Глава 1
Постановка задачи последовательного выбора моделей

Проблема выбора структуры модели является фундаментальной в области машинного обучения интеллектуального анализа данных. Проблема выбора структуры модели глубокого обучения формулируется следующим образом: решается задача классификации или регрессии на заданной или пополняемой выборке $\mathfrak{D}$. 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 по параметрам нелинейный функций. Под структурой модели понимается значения структурных параметров модели, т.е. величин, задающих вид итоговой суперпозиции.

Формализуем описанную выше задачу.

Определение 1. *Объектом* назовем пару $(\mathbf{x}, y), \mathbf{x} \in \mathbb{X} = \mathbb{R}^n, y \in \mathbb{Y}$. В случае задачи классификации $\mathbb{Y}$ является распределением вероятностей принадлежности объекта $\mathbf{x} \in \mathbb{X}$ множеству классов $\{1, \ldots, Z\}$: $\mathbb{Y} \subset [0,1]^Z$, где Z — число классов. В случае задачи регрессии $\mathbb{Y}$ является некоторым подмножеством вещественных чисел $y \in \mathbb{Y} \subseteq \mathbb{R}$. Объект состоит из двух частей: $\mathbf{x}$ соответствует *признаковому описанию объекта*, y — *метке объекта*.

Задана простая выборка

$$\mathfrak{D} = \{(\mathbf{x}_i, y_i)\}, i = 1, \ldots, m, \tag{1.1}$$

состоящая из множества объектов

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

Определение 2. *Моделью* $\mathbf{f}(\mathbf{w}, \mathbf{x})$ назовем дифференцируемую по параметрам $\mathbf{w}$ функцию из множества признаковых описаний объекта во множество меток:

$$\mathbf{f} : \mathbb{X} \times \mathbb{W} \to \mathbb{Y},$$

где $\mathbb{W}$ — пространство параметров функции $\mathbf{f}$.

Специфика задачи выбора модели *глубокого обучения* заключатеся в том, что модели глубокого обучения могут иметь значительное число параметров, что приводит к неприменимости ряда методов оптимизации и выбора модели. Перейдем к формальному описанию параметрического семейства моделей глубокого обучения.

Определение 3. Пусть задан направленный граф (V, E). Пусть для каждого ребра $(j,k) \in E$ определен вектор базовых функций мощности $K^{j,k}$: $\mathbf{g}^{j,k} = [\mathbf{g}_0^{j,k}, \ldots, \mathbf{g}_{K^{j,k}}^{j,k}]$. Пусть также для каждой вершины $v \in V$ определена функция агрегации $\mathbf{agg}_v$. Граф (V, E) в совокупности со множестом векторов базовых

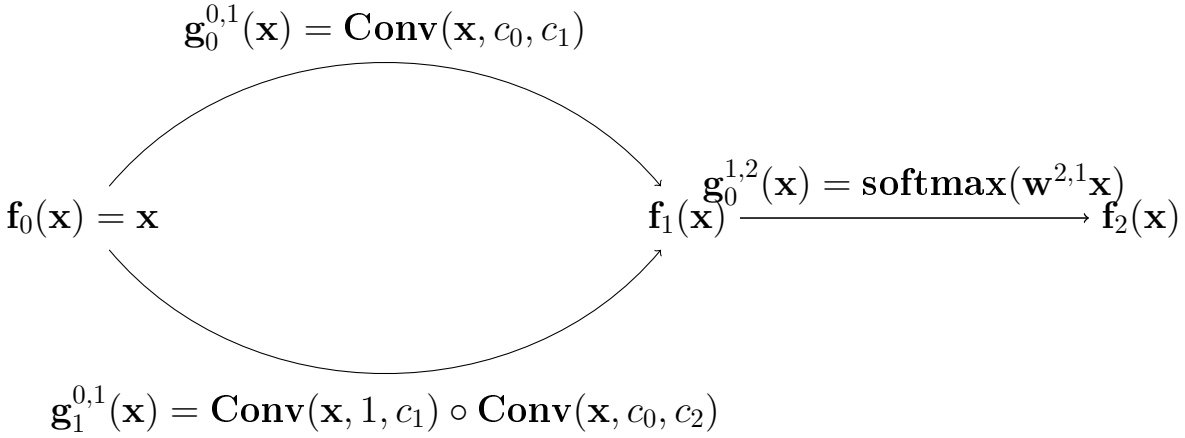


Рис. 1.1. Пример параметрического семейства моделей глубокого обучения: семейство описывает сверточную нейронную сеть.

функций $\{\mathbf{g}^{j,k}, (j,k) \in E\}$ и множеством функций агрегаций $\{\mathbf{agg}_v, v \in V\}$ называется *параметрическим семейством моделей* $\mathfrak{F}$, если функция, задаваемая как

$$\mathbf{f}_k(\mathbf{x}) = \mathbf{agg}_k\left(\left\{\langle \boldsymbol{\gamma}^{j,k}, \mathbf{g}^{j,k}\rangle\left(\mathbf{f}_j(\mathbf{x})\right) | j \in \mathrm{Adj}(v_k)\right\}\right), \quad \mathbf{f}_0(\mathbf{x}) = \mathbf{x} \tag{1.2}$$

является моделью при любых значениях векторов, $\boldsymbol{\gamma}^{j,k} \in [0,1]^{K^{j,k}}$.

Примером функций агрегации выступают функции суммы и конкатенации векторов.

Определение 4. Функции $\mathbf{f}_1, \ldots, \mathbf{f}_{|V|}$ из (1.2) назовем называть *слоями или подмоделями* модели $\mathbf{f}$.

Пример параметрического семейства моделей, которое описывает сверточную нейронную сеть, представлена на Рис. 1.1. Семейство задает множество моделей с двумя операциями свертки с одинаковым размером фильтра c_0 и различным числом каналов c_1 и c_2. Единичная свертка с c_1 каналами $\mathbf{Conv}(\mathbf{x}, c_1, 1)$ требуется для выравнивания размерностей скрытых слоев. Каждая модель параметрического семейства задается формулой:

$$\mathbf{f} = \mathbf{agg}_2\left(\left\{\boldsymbol{\gamma}_0^{1,2}\mathbf{g}_0^{1,2}\left(\mathbf{agg}_1\left(\{\boldsymbol{\gamma}_0^{0,1}\mathbf{g}_0^{0,1}(\mathbf{x}), \boldsymbol{\gamma}_1^{0,1}\mathbf{g}_1^{0,1}(\mathbf{x})\}\right)\right)\right\}\right).$$

Положим, что функции агрегации $\mathbf{agg}_1, \mathbf{agg}_2$ являются операциями суммы. Заметим, что к вершине "2" ведет только одно ребро, поэтому операцию суммы можно опустить. Итоговая формула модели задается следующим образом:

$$\mathbf{f} = \boldsymbol{\gamma}_0^{1,2}\mathbf{softmax}\left(\boldsymbol{\gamma}_0^{0,1}\mathbf{Conv}(\mathbf{x}, c_0, c_1)(\mathbf{x}) + \boldsymbol{\gamma}_1^{0,1}\mathbf{Conv}(\mathbf{x}, 1, c_1) \circ \mathbf{Conv}(\mathbf{x}, c_0, c_2)(\mathbf{x})\right).$$

Определение 5. *Параметрами* модели $\mathbf{f}$ из параметрического семейства моделей $\mathfrak{F}$ назовем конкатенацию векторов параметров всех базовых функций $\{\mathbf{g}^{j,k} | (j,k) \in E\}, \mathbf{w} \in \mathbb{W}$. Вектор параметров базовой функции $\mathbf{g}_l^{j,k}$ будем обозначать как $\mathbf{w}_l^{j,k}$.

Определение 6. Структурой $\mathbf{\Gamma}$ модели $\mathbf{f}$ из параметрического семейства моделей $\mathfrak{F}$ назовем конкатенацию векторов $\boldsymbol{\gamma}^{j,k}$. Множество всех возможных значений структуры $\mathbf{\Gamma}$ будем обозначать как $\mathbb{\Gamma}$. Векторы $\boldsymbol{\gamma}^{j,k}, (j,k) \in E$ назовем *структурными параметрами модели.*

Определение 7. *Параметризацией* множества моделей M назовем параметрическое семейство моделей $\mathfrak{F}$, такое что для каждой модели $\mathbf{f} \in M$ существуют значение структуры модели $\mathbf{\Gamma}$ при котором функция $\mathbf{f}$ совпадает с функцией (1.2).

TODO Можно доказать, что для любого множества хороших (дифференцируемых?) моделей существует параметризация.

Рассмотрим варианты ограничений, которые накладываются на структурные параметры $\boldsymbol{\gamma}_{j,k}$ параметрического семейства моделей. Цель данных ограничений — уточнение архитектуры модели глубокого обучения, которую требуется получить.

