытая переменная параметризуется как произведение некоторой случайной величины $\mathbf{z}$ на вектор, отвечающий за релевантность каждой компоненты скрытой переменной:



5.2 Суррогаты

В работе [3] предлагается моделировать качество модели гауссовым процессом, параметрами которого выступают гиперпараметры исходной модели. Модель, аппроксимирующая качество исходной модели, называется суррогатом.

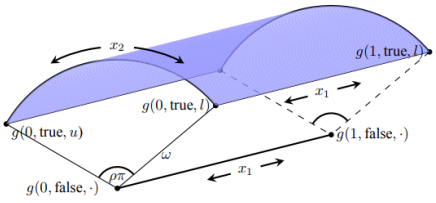
Одна из основных проблем использования гауссового процесса как суррогатной модели — кубическая сложность оптимизации. В работе [30] предлагается использовать случайные подпространства гиперпараметров для ускоренной оптимизации. В работе [31] предлагается кобминация из множества гауссовых моделей и линейной модели, позволяющая модели нелинейные зависимости гиперпараметров, а также существенно сократить сложность оптимизации.

В работе [2] предлагается рассматривать RBF-модель для аппроксимации качества исходной модели, что позволяет ускорить процесс оптимизации суррогатной модели. В [1] рассматривается глубокая нейронная сеть в качестве суррогатной функции. Вместо интеграла правдоподобия, который оценивается в случае использования гауссового процесса в качестве суррогата, используется максимум апостериорной веротяности.

Важным параметром гауссовых процессов является функиия ядра гауссового процесса, полностью определяющая процесс в случае нулевого среднего. В работе [32] предлагается функция ядра, определенная на графах:

$$k(x, y) = r(d(x, y)),$$

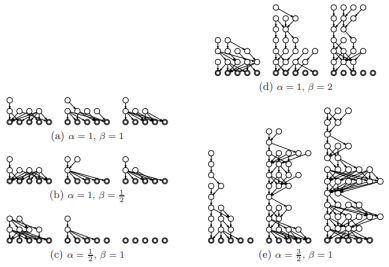
где d — геодезическое расстояние между вершинами графа, r — некоторая вещественная функция (наверно положительно определенная, но это не указано явно в статье). В работе [33] рассматривается задач выбора структуры нейросети, предлагается ядро специального вида, позволюящее учитывать только те гиперпапараметры, которые есть в обеих сравниваемых моделях: к примеру, для двуслойной и трехслойной нейросети будут уччитываться гиперпараметры, отвечающие только за первые два слоя.



5.3 Адаптивное изменение структуры

В работе [34] рассматривается порождение unsupervised-моделей с использованием расширения процесса Индийского Буфета:

$$p(K^{(m+1)} = k \mid K^{(m)}, \alpha, \beta) = \frac{1}{k!} \exp\left\{-\lambda(K^{(m)}; \alpha, \beta)\right\} \lambda(K^{(m)}; \alpha, \beta)^k,$$



В работе [35] предлагается упрощенная модель Индийского Буфета:

$$-\log p(x, W, z) \sim \sum_{i-1}^{N} \|x_i - W z_i\|_2^2 + \lambda^2 K$$

В работе [36] предлагается параметризация структуры модели с использованием Бернуллиевских величин: каждая величина отвечает за включение или выключение слоя сети.

5.4 Порождающие модели

В работе [37] было предложено обобщение вариационного автокодировщика на случай частичного обучения: итоговая модель вариационного автокодировщика является порождающей моделью, учитывающий метки объектов.

В работе [38] рассматривается обобщение вариационного автокодировщика на случай более общих графических моделей. Рассматривается проводить оптимизацию сложных графических моделей в единой процедуре.