1. Структурные параметры лежат на веришнах булевого куба: $\boldsymbol{\gamma}_{j,k} \in \{0,1\}^{K^{j,k}}$. Структурные параметры $\boldsymbol{\gamma}_{j,k}$ интерпретируются как параметр включения или выключения компонент вектора базовых функций $\mathbf{g}^{j,k}$ в итоговую модель.
2. Структурные параметры лежат внутри булевого куба: $\boldsymbol{\gamma} \in [0,1]^{K^{j,k}}$. Релаксированная версия предыдущих ограничений, позволяющая проводить градиентную оптимизацию для структурных параметров.
3. Структурные параметры лежат на веришнах симплекса: $\boldsymbol{\gamma}_{j,k} \in \bar{\Delta}^{K^{j,k}-1}$. Каждый вектор структурных параметров $\boldsymbol{\gamma}_{j,k}$ имеет только одну ненулевую компоненту, определяющую какая из базовых функций $\mathbf{g}^{j,k}$ войдет в итоговую модель. Примером параметрического семейства моделей, требующим такое ограничение является семейство полносвязанных нейронных сетей с одним скрытым слоем и двумя значениями коилчества нейронов на скрытом слое. Схема семейства представлена на Рис. 1.5. Данное семейство можно представить как семейство с двумя базовыми функциями вида $\mathbf{g} = \boldsymbol{\sigma}(\mathbf{wx})$, где матрицы параметров каждой из функций $\mathbf{g}^{1,1}, \mathbf{g}^{1,2}$ имеют фиксированное число нулевых столбцов. Количество этих столбцов определяет размерность итогового скрытого пространства (невырожденного?) или числа нейронов на скрытом слое.
4. Структурные параметры лежат внутри симплекса: $\boldsymbol{\gamma}_{j,k} \in \Delta^{K^{j,k}-1}$. Релаксированная версия предыдущих ограничений, позволяющая проводить градиентную оптимизацию для структурных параметров. Значений стуктурных параметров $\boldsymbol{\gamma}_{j,k}$ интерпретируются как вклад каждой компоненты вектора базовых функций $\mathbf{g}^{j,k}$ в итоговую модель.

Пример, иллюстрирующий представленные выше ограничения, изображен на Рис. 1.2. В данной работе рассматривается случай, когда на структурные параметры наложено ограничение 4. Данные ограничения позволяют решать задачу выбора модели как для семейства моделей типа многослойных полносвяз-

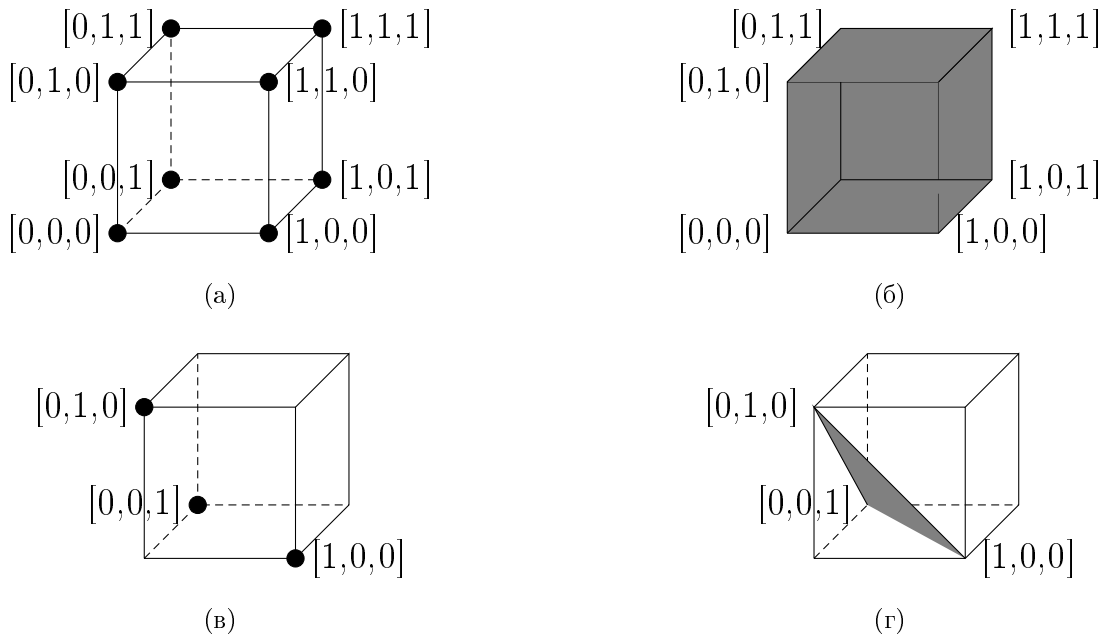


Рис. 1.2. Примеры ограничений для одного структурного параметра $\boldsymbol{\gamma}, |\boldsymbol{\gamma}| = 3$. а) структурный параметр лежит на вершинах куба, б) структурный параметр лежит внутри куба, в) структурный параметр лежит на верщинах симплекса, г) структурный параметр лежит внутри симплекса.

ных нейронных сетей, так и для более сложных параметрических семейств [1].

Для дальнейшей постановки задачи введем понятие вероятностной модели, и связанных с ним определений. Будем полагать, что для параметров модели $\mathbf{w}$ и структуры $\boldsymbol{\Gamma}$ задано распределение $p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h})$, соответствующее предположениям о распределении структуры и параметров.

Определение 8. *Гиперпараметрами* $\mathbf{h} \in \mathbb{H}$ модели назовем параметры распределения $p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h})$.

Определение 9. *Априорным распределением* параметров и структуры модели назовем вероятностное распределение, соответствующее предположениям о распределении параметров модели:

$$p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h}) : \mathbb{W} \times \Delta\boldsymbol{\Gamma} \times \mathbb{H} \to \mathbb{R}^+,$$

где $\mathbb{W}$ — множество значений параметров модели.

Одной из возможных частных постановок задачи выбора структуры модели является *двусвязный байесовский вывод.* На *первом уровне* байесовского вывода находится апостериорное распределение параметров.

Определение 10. *Апостериорным распределением* назовем распределение вида

$$p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{y}, \mathbf{X}, \mathbf{h}) = \frac{p(\mathbf{y}|\mathbf{w}, \boldsymbol{\Gamma}, \mathbf{X}, \mathbf{h}) p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h})}{p(\mathbf{y}|\mathbf{X})} \propto p(\mathbf{y}|\mathbf{w}, \boldsymbol{\Gamma}, \mathbf{X}, \mathbf{h}) p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h}). \quad (1.3)$$

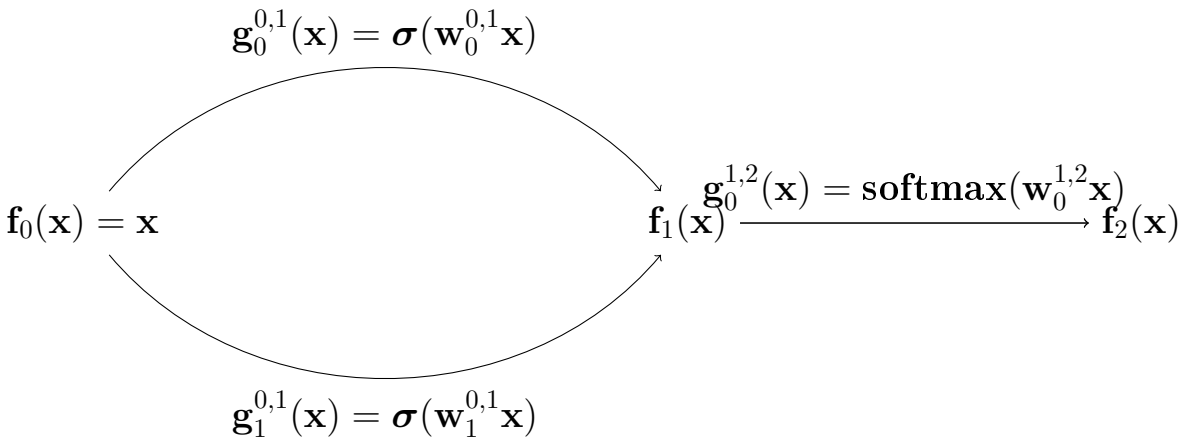


Рис. 1.3. Пример параметрического семейства моделей глубокого обучения: семейство описывает многослойную полносвязную нейронную сеть с одним скрытым слоем и нелинейной функцией активации $\boldsymbol{\sigma}$.

Определение 11. *Вероятностной моделью глубокого обучения* назовем совместное распределение вида

$$p(y, \mathbf{w}, \boldsymbol{\Gamma}|\mathbf{x}, \mathbf{h}) = p(y|\mathbf{x}, \mathbf{w}, \boldsymbol{\Gamma})p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h}) : \mathbb{Y} \times \mathbb{W} \times \Delta\boldsymbol{\Gamma} \times \mathbb{R}^+.$$

Определение 12. *Функцией правдоподбия выборки* назовем величину

$$p(y|\mathbf{X}, \mathbf{w}, \boldsymbol{\Gamma}) : \mathbb{Y} \times \mathbb{X} \times \mathbb{W} \times \lneqq \to \mathbb{R}^+.$$

Для каждой модели определена функция правдоподобия $p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\Gamma})$.

На *втором уровне* байесовского вывода осуществляется выбор модели на основе правдоподобия модели.

Определение 13. *Правдоподобием модели* назовем величину

$$p(y|\mathbf{X}, \mathbf{h}) = \int_{\mathbf{w},\boldsymbol{\Gamma}} p(\mathbf{y}|\mathbf{X}, \mathbf{w}, \boldsymbol{\Gamma})p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h})d\mathbf{w}d\boldsymbol{\Gamma}. \tag{1.4}$$

Получение значений апостериорного распределения и правдоподобия модели сетей глубокого обучения является вычислительно сложной процедурой. Для получения оценок на данные величины используют методы, такие как аппроксимация Лапласа [2] и вариационная нижняя оценка [3]. В данной работе в качестве метода получения оценок правдоподобия модели выступает вариационное распределение.

Определение 14. *Вариационным распределением* назовем параметрическое распределение $q(\mathbf{w}, \boldsymbol{\Gamma})$, являющееся приближением апостериорного распределения параметров и структуры $p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{X}, \mathbf{y}, \mathbf{h})$.

Определение 15. *Вариационными параметрами* модели $\boldsymbol{\theta} \in \mathbb{R}^u$ назовем параметры вариационного распределения q.

Определение 16. Пусть задано вариационное распределения q. *Функцией потерь* $L(\boldsymbol{\theta}|\mathbf{h}, \mathbf{X}, \mathbf{y})$ для модели $\mathbf{f}$ назовем дифференцируемую функцию, принимаемую за качество модели на обучающей выборки при параметрах модели, получаемых из распределения q.

В качестве функции L может выступать минус логарифм правдоподобия выборки $\log p(y|\mathbf{X}, \mathbf{w}, \boldsymbol{\Gamma})$ и логарифм апостериорной вероятности $\log p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{y}, \mathbf{X}, \mathbf{h})$ параметров и структуры модели на обучающей выборке.

Определение 17. Пусть задано вариационное распределения q и функция потерь L. *Функцией валидации* $Q(\mathbf{h}|\boldsymbol{\theta}, \mathbf{X}, \mathbf{y})$ для модели $\mathbf{f}$ назовем дифференцируемую функцию, принимаемую за качество модели при векторе $\boldsymbol{\theta}$, заданном неявно.

В данной работе задача выбора структуры модели и параметров модели ставится как двухуровневая задача оптимизации:

$$\mathbf{h}^* = \arg\min_{\mathbf{h} \in \mathbb{H}} Q(\mathbf{h}|\boldsymbol{\theta}^*, \mathbf{X}, \mathbf{y}), \tag{1.5}$$

где $\boldsymbol{\theta}^*$ — решение задачи оптимизации

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta} \in \mathbb{R}^u} L(\boldsymbol{\theta}|\mathbf{h}, \mathbf{X}, \mathbf{y}). \tag{1.6}$$

Определение 18. *Выбором модели* $\mathbf{f}$ назовем решение двухуровневой задачи оптимизации (1.5).

Рассмотрим для примера базовый вариант выбора модели с применением функций q, L, Q.

Пример 1. Будем полагать, что задано разбиение выборки на обучающую $\mathfrak{D}_{\text{train}}$ и валидационную $\mathfrak{D}_{\text{valid}}$ части. Положим в качестве вариационных параметров $\boldsymbol{\theta}$ параметры $\mathbf{w}$ и структуры $\boldsymbol{\Gamma}$ модели:

$$\boldsymbol{\theta} = [\mathbf{w}, \boldsymbol{\Gamma}].$$

Пусть также задано априорное распределение $p(\mathbf{w}, \boldsymbol{\Gamma}|\mathbf{h})$. Положим в качестве функции L минус логарифм апостериорной вероятности модели:

$$L = -\sum_{\mathbf{x}, y \in \mathfrak{D}_{\text{train}}} \log p(y, \mathbf{w}, \boldsymbol{\Gamma}|\mathbf{x}).$$

Положим в качестве функции Q минус правдоподобие выборки при условии параметров $\mathbf{w}$ и структуры $\boldsymbol{\Gamma}$:

$$Q = -\sum_{\mathbf{x}, y \in \mathfrak{D}_{\text{valid}}} \log p(y|\mathbf{x}, \mathbf{w}, \boldsymbol{\Gamma}).$$

Оптимизация параметров и структуры производится по обучающей выборке. Гиперпараметры $\mathbf{h}$ выступают в качестве регуляризатора, чья оптимизация производится по валидационной выборке. Подобная оптимизация позволяет предотвратить переобучение модели [4].

Частным случаем задачи выбора структуры глубокой сети является выбор обобщенно-линейных моделей. Отдельные слои полносвязанных нейросетей являются обобщенно-линейными модели. Задачу выбора обобщенно-линейной моделей сводится к задаче выбора признаков, методы решения которой делятся на три группы [5]:

1. Фильтрационные методы. Не используют какой-либо информации о модели, а отсекают признаки только на основе статистических показателей, учитывающих взаимосвязь признаков и меток объектов.
2. Оберточные методы анализируют подмножества признаков. Они выбирают не признаки, а подмножества признаков, что позволяет учесть корреляция признаков.
3. Методы погружения оптимизируют модели и проводят выбор признаков в единой процедуре, являясь комбинацией предыдущих типов отбора признаков.

1.1. Критерии выбора модели глубокого обучения

В данном разделе рассматриваются различные критерии выбора моделей глубокого обучения, соответствующие функции валидации Q. В данной работе в качестве критерия выбора модели предлагается субоптимальная сложность модели. Под сложностью модели понимается *правдоподобие модели* (1.4), являющееся байесовской интерпретацией *мнимальной длины описания* [6], т.е. минимальное количество информации, которое требуется передать о модели и о выборке:

$$\mathrm{MDL}(\mathbf{y}, \mathbf{f}) = \mathrm{Len}(\mathbf{y}|\mathbf{w}^*, \mathbf{f}) + \mathrm{COMP}(\mathbf{f}), \tag{1.7}$$

где $\mathrm{Len}(\mathbf{y}|\mathbf{w}^*, \mathbf{f})$ — *длина описания* матрицы $\mathbf{y}$ с использованием модели $\mathbf{f}$ и оценки вектора параметров $\mathbf{w}^*$, полученных методом наибольшего правдоподобия, а $\mathrm{COMP}(\mathbf{f})$ — величина, характеризующая *параметрическую сложность* модели, т.е. способность модели описать произвольную выборки из $\mathbb{X}$ [6].

В общем случае правдоподобие модели является трудновычислимым. Для получения оценки правдоподобия используются вариационные методы получения оценки правдоподобия [7], основанные на аппроксимации неизвестного другим заданным распределением. Под субоптимальной сложностью понимается вариационная оценка правдоподобия модели. Альтернативной величиной, характеризующей сложность модели, выступает рademахеровская сложность (1.13). Данная величина используется как критерий для продолжения итеративного построения модели в [8].

В работе [9] рассматривается ряд критериев сложности моделей глубокого обучения и их взаимосвязь. В работе [10] в качестве критерия сложности модели выступает показатель нелинейности, характеризуемый степенью полинома Чебышева, аппроксимирующего функцию. В работе [11] анализируется показатель избыточности параметров сети. Утверждается, что по небольшому набору

параметров в глубокой сети с большим количеством избыточных параметров возможно спрогнозировать значения остальных. В работе [12] рассматривается показатель робастности моделей, а также его взаимосвязь с топологией выборки и классами функций, в частности рассматривается влияние функции ошибки и ее липшицевой константы на робастность моделей. Схожие идеи были рассмотрены в работе [13], в которой исследуется устойчивость классификации модели под действием шума. В ряде работ [14, 7, 2, 15, 16, 17] в качестве критерия выбора модели выступает правдоподобие модели. В работах [2, 15, 16, 17] рассматривается проблема выбора модели и оценки гиперапараметров в задачах регрессии. Альтернативным критерием выбора модели является минимальная длина описания [6], являющаяся показателем статистической сложности модели и заданной выборки. В работе [6] рассматриваются различные модификации и интерпретации минимальной длины описания, в том числе связь с правдоподобием модели.