Для вывода предлагается использовать нейронные сети. Другая модификация вариационного автокодировщика представлена в работе [39], авторы рассматривают использование процесса сломанной трости в вариационном автокодировщике, тем самым получая модель со стохастической размерностью скрытой переменной. В работе [40] рассматривается смесь автокодировщиков, где смесь моделируется процессом Дирихле.

В работе [41] предлагается подход к оптимизации неизвестного распределения с помощью вариационного вывода. Авторы предлагают решать задачу оптимизации итеративно, добавляя в модель новые компоненты вариационного распределения, проводится аналогия с бустингом.

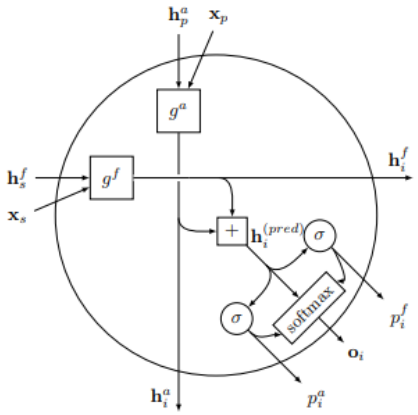
5.5 Состязательные модели

6 Способы прогнозирования графовых структур

В разделе собраны ключевые работы по порождению графовых моделей.

В работе [42] предлагается метод прогнозирования графовой структуры на основе линейного программирования. Предлагается свести проблему поиска графовой структуры к комбинаторной проблеме.

В работе [43] предлагается метод прогнозирования структур деревьев, основанный на дважды-рекуррентных нейросетях (doubly-reccurent), т.е. на сетях, отдельно предсказывающих глубину и ширину уровней деревьев.



7 Эвристические и прикладные методы

7.1 Эвристические методы

В работе [44] предлагается метод анализа структуры сети на основе линейных классификаторов, построенных на промежуточных слоях нейросети. Схожий метод был предложен в [45], где классификаторы на промежуточных уровнях используются для уменьшения вычислений при выполнении вывода и предсказаний. Промежуточные классификаторы.работают как решающий список http://www.eecs.harvard.edu/ htk/publication/2016-icpr-teerapittayanon-mcdanel-kung.pdf

В работе [?] предлагается инкрементальный метод построения нейросети: на каждом этапе построения в модель добавляются новые слои. Для улучшения качества модели, слои добавляются в начало модели, и затем проходят оптимизацию.

7.2 Структуры сетей специального вида

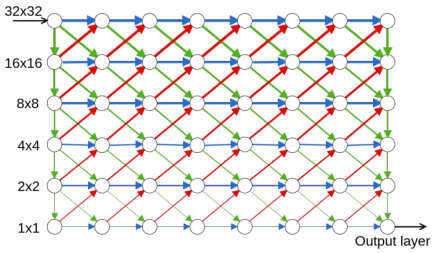
В данном разделе представлены работы по поиску оптимальной структуры сети, описывающие частные случаи поиска оптимальных моделей со структурами специального вида.

В работе [46] рассматривается оптимизация моделей нейросетей с бинарной функцией активацией. Задача оптимизации сводится к задаче mixed integer программирования, которая решается методами выпуклого анализа.

SKIP-сети, нужно ли писать? ResNet?

В работе [47] предлагается метод построения сети глубокого обучения, структура которой выбирается с использованием обучения без учителя. Критерий оптимальности модели использует оценки энергитических функций и ограниченной машины Больцмана.

В работах [48, 49] рассматривается выбор архитектуры сети с использованием *суперсетей*: больших связанных между собой сетей, образующих граф, пути в котором определяют итоговую архитектуру нейросети. В работе [49] рассматриваются стохастические суперсет, позволяющие выбрать структуру нейросети за органиченное время оптимизации. Схожий подход был предложен в работе [48], где предлагается использовать эволюционные алгоритмы для запоминания оптимальных подмоделей и переноса этих моделей в другие задачи.