Одним из методов получения приближенного значения правдоподобия модели является вариационный метод получения нижней оценки правдоподобия [7]. В работе [18] рассматривается стохастическая версия вариационного метода. В [3] рассматривается алгоритм получения вариационной нижней оценки правдоподобия для оптимизации гиперпараметров моделей глубокого обучения. В работе [19] рассматривается взаимосвязь градиентных методов получения вариационной нижней оценки интеграла с методом Монте-Карло. В [20] рассматривается стохастический градиентный спуск в качестве оператора, порождающего распределение, аппроксимирующее апостериорное распределение параметров модели. В работе отмечается, что стохастический градиентный спуск не оптимизирует вариационную оценку правдоподобия, а приближает ее только до некоторого числа итераций оптимизации. Схожий подход рассматривается в работе [21], где также рассматривается стохастический градиентный спуск в качестве оператора, порождающего апостериорное распределение параметров. В работе [22] предлагается модификация стохастического градиентного спуска, аппроксимирующая апостериорное распределение.

Альтернативным методом выбора модели является выбор модели на основе скользящего контроля [23, 2]. Проблемой такого подхода является высокая вычислительная сложность [24, 25]. В работах [26, 27] рассматривается проблема смещения оценок качества модели и гиперпараметров, получаемых при использовании k-fold метода скользящего контроля, при котором выборка делится на k-частей с обучением на $k-1$ части и валидацией результата на оставшейся части выборки.

1.2. Оптимизация параметров в задаче выбора структуры модели

Один из подходов к выбору оптимальной модели заключается в итеративном удалении наименее информативных параметров модели. В данном разделе собраны методы оптимизации структуры существующей модели.

Алгоритмы прореживания параметров модели. В [28] предлагается удалять неинформативные параметры модели. Для этого находится точка оптимума $\boldsymbol{\theta}^*$ функции L, и производится разложение функции L в ряд Тейлора в окрестности $\boldsymbol{\theta}^*$:

$$L(\boldsymbol{\theta}^* + \Delta\boldsymbol{\theta}) - L(\boldsymbol{\theta}^*) = \frac{1}{2}\Delta\boldsymbol{\theta}^{\mathsf{T}}\mathbf{H}\Delta\boldsymbol{\theta} + o(||\Delta\boldsymbol{\theta}||^3), \tag{1.8}$$

где $\mathbf{H}$ — гессиан функции L. Связь между параметрами не учитывается, поэтому гессиан матрицы L является диагональным. Положим в качестве операции удаления параметра замену его значения на ноль. Выбор наиболее неинформативного параметра сводится к задаче условной минимизации (1.8) при условиях вида

$$\theta_i + \Delta\theta_i = 0, \quad \theta_i \in \boldsymbol{\theta}.$$

В результате решения данной задачи минимизации каждому параметру определяется функция выпуклости

$$\text{saliency}(\theta_i) = \frac{\theta_i^2}{2H_{i,i}}.$$

Данная функция характеризует информативность параметра.

В [29] было предложено развитие данного метода. В отличие от [28] не вводится предположений о диагональности гессиана функции ошибок, поэтому удаление неинформативных параметров модели производится точнее. Для получения оценок гессиана и его обратной матрицы применяется итеративный алгоритм.

Алгоритмы компрессии параметров модели. В [30, 31, 32] предлагаются методы компрессии параметров сетей глубокого обучения. Основным отличием задачи прореживания от задачи компрессии выступает эксплуатационное требование: если прореживание используется для получения оптимальной и наиболее устойчивой модели, то компрессия производится для уменьшения потребляемых вычислительных ресурсов при сохранении основных эксплуатационных характеристик исходной модели [31]. В [32] предлагается итеритавиное использование регуляризации типа DropOut [33] для прореживания модели. В [30, 31] используются методы снижения вычислительной точности представления парамеров модели на основе кластеризации параметров $\mathbf{w}$ модели: вместо значений параметров предлагается хранить идентификатор кластера, соответствующего параметру, что существенно снижает количество требуемой памяти. В [31] предлагается метод компрессии, основанный на кластеризации значений параметров модели и представлении их в сжатом виде на основе кодов Хаффмана.

Байесовские методы прореживания параметров модели. Байесовский подход к порождению и выбору моделей заключается в использовании

вероятностных предположений о распределении параметров и структуры в параметрических семействах моделей. Такой подход позволяет учитывать при выборе моделей не только эксплуатационные критерии качества модели, такие как точность итоговой модели и количество параметров в ней, но и некоторые статистические характеристики модели.

В работе [4] рассматривается задача оптимизации гиперпараметров. Авторы предлагают оптимизировать константы l_2-регуляризации отдельно для каждого параметра модели, проводится параллель с методами автоматического определения релевантности параметров (англ. automatic relevance determination, ARD) [14]. Идея автоматического определения релевантности заключается в выборе оптимальных начений гиперпараметров $\mathbf{h}$ с дальнейшим удалением неинформативных параметров. Неинформативными параметрами являются те параметры, которые с высокой вероятностью равны нулю относительно априорного или апостериорного распределения.

В работе [3] был предложен метод, основанный на получении вариационной нижней оценки правдоподобия модели. В качестве критерия информативности параметра выступает отношение вероятности нахождения параметра в пределах апостериорного распределения к вероятности равенства параметра нулю:

$$\left|\frac{\mu_j}{\sigma_j}\right|,$$

где μ_j, σ_j — среднее и дисперсия аппроксимирующего распределения q для параметра w_j.

Идея данного метода была развита в [34], где также используются вариационные методы. В отличие от [3], в [34] рассматривается ряд априорных распределений параметров, позволяющих прореживать модели более эффективно:

1. Нормальное распределение с лог-равномерным распределением дисперсии. Для каждого параметра $w \in \mathbf{w}$ задается группа параметров $\boldsymbol{\omega} \in \boldsymbol{\Omega}$, где $\boldsymbol{\Omega}$ — множество всех групп параметров:

$$p(\mathbf{w}, \mathbf{s}) \propto \prod_{\boldsymbol{\omega} \in \boldsymbol{\Omega}} \frac{1}{|\boldsymbol{\omega}|} \prod_{w \in \boldsymbol{\omega}} \mathcal{N}(w|\mathbf{0}, \boldsymbol{\omega}).$$

2. Априорное распределение задается произведением двух случайных величин $s_{\text{general}}, s_{jk}$ с половинным распределением Коши $\mathcal{C}^+$: одно ответственно за отдельный параметр, другое — за общее распределение параметров:

$$s_{\text{general}} \sim \mathcal{C}^+(0, \lambda), \quad s_{jk} \sim \mathcal{C}^+(0, 1), \quad \hat{w}_{jk} \sim \mathcal{N}(0, 1), \quad w_{jk} \sim \hat{w}_{jk} s_j s_{\text{general}},$$

где $\lambda \in \mathbf{h}$ — параметр распределения.

1.3. Оптимизация гиперпараметров модели

В данном разделе рассматриваются работы, посвященные методам оптимизации гиперпараметров. Методы, используемые для оптимизации гиперпара-

метров моделей глубокого обучения должны быть эффективными по вычислительным затратам в силу высокой вычислительной сложности оптимизации параметров модели. В [35, 36] рассматривается задача оптимизации гиперпараметров стохастическими методами. В [35] проводится сравнение случайного поиска значений гиперпараметров с переборным алгоритмом. В [36] производится сравнение случайного поиска и алгоритмов, основанных на вероятностных моделях.

Градиентные методы оптимизации гиперпараметров.

Определение 19. Назовем *оператором оптимизации* алгоритм T выбора вектора параметров $\boldsymbol{\theta}'$ по параметрам предыдущего шага $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}' = T(\boldsymbol{\theta}|L, \mathbf{y}, \mathbf{X}, \mathbf{h}, \boldsymbol{\beta}), \tag{1.9}$$

где $\boldsymbol{\beta}$ — параметры оператора оптимизации или *метапараметры*.

Пример схожего описания оптимизации модели с использованием оператора оптимизации можно найти в [20].

Частным случаем оператора оптимизации является оператор стохастического спуска:

$$T(\boldsymbol{\theta}|L, \mathbf{X}, \mathbf{y}, \mathbf{h}, \boldsymbol{\beta}) = \boldsymbol{\theta} - \beta_{\text{lr}} \nabla L(\boldsymbol{\theta}|\mathbf{h}, \hat{\mathbf{X}}, \hat{\mathbf{y}}), \tag{1.10}$$

где β_{lr} — шаг градиентного спуска, $\hat{\mathbf{y}}, \hat{\mathbf{X}}$ — случайная подвыборка заданной мощности выборки $\mathfrak{D}$.

В случае оптимизации гиперпараметров оператор оптимизации применяется не к вариационным параметрам $\boldsymbol{\theta}$, а к гиперпараметрам $\mathbf{h}$:

$$\mathbf{h} = T(\mathbf{h}|Q, \mathbf{X}, \mathbf{y}, \boldsymbol{\theta}^*, \boldsymbol{\beta}), \tag{1.11}$$

где $\boldsymbol{\theta}^*$ — вариационные параметры, полученые в ходе решения задачи оптимизации (1.6).

В случае, если для решения задачи (1.6) применяется несколько шагов оператора оптимизации (1.9), $\boldsymbol{\theta}^*$ рассматривается как рекурсивная функция от начального приближения вариационных параметров $\boldsymbol{\theta}^0$ и вектора гиперпараметров $\mathbf{h}$:

$$\boldsymbol{\theta}^* = T \circ \cdots \circ T(\boldsymbol{\theta}^0|L, \mathbf{X}, \mathbf{y}, \mathbf{h}, \boldsymbol{\beta}) = \boldsymbol{\theta}^*(\boldsymbol{\theta}^0, \mathbf{h}). \tag{1.12}$$

Решение задачи оптимизации (1.11) при (1.12) является вычислительно сложным, поэтому применяются методы, аппроксимирующие применение градиентных методов при (1.12).

В [37] рассматривается оптимизация гиперпараметров градиентными методами для квадратичной функции потерь. В [4] в качестве оператора оптимизации гиперпараметров выступает метод градиентного спуска с моментом. Показано, что использование момента значительно снижает количество вычислительных ресурсов, требуемых для проведения оптимизации. В [38] предлагается

аппроксимация градиентного метода, использующая предположение о линейности функции (1.12) от начального приближения $\boldsymbol{\theta}^0$. В [39] предлагается использовать численные методы для приближенного вычисления оператора оптимизации гиперпараметров. В [40] в качестве аппроксимации (1.12) предлагается рассматривать только последний шаг оптимизации:

$$\boldsymbol{\theta}^* \approx T(\boldsymbol{\theta}^{\eta-1}|L, \mathbf{y}, \mathbf{X}, \mathbf{h}, \boldsymbol{\beta}),$$

где η — число шагов оптимизации.

Суррогатный выбор моделей. Идея суррогатных моделей заключается в аппроксимации модели или параметрического семейства моделей вычислительно менее сложной функцией.

В работе [41] предлагается моделировать качество модели Q (1.4) гауссовым процессом, параметрами которого выступают гиперпараметры исходной модели.

Одна из основных проблем использования гауссового процесса как суррогатной модели — кубическая сложность оптимизации. В работе [42] предлагается использовать случайные подпространства гиперпараметров для ускоренной оптимизации. В работе [43] предлагается кобминация из множества гауссовых моделей и линейной модели, позволяющая модели нелинейные зависимости гиперпараметров, а также существенно сократить сложность оптимизации.

В работе [44] предлагается рассматривать RBF-модель для аппроксимации качества Q исходной модели, что позволяет ускорить процесс оптимизации суррогатной модели. В [45] рассматривается глубокая нейронная сеть в качестве суррогатной функции. Вместо интеграла правдоподобия (1.4), который оценивается в случае использования гауссового процесса в качестве суррогата, используется максимум апостериорной веротяности (1.3).

Одним из параметров гауссовых процессов является функиия ядра гауссового процесса, полностью определяющая процесс в случае нулевого среднего. В работе [46] предлагается функция ядра, определенная на графах:

$$k(v_1, v_2) = r(d(v_1, v_2)),$$

где d — геодезическое расстояние между вершинами графа, r — некоторая вещественная функция, $v_1, v_2 \in V$.

В работе [47] рассматривается задач выбора структуры нейросети. Предлагается метод построения ковариационной функции для сравнения разнородных графов, соответствующих разным моделям нейронных сетей. Ковариационная функция основывается на метрике, заданной на некоторых характеристиках $g(v)$ вершин, возможно не определенных для сравниваемых графов:

$$d_v\big((V_1, E_1), (V_2, E_2)\big) = \begin{cases} 0, v \notin V_1, v \notin V_2, \\ \lambda_1\sqrt{2}\sqrt{1 - \cos(\pi\lambda_2 \frac{g_1 - g_2}{\sup(g) - \inf(g)})}, v \in V_1, v \in V_2, \\ \lambda_1 \text{ иначе}, \end{cases}$$

где λ_1, λ_2 — параметры функции d_v.

1.4. Порождениие и выбор структуры модели глубокого обучения

В данном разделе рассматриваются работы, посвященные порождению и модификации структуры моделей. В отличие от работ, описанных в предыдущих разделах, в следующих работах рассматриваемым объектом является не отдельный параметр, а подмодель или группа параметров, входящая в эту подмодель.

Графовое представление структуры модели. Одним из возможных представлений структуры моделей глубокого обучения 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 соответсвтует походам, описанным в [48], а также в библиотеках типа TensorFlow [49], Theano [50], Pytorch [51], 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 ряде работ, посвященных байесовской оптимизации [45, 44, 41], модель рассматривается как черный ящик, над которым производится ограниченный набор операций типа "произвести оптимизацию параметров" и "предсказать значение зависимой переменной по независимой переменной и параметрам модели". Подход, описанный в данных работах, также коррелирует с библиотеками машинного обучения, такими как Weka [52], RapidMiner [53] или sklearn [54], в которых модель машинного обучения рассматривается как черный ящик.

В [55] представлен обзор по графовому описанию моделей глубокого обучения, предлагается метод фомрального описания графовых сетей (англ. Graph Network), являющийся обобщением предложенных ранее графовых описаний моделей.

В работе [56] рассматриваются подходы к порождению моделей глубокого обучения. Предлагается формализация пространства поиска и формальное описание элементов пространства моделей. Приведем пример описания параметрического семейства моделей, соответствующего схеме из Рис. 1.1 при условии, что структурные параметры $\boldsymbol{\gamma}$ имеют только одну ненулевую компоненту:

```
(Concat
  OR(
    (Conv2D [c_0] [c_1] [1],
    (Concat(
      (Conv2D [c_0] [c_2] [1],
      (Conv2D [1] [c_1] [1])),
  (Affine [10]),
  (SoftMax)).
```

TODO: в статье нет сотфмакса

Прогнозирование графовых структур. В работе [57] предлагается метод прогнозирования графовой структуры на основе линейного программирования. Предлагается свести проблему поиска графовой структуры к комбинаторной проблеме. В работе [58] предлагается метод прогнозирования структур деревьев, основанный на дважды-рекуррентных нейросетях (англ. doubly-reccurent), т.е. на сетях, отдельно прогнозирующих глубину и ширину уровней деревьев.

Стохастическое порождение структур. Одним из возможных направлений для порождения структр моделей глубокого обучения выступает стохастическое порождение структур. Данный тип порождения предполагает, что структуры порождаются случайно в соответствие вариационным распределением, заданным на структурах $q(\mathbf{\Gamma})$. Затем выбирается одна, либо несколько наилучших структур с учетом валидационной функции Q или внешних, возможно недиффиренецируемых, критериев качества. Итоговая модель получается путем оптимизации параметров модели при выбранной структуре $\mathbf{\Gamma}$. Заметим, что в ряде работ, одновременно порождается не только структура модели, но и итоговые параметры.

В работе [59] рассматривается порождение моделей, оптимизируемых без учителя. Модель представляется многослойным перцептроном вида:

$$\mathbf{f} = \mathbf{f}_{|\mathrm{V}|} \circ \cdots \circ \mathbf{f}_1(\mathbf{x}), \quad \mathbf{f}_i(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{w}^i \odot \mathbf{H}^i \mathbf{x}),$$

где $\mathbf{H}^i$ — бинарные матрицы, определяющие вклад каждого параметра из $\mathbf{w}^i$ в итоговую модель, знаком $\odot$ обозначается покомпонентное перемножение.

Порождение моделей производится с использованием композиции процессов индийский буфетов. Процесс индийского буфет заключается в итеративном построении матрицы $\mathbf{H}^i$ с ограниченным, но не заданным наперед количеством столбцов. Интепретируя количество столбцов матрицы как размер i-го слоя предлагается метод, позволяющий выбирать стохастически порождать модели с разилчной размерностью скрытых слоев.

В работе [60] предлагается метод выбора модели сверточной нейронной сети. Используется функция потерь, основанная на аппроксимации априорного распределения процесса индийского буфета для каждой базовой функции $\mathbf{g}_i$, являющейся i-м отображением объектов:

$$L = \sum_{\mathbf{x}} ||\mathbf{x} - \sum_{j=1}^{K} ||\mathbf{x} - \sum_{j=1}^{K} \mathbf{w}^j * \mathbf{g}_i(\mathbf{x})||_2^2 + \lambda^2 K,$$

где K — параметр, отвечающий за количество сверток, λ — параметр алгоритма, знаком $*$ обозначается операция свертки (TODO: проверить).

В работе [61] предлагается ввести априорное распределение Бернулли на структурные параметры $\boldsymbol{\gamma}^i$.

В [62] рассматривается задача выбора архитектуры с помощью большого

количества параллельных запусков обучения моделей. Предлагаются критерии ранней остановки процедуры оптимизации обучения моделей.