В работах [50, 51, 52] рассматриваются методы деформации нейросетей. В работе [52] предлагается метод оптимального разделения нейросети на несколько независимых сетей для уменьшения количества связей и, как следствие, уменьшения сложности оптимизации модели. В работе [50] предлагается метод сохранения результатов оптимизации нейросети при построении новой более глубокой или широкой нейросети. В работе [51] рассматривается задача расширения сверточной нейросети, нейросеть рассматривается как граф.

Список литературы

[1] Scalable Bayesian Optimization Using Deep Neural Networks / Jasper Snoek, Oren Rippel, Kevin Swersky et al. // Proceedings of the 32nd International Conference on Machine Learning / Ed. by Francis Bach, David Blei. — Vol. 37 of *Proceedings of Machine Learning Research*. — Lille, France: PMLR, 2015. — 07–09 Jul. — Pp. 2171–2180. http://proceedings.mlr.press/v37/snoek15.html.

[2] Hyperparameter optimization of deep neural networks using non-probabilistic RBF surrogate model / Ilija Ilievski, Taimoor Akhtar, Jiashi Feng, Christine Annette Shoemaker // *arXiv preprint arXiv:1607.08316*. — 2016.

[3] *Snoek Jasper, Larochelle Hugo, Adams Ryan P.* Practical bayesian optimization of machine learning algorithms // Advances in neural information processing systems. — 2012. — Pp. 2951–2959.

[4] *Li Jundong, Liu Huan.* Challenges of feature selection for big data analytics // *IEEE Intelligent Systems*. — 2017. — Vol. 32, no. 2. — Pp. 9–15.

[5] *Schmidhuber Juergen, Zhao Jieyu, Wiering MA.* Simple principles of metalearning // *Technical report IDSIA*. — 1996. — Vol. 69. — Pp. 1–23.

[6] *Arnold Ludovic, Ollivier Yann.* Layer-wise learning of deep generative models // *arXiv preprint arXiv:1212.1524*. — 2012.

[7] *Negrinho Renato, Gordon Geoff.* Deeparchitect: Automatically designing and training deep architectures // *arXiv preprint arXiv:1704.08792.* — 2017.

[8] *Schmidhuber Jürgen.* A neural network that embeds its own meta-levels // Neural Networks, 1993., IEEE International Conference on / IEEE. — 1993. — Pp. 407–412.

[9] Meta-SGD: Learning to Learn Quickly for Few Shot Learning / Zhenguo Li, Fengwei Zhou, Fei Chen, Hang Li // *arXiv preprint arXiv:1707.09835.* — 2017.

[10] *Wang Yu-Xiong, Hebert Martial.* Learning to learn: Model regression networks for easy small sample learning // European Conference on Computer Vision / Springer. — 2016. — Pp. 616–634.

[11] Learning to learn by gradient descent by gradient descent / Marcin Andrychowicz, Misha Denil, Sergio Gomez et al. // Advances in Neural Information Processing Systems. — 2016. — Pp. 3981–3989.

[12] Progressive neural architecture search / Chenxi Liu, Barret Zoph, Jonathon Shlens et al. // *arXiv preprint arXiv:1712.00559.* — 2017.

[13] Toward Optimal Run Racing: Application to Deep Learning Calibration / Olivier Bousquet, Sylvain Gelly, Karol Kurach et al. // *arXiv preprint arXiv:1706.03199.* — 2017.

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

[15] Accelerating neural architecture search using performance prediction / Bowen Baker, Otkrist Gupta, Ramesh Raskar, Nikhil Naik // *CoRR, abs/1705.10823.* — 2017.

[16] Learning transferable architectures for scalable image recognition / Barret Zoph, Vijay Vasudevan, Jonathon Shlens, Quoc V Le // *arXiv preprint arXiv:1707.07012.* — 2017.

[17] Efficient Architecture Search by Network Transformation / Han Cai, Tianyao Chen, Weinan Zhang et al. — 2018.