Последовательный выбор структуры модели. В работе [63] приводятся теоретические оценки построения нейросетей с использованием жадных стратегий, при которых построение модели производится итеративно последовательным увеличением числа нейронов в сети. В работе [64] предлагается жадная стратегия выбора модели нейросети с использованием релевантных априорных распределений, т.е. параметрических распределений, оптимизация параметров которых позволяет удалить часть параметров из модели. Данный метод был к задаче построения модели метода релевантных векторов [65].

В работах[66, 67] рассматривается послойное построение модели с отдельным критерием оптимизации для каждого слоя. В работах [68, 69, 70] предлагается декомпозиция модели на порождающую и разделяющую, оптимизируемых последовательно.

В работах [71, 8] предлагается наращивание моделей, основанное на бустинге. Рассматривается задача построения нейросетевых моделей специального типа:

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}_{|V|} \circ \mathbf{f}_{|V|-1} \circ \dots \mathbf{f}_0(\mathbf{x}), \quad \mathbf{f}_{i+1}(\mathbf{x}) = \boldsymbol{\sigma}\left(\mathbf{f}_i(\mathbf{x})\right) + \mathbf{f}_i(\mathbf{x}),$$

приводится параметризация модели, позволяющая рассматривать декомпозировать модель на слабые классификаторы. В [8] рассматривается задача выбора полносвязной нейронной сети для задачи бинарной классификации, $Z = 2$. На каждом шаге построения выбирается одно из двух расширений модели, каждое из которых рассматирвается как слабый классификатор: сделать модель шире или сделать модель глубже. Пример работы AdaNet представлен на Рис. 1.4. Построение модели заканчивается при условии снижении радемахеровской сложности:

$$\mathfrak{R} = \frac{1}{m}\mathsf{E}_{b_1,\dots,b_{|V|}}\sup_{\mathbf{w}}\sum_{i=1}^{m} b_i \arg\max_{c=\{0,1\}} f^c(\mathbf{x}_i, \mathbf{w}), \tag{1.13}$$

где b_i — реализация случайной дискретной величины, равновероятно принимающей значений -1 и 1, f^c — c-я компонента модели $\mathbf{f}$.

В работе [72] рассматривается задача порождения сверточных нейронных сетей. Предлагается проводить последовательный выбор структуры модели по восходящему числу параметров: начиная от сетей с одной подмоделью и итеративно увеличивая количество подмоделей. В силу высокой вычислительной сложности данного подхода, вместо последовательного порождения моделей, предлагается провести оптимизацию рекуррентной нейронной сети, которая предсказывает качество модели по заданным подмоделям, и на основе данного предсказания выбрать наилучшую модель.

В работе [73] предлагается метод анализа структуры сети на основе линейных классификаторов, построенных на промежуточных слоях нейросети. Схожий метод был предложен в [74], где классификаторы на промежуточных уровнях используются для уменьшения вычислений при выполнении вывода и

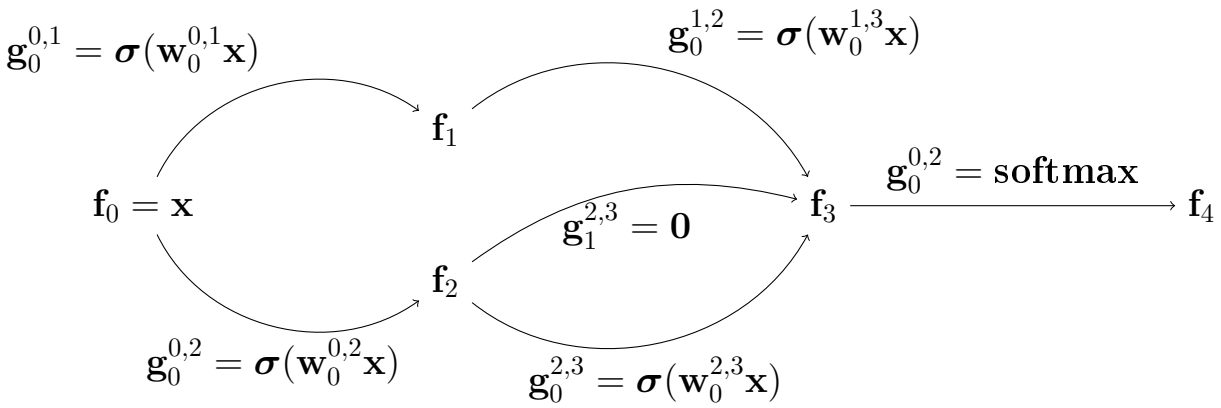


Рис. 1.4. Пример итерации алгоритма AdaNet [8]. Рассматривается две альтернативные модели: модель с углублением сети (соответствует занулению функции $\mathbf{f}_2$ с использованием базовой функции $\mathbf{g}_1^{2,3}$) и модель с расширением сети (соответствует базовой функции $\mathbf{g}_0^{2,3}$).

В качестве функции агрегации для подмодели $\mathbf{f}_3$ выступает конкатенация: $\mathbf{agg}_3 = \mathbf{concat}$.

предсказаний. Промежуточные классификаторы работают как решающий список.

В работе [75] предлагается инкрементальный метод оптимизации нейросети. На первом этапе модель декомпозируется на несколько подмоделей, при которой модель последовательностью слоев $\mathbf{f}_1, \ldots, \mathbf{f}_{|V|}$. Проводится последовательная оптимизация моделей вида:

1) $\mathbf{f} = \mathbf{f}_{|V|}(\mathbf{x})$;
2) $\mathbf{f} = \mathbf{f}_{|V-1|} \circ \mathbf{f}_{|V|}(\mathbf{x})$;
3) ...
4) $\mathbf{f} = \mathbf{f}_1 \circ \cdots \circ \mathbf{f}_{|V|}(\mathbf{x})$.

Оптимизация структуры модели на основе обучения с подкреплением. В [76] предлагается итеративная схема выбора архитектуры сверточной нейросети с использованием обучения с подкреплением. Распределение структур и параметров $q(\mathbf{w}, \mathbf{\Gamma})$ задается рекуррентной нейронной сетью, которая определяет значение параметров модели и наличие ребер с ненулевыми операциями между вершинами графов модели. Параметры рекуррентной нейронной сети оптимизируются на основе значения функции Q, получаемого на каждой итерации алгоритма.

В работе [77] предлагается алгоритм построения регрессионной модели для оценки финального качества модели и ранней остановки оптимизации моделей. Он позволяет существенно ускорить поиск моделей, представленный в [76]. В [78] рассматривается задача переноса архитектуры нейросети, чья структуры была выбрана по выборке, меньшей мощности. Как и в [76] предлагается метод параметризации сверточной нейронной сети в виде графа. Предложенная параметризация позволяет задать более мощное параметрическое семейство

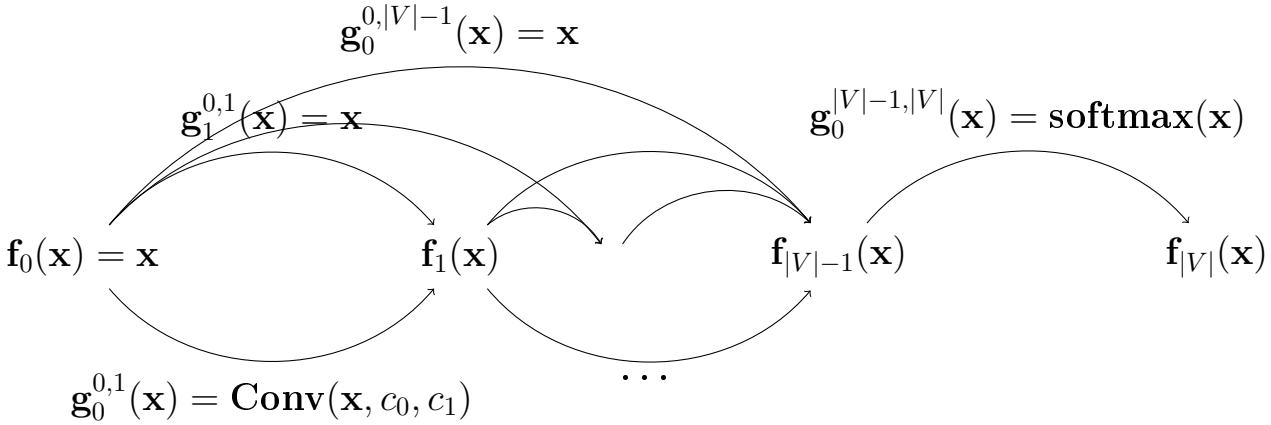


Рис. 1.5. Пример параметрического семейства моделей глубокого обучения, описываемый в [76]. Каждая подмодель $\mathbf{f}_j$ является линейной комбинацией базовых функций: свертки и результата работы предыдущих подмоделей (англ. skip-connection).

моделей, чем в [76]. Модель представляется в виде последовательности суперпозиций подмоделей, называемых клетками (англ. normal cell и reduction cell). Каждая из этих клеток содержит следующее множество нелинейных операций $\mathbf{g}$, состоящее из тождественной операции $\mathbf{g}(\mathbf{x}) = \mathbf{x}$, а также множество сверток с фиксированным количеством каналов и размером фильтров и функций субдискретизации или пулинга. Алгоритм выбора структуры модели рекуррентной сетью выглядит следующим образом на шаге j:

1) выбрать вершину v' из вершин v_{j-1}, v_{j-2} из данной клетки или вершину из предыдущих клеток;
2) выбрать вершину v'' из вершин v_{j-1}, v_{j-2} из данной клетки или вершину из предыдущих клеток;
3) выбрать базовую функцию $\mathbf{g}'$ для применения к вершине v';
4) выбрать базовую функцию $\mathbf{g}''$ для применения к вершине v'';
5) выбрать функцию агрегации результатов применения операций $\mathbf{g}', \mathbf{g}''$: сумму или конкатенацию.

В отличие от предыдущих работ, в работе [79] предлагается подход к инкрементальному обучению нейросети, основанном на модификации модели, полученной на предыдущем шаге. Рассматривается две операции над нейросетью: расширение и углубление сети.

В работах [80, 81, 82] рассматриваются методы деформации нейросетей. В работе [82] предлагается метод оптимального разделения нейросети на несколько независимых сетей для уменьшения количества связей и, как следствие, уменьшения сложности оптимизации модели. В работе [80] предлагается метод сохранения результатов оптимизации нейросети при построении новой более глубокой или широкой нейросети. В работе [81] рассматривается задача расширения сверточной нейросети, нейросеть рассматривается как граф.

В работе [1] используется представление модели из [78]. Вместо обучения с подкреплением испольщуются градиентная оптимизация структуры и параметров, выполненная в единой процедуре.

1.5. Метаоптимизация моделей глубокого обучения

Задача выбора структуры модели тесно связана с раздел машинного обучения под названием *метаобучение* или *метаоптимизация*. Под метаобучением понимаются алгоритмы машинного обучения [83], которые:

1) оценивают и сравнивают методы оптимизации моделей;
2) оценивают возможные декомпозиции процесса оптимизации моделей;
3) на основе полученных оценок предлагают оптимальные стратегии оптимизации моделей и отвергают неоптимальные.

В работе [84] предлагается подход к адаптивному изменению параметров сети. В качестве оператора оптимизации параметров рассматривается величина:

$$T(\boldsymbol{\theta}|L, \mathbf{y}, \mathbf{X}, \mathbf{h}, \boldsymbol{\beta}) = \boldsymbol{\theta} + \mathbf{f}_{\text{optim}}(\mathbf{f}_{\text{mod}}(\boldsymbol{\theta})),$$

где $\mathbf{f}_{\text{mod}}$ — функция, определяющая номер параметра из $\boldsymbol{\theta}$, подлежащего оптимизации, а $\mathbf{f}_{\text{optim}}$ — величина изменения параметра. В [84] также предлагается подмодель $\mathbf{f}_{\text{ana}}$, определяющая номер параметра, подлежащего дальнейшему анализу. Подход, описанный в данной работе, предполагает оптимизацию и анализ не только самой модели $\mathbf{f}$, но и дополнительных моделей $\mathbf{f}_{\text{mod}}, \mathbf{f}_{\text{ana}}, \mathbf{f}_{\text{optim}}$.

В работе [85] рассматривается оптимизация метапараметров (шага градиентного спуска β_{lr} и начального распределения параметров $\boldsymbol{\theta}^0$). Рассматривается задача оптимизации параметров модели в случае, когда количество примеров невелико. Для этого проводится оптимизация оператора оптимизации, который выглядит следующим образом:

$$T(\boldsymbol{\theta}|L, \mathbf{y}, \mathbf{X}, \mathbf{h}, \boldsymbol{\beta}) = \boldsymbol{\theta}^0 - \boldsymbol{\beta}\nabla L(\boldsymbol{\theta}^0|\mathbf{h}, \mathbf{X}, \mathbf{y},),$$

где векторы $\boldsymbol{\theta}^0$ и $\boldsymbol{\beta}$ являются параметрами оператора T. Задача оптимизации параметров оператора T рассматривается как задача многзадачного обучения (англ. multitask learning), когда оператор оптимизируется с учетом нескольких различных выборок и различных функций L, определенных отдельно для каждой выборки.

В работе [86] рассматирвается задача восстановления параметров модели по параметрам другой модели, чьи параметры были получены оптимизацией функции потерь на выборке меньшей мощности. Задачу можно рассматривать как задачу нахождения параметров некоторого оператора оптимизации $T : \boldsymbol{\theta}^0 \rightarrow \boldsymbol{\theta}$, где $\boldsymbol{\theta}^0$ — параметры модели, оптимизированной на небольшой выборке. Предлагается функция оптимизации:

$$T = \arg\min ||\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^0||_2^2 + \beta_\lambda L(\boldsymbol{\theta}|\mathbf{h}, \hat{\mathbf{X}}, \hat{\mathbf{y}}),$$

где $\boldsymbol{\theta}$ — параметры модели, обученной по полной выборке $\mathfrak{D}$, $\hat{\mathfrak{D}}$ — выборка меньшей мощности, β_λ — настраиваемый метапараметр.

В работе [87] рассматривается оптимизация метапараметров оператора оптимизации с помощью модели долгой краткосрочной памяти LSTM, которая выступает альтернативе аналитических алгоритмов, таких как Adam [88] или AdaGrad [89]. LSTM имеет небольшое число параметров, т.к. для каждого метапараметра используется свой экземпляр модели LSTM с одинаковыми параметрами для каждого экземпляра. Оптимизируемый функционал является суммой значений функции потерь L на нескольких шагах оптимизации:

$$Q = \sum_{t=1}^{\eta} L(\boldsymbol{\theta}^t),$$

где η — число шагов оптимизации, $\boldsymbol{\theta}^t$ — оптимизируемые параметры модели на шаге оптимизации t.

1.6. Выбор структур моделей специального вида

В данном разделе представлены работы по поиску оптимальных моделей со структурами специального вида.

В работе [90] рассматривается оптимизация моделей нейросетей с бинарной функцией активацией. Задача оптимизации сводится к задаче mixed integer программирования, которая решается методами выпуклого анализа. В работе [91] 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 [92, 93] рассматривается выбор архитектуры сети с использованием *суперсетей*: связанных между собой подмоделей, образующих граф, каждый путь из нулевой вершины в последнюю которого определяет модель глубокого обучения. Пример графа, описывающего суперсеть представлен на Рис. 1.6. В работе [93] рассматриваются стохастические суперсети, позволяющие выбрать структуру нейросети за органиченное время оптимизации. Схожий подход был предложен в работе [92], где предлагается использовать эволюционные алгоритмы для запоминания оптимальных подмоделей и переноса этих моделей в другие задачи.

Порождающие модели. Порождающими моделями называются модели, приближающие совместное распределение объектов и соответствующих им меток $p(\mathbf{X}, \mathbf{y})$. Частным случаем порождающих моделей являются модели, приближающие только распределение векторов объектов $\mathbf{X}$. Подобный случай будем считать частным случаем классификации при пустом множестве меток классов ($Z = 0$).

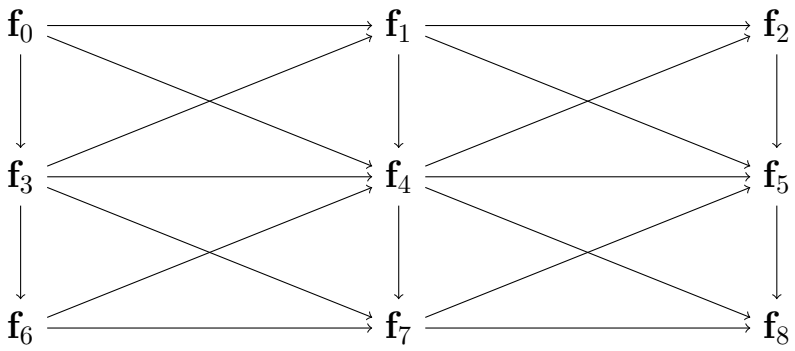


Рис. 1.6. Пример суперсети. Каждый путь из подмодели $\mathbf{f}_0$ в конечную модель $\mathbf{f}_8$ задает модель глубокого обучения.

В качестве порождающих моделей в сетях глубокого обучения выступают ограниченные машины Больцмана [94] и автокодировщики [95]. В работе [96] рассматриваются алгоритмы регуляризации автокодировщиков, позволяющих формально рассматривать данные модели как порождающие модели с использованием байесового вывода. В работе [97] рассматриваются регуляризованные автокодировщики и свойства оценок их правдоподобия. В работе [98] предлагается обобщение автокодировщика с использованием вариационного байесовского вывода [7]. В работе [99] рассматриваются модификации вариационного автокодировщика и ступенчатых сетей [100] для случая построения многослойных порождающих моделей.