[18] *Cun Yann Le, Denker John S., Solla Sara A.* Optimal Brain Damage // Advances in Neural Information Processing Systems. — Morgan Kaufmann, 1990. — Pp. 598–605.

[19] *Hassibi Babak, Stork David G, Wolff Gregory J.* Optimal brain surgeon and general network pruning // Neural Networks, 1993., IEEE International Conference on / IEEE. — 1993. — Pp. 293–299.

[20] *Graves Alex.* Practical Variational Inference for Neural Networks // Advances in Neural Information Processing Systems 24 / Ed. by J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett et al. — Curran Associates, Inc., 2011. — Pp. 2348–2356. http://papers.nips.cc/paper/4329-practical-variational-inference-for-neural-networks.pdf.

[21] *Louizos Christos, Ullrich Karen, Welling Max.* Bayesian compression for deep learning // Advances in Neural Information Processing Systems. — 2017. — Pp. 3290–3300.

[22] Learning both Weights and Connections for Efficient Neural Network / Song Han, Jeff Pool, John Tran, William Dally // Advances in Neural Information Processing Systems 28 / Ed. by C. Cortes, N. D. Lawrence, D. D. Lee et al. — Curran Associates, Inc., 2015. — Pp. 1135–1143. http://papers.nips.cc/paper/5784-learning-both-weights-and-connections-for-efficient-neural-network.pdf.

[23] Dropout: A simple way to prevent neural networks from overfitting / Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky et al. // *The Journal of Machine Learning Research.* — 2014. — Vol. 15, no. 1. — Pp. 1929–1958.

[24] Incremental network quantization: Towards lossless cnns with low-precision weights / Aojun Zhou, Anbang Yao, Yiwen Guo et al. // *arXiv preprint arXiv:1702.03044.* — 2017.

[25] *Han Song, Mao Huizi, Dally William J.* Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding // *arXiv preprint arXiv:1510.00149.* — 2015.

[26] Learning deep resnet blocks sequentially using boosting theory / Furong Huang, Jordan Ash, John Langford, Robert Schapire // *arXiv preprint arXiv:1706.04964.* — 2017.

[27] *Maclaurin Dougal, Duvenaud David, Adams Ryan.* Gradient-based Hyperparameter Optimization through Reversible Learning // Proceedings of the 32nd International Conference on Machine Learning (ICML-15) / Ed. by David Blei, Francis Bach. — JMLR Workshop and Conference Proceedings, 2015. — Pp. 2113–2122. http://jmlr.org/proceedings/papers/v37/maclaurin15.pdf.

[28] *MacKay David J. C.* Information Theory, Inference & Learning Algorithms. — New York, NY, USA: Cambridge University Press, 2002.

[29] *Karaletsos Theofanis, Rätsch Gunnar.* Automatic Relevance Determination For Deep Generative Models // *arXiv preprint arXiv:1505.07765.* — 2015.

[30] Bayesian Optimization in High Dimensions via Random Embeddings. / Ziyu Wang, Masrour Zoghi, Frank Hutter et al. // IJCAI. — 2013. — Pp. 1778–1784.

[31] Bayesian Optimization with Tree-structured Dependencies / Rodolphe Jenatton, Cedric Archambeau, Javier González, Matthias Seeger // International Conference on Machine Learning. — 2017. — Pp. 1655–1664.

[32] Structure Optimization for Deep Multimodal Fusion Networks using Graph-Induced Kernels / Dhanesh Ramachandram, Michal Lisicki, Timothy J Shields et al. // *arXiv preprint arXiv:1707.00750*. — 2017.

[33] Raiders of the lost architecture: Kernels for Bayesian optimization in conditional parameter spaces / Kevin Swersky, David Duvenaud, Jasper Snoek et al. // *arXiv preprint arXiv:1409.4011*. — 2014.

[34] *Adams Ryan, Wallach Hanna, Ghahramani Zoubin*. Learning the structure of deep sparse graphical models // Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. — 2010. — Pp. 1–8.

[35] *Feng Jiashi, Darrell Trevor*. Learning the structure of deep convolutional networks // Proceedings of the IEEE international conference on computer vision. — 2015. — Pp. 2749–2757.

[36] *Shirakawa Shinichi, Iwata Yasushi, Akimoto Youhei*. Dynamic Optimization of Neural Network Structures Using Probabilistic Modeling // *arXiv preprint arXiv:1801.07650*. — 2018.

[37] Semi-supervised Learning with Deep Generative Models / Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, Max Welling // Advances in Neural Information Processing Systems 27 / Ed. by Z. Ghahramani, M. Welling, C. Cortes et al. — Curran Associates, Inc., 2014. — Pp. 3581–3589. http://papers.nips.cc/paper/5352-semi-supervised-learning-with-deep-generative-models.pdf.

[38] Composing graphical models with neural networks for structured representations and fast inference / Matthew Johnson, David K Duvenaud, Alex Wiltschko et al. // Advances in neural information processing systems. — 2016. — Pp. 2946–2954.

[39] *Nalisnick Eric, Smyth Padhraic*. Deep Generative Models with Stick-Breaking Priors // *arXiv preprint arXiv:1605.06197*. — 2016.

[40] *Abbasnejad M Ehsan, Dick Anthony, van den Hengel Anton*. Infinite variational autoencoder for semi-supervised learning // 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) / IEEE. — 2017. — Pp. 781–790.

[41] *Miller A. C., Foti N., Adams R. P.* Variational Boosting: Iteratively Refining Posterior Approximations // *ArXiv e-prints*. — 2016. — nov.

[42] Learning Bayesian network structure using LP relaxations / Tommi Jaakkola, David Sontag, Amir Globerson, Marina Meila // Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. — 2010. — Pp. 358–365.

[43] *Alvarez-Melis David, Jaakkola Tommi S.* Tree-structured decoding with doubly-recurrent neural networks. — 2016.

[44] *Alain Guillaume, Bengio Yoshua.* Understanding intermediate layers using linear classifier probes // *arXiv preprint arXiv:1610.01644.* — 2016.

[45] *Teerapittayanon Surat, McDanel Bradley, Kung HT.* Branchynet: Fast inference via early exiting from deep neural networks // Pattern Recognition (ICPR), 2016 23rd International Conference on / IEEE. — 2016. — Pp. 2464–2469.

[46] *Friesen Abram L, Domingos Pedro.* Deep Learning as a Mixed Convex-Combinatorial Optimization Problem // *arXiv preprint arXiv:1710.11573.* — 2017.

[47] *Kristiansen Gus, Gonzalvo Xavi.* EnergyNet: Energy-based Adaptive Structural Learning of Artificial Neural Network Architectures // *arXiv preprint arXiv:1711.03130.* — 2017.

[48] Pathnet: Evolution channels gradient descent in super neural networks / Chrisantha Fernando, Dylan Banarse, Charles Blundell et al. // *arXiv preprint arXiv:1701.08734.* — 2017.

[49] *Veniat Tom, Denoyer Ludovic.* Learning time-efficient deep architectures with budgeted super networks // *arXiv preprint arXiv:1706.00046.* — 2017.

[50] *Chen Tianqi, Goodfellow Ian, Shlens Jonathon.* Net2net: Accelerating learning via knowledge transfer // *arXiv preprint arXiv:1511.05641.* — 2015.

[51] Forward thinking: Building and training neural networks one layer at a time / Chris Hettinger, Tanner Christensen, Ben Ehlert et al. // *arXiv preprint arXiv:1706.02480.* — 2017.

[52] *Miranda Conrado S, Von Zuben Fernando J.* Reducing the Training Time of Neural Networks by Partitioning // *arXiv preprint arXiv:1511.02954.* — 2015.