В ряде работ [101, 102, 103, 104, 105] рассматривается подход к построению порождающих моделей глубокого обучения, при котором каждая подмодель $\mathbf{f}_i$ приближает распределение некоторой случайной величины $\mathbf{z}_i$, которая влияет на итоговое распределение $p(\mathbf{X}, \mathbf{y}) = \int_{\mathbf{z}_1,\ldots,\mathbf{z}_{|V|}} p(\mathbf{X}, \mathbf{y}|\mathbf{z}_1, \ldots, \mathbf{z}_{|V|}) p(\mathbf{z}_1, \ldots, \mathbf{z}_{|V|}) d\mathbf{z}_1 \ldots d\mathbf{z}_{|V|}$. Подобный подход позволяет использовать вероятностную интерпретацию для каждой отдельной подмодели.

В работе [101] рассматривается обобщение вариационного автокодировщика на случай более общих графических моделей. Предлагается проводить оптимизацию сложных графических моделей в единой процедуре. Для вывода предлагается использовать нейронные сети. Другая модификация вариационного автокодировщика представлена в работе [102], авторы рассматривают использование процесса сломанной трости в вариационном автокодировщике, тем самым получая модель со стохастической размерностью скрытой переменной. В [103] рассматривается смесь автокодировщиков, где смесь моделируется процессом Дирихле.

В работе [104] предлагается подход к оптимизации неизвестного распределения с помощью вариационного вывода. Предлагается решать задачу оптимизации итеративно, добавляя в модель новые компоненты вариационного распределения, проводится аналогия с бустингом.

В работе [105] рассматривается задача построения порождающих моделей с дискретными значениями скрытых переменных $\mathbf{z}_i$, предлагается критерий для послойного обучения порождающих моделей:

$$Q = \sum_{\mathbf{x}} \log \sum_{i} p(\mathbf{x}|\mathbf{z}_i) q(\mathbf{z}) \to \max,$$

где q — аппроксимирующее распределение для случайной величины $\mathbf{z}$.

В работе [106] 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. Схема порождения выборки $\mathbf{X}$ представлена на Рис. 1.7.

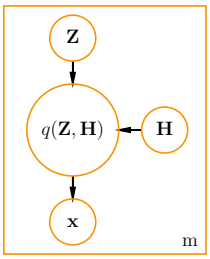


Рис. 1.7. Схема порождения вектора объектов $\mathbf{X}$, представленная в [106].

В данной работе предлагается метод последовательного порождения моделей глубокого обучения, основывающийся на применении вариационного вывода. Вариационный ввывод позволяет получить оценки правдоподобия модели с небольшими вычислительными затратами, а также проследить потенциальное начало переобучения модели без использования контрольной выборки. Для регуляризации структуры модели предлагается ввести априорное распределение на структуре, позволяющее проводить оптимизацию модели и ее структуры в различных режимах. В качестве метода оптимизации гиперпараметров выступают градиентные методы, что позволяет эффективно производить оптимизацию большого числа гиперпараметров, сопоставимого с числом параметров модели.

Глава 2
Анализ прикладных задач порождения и выбора моделей глубокого обучения

2.1. Выбор модели классификации временных рядов

В данном разделе рассматривается задача построения сети глубокого обучения для классификации временных рядов, где под временным рядом понимается упорядоченный по времени набор изменения некоторой случайной величины.

Для решения этой задачи используются методы глубокого обучения. Решению прикладных задач методами глубокого обучения посвящено значительное число современных работ. Работы [107, 108, 109] посвящены классификации временных рядов с использованием методов глубокого обучения. В работе [108] используются рекуррентные нейронные сети. В работе [109] для классификации временных рядов рассматриваются различные комбинации ограниченной машины Больцмана, автокодировщика и двуслойной нейронной сети. Исследуется суперпозиция, состоящая из ограниченной машины Больцмана, автокодировщика и двуслойной нейронной сети [95]. В работах [110, **?**] рассматривается проблема неустойчивости сети. В работе [110] исследуется поведение модели глубокого обучения как липшицевой функции. Работа [111] посвящена рекуррентной модификации модели ограниченной машины Больцмана [**?**] для классификации временных рядов. Схожие идеи предлагаются в работе [112], посвященной построению рекурсивного автокодировщика.

В данном разделе решается прикладная задача классификации временных рядов. В качестве данных для вычислительного эксперимента используются данные с акселерометров мобильных телефонов [113]. Для решения задачи оптимизации используется алгоритм обратного распространения ошибок с послойным предобучением сети и дальнейшей настройкой параметров всех слоев [114].

Постановка задачи. Рассматривается задача классификации. Моделью классификации $\mathbf{f}$ выступает суперпозиция подмоделей, аналогичная (**??**):

$$\mathbf{f}(\mathbf{w}, \mathbf{x}) = \mathbf{f}_1(\mathbf{f}_2(\dots \mathbf{f}_{|V|}(\mathbf{x}))) : \mathbb{R}^n \to [0,1]^Z, \tag{2.1}$$

где $\mathbf{f}_v, v \in \{1, \dots, |V|\}$, — модели, параметрическое семейство вектор-функции; $\mathbf{w}$ — вектор параметров моделей; r-ю компоненту вектора $\mathbf{f}(\mathbf{x}, \mathbf{w})$ будем интерпретировать как вероятность отнесения объекта $\mathbf{x}_i$ к классу с меткой r (**??**).

Требуется минимизировать функцию ошибки L на обучающей выборке $\mathfrak{D}$, где L — сумма отрицательных логарифмов правдоподобия по всем объектам выборки

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{w}|\mathfrak{D}),$$

где

$$L(\mathbf{w}|\mathfrak{D}) = -\sum_{(\mathbf{x},y)\in\mathfrak{D}} \sum_{r=1}^{Z} [y_i = r] \log p(y = r|\mathbf{x}, \mathbf{w}).$$

Структура сети глубокого обучения. Предлагается использовать в качестве алгоритма решения задачи суперпозицию, состоящую из трех основных компонент: ограниченной машины Больцмана, автокодировщика и двуслойной нейросети с softmax-классификатором.

Ограниченная машина Больцмана. Ограниченная машина Больцмана представляет собой двудольный граф, где первая доля соответствует переменной $\mathbf{x}$, а вторая доля — бинарному вектору $\mathbf{h}$ длины n'. Рассмотрим случай,

когда вектор $\mathbf{x}$ принимает бинарные значения. Определим энергию пары входного слоя $\mathbf{x}$ и скрытого слоя $\mathbf{h}$ следующим образом:

$$E(\mathbf{x}, \mathbf{h}) = -\mathbf{x}^{\mathrm{T}} \cdot \mathbf{b}_{\mathrm{vis}} - \mathbf{h}^{\mathrm{T}} \cdot \mathbf{b}_{\mathrm{hid}} - \mathbf{h}^{\mathrm{T}} \mathbf{w}_{\mathrm{RBM}} \mathbf{x},$$

где $\mathbf{b}_{\mathrm{vis}}, \mathbf{b}_{\mathrm{hid}}, \mathbf{w}_{\mathrm{RBM}}$ — параметры модели.

Пусть совместное распределение пары векторов $\mathbf{x}, \mathbf{h}$ задано следующим образом:

$$p(\mathbf{x}, \mathbf{h}) = \frac{1}{J} \exp\big(-E(\mathbf{x}, \mathbf{h})\big),$$

где J — нормировочный коэффициент:

$$J = \sum_{\mathbf{x} \in \{0,1\}^n, \mathbf{h} \in \{0,1\}^{n'}} \exp\big(-E(\mathbf{x}, \mathbf{h})\big).$$

Функция вероятностей вектора $\mathbf{x}$ есть сумма вероятностей по всем скрытым состояниям вектора $\mathbf{h}$:

$$p(\mathbf{x}) = \sum_{\mathbf{h} \in \{0,1\}^{n'}} p(\mathbf{x}, \mathbf{h}).$$

Определим элемент суперпозиции (2.1):

$$\mathbf{f}_{\mathrm{RBM}}(\mathbf{x}) = \mathsf{E}(\mathbf{h}|\mathbf{x}). \tag{2.2}$$

Настройка параметров модели (2.2) осуществляется решением задачи оптимизации

$$\mathbf{w}^*_{\mathrm{RBM}}, \hat{\mathbf{b}}_{\mathrm{vis}}, \hat{\mathbf{b}}_{\mathrm{hid}} = \underset{\mathbf{w}_{\mathrm{RBM}}, \mathbf{b}_{\mathrm{vis}}, \mathbf{b}_{\mathrm{hid}}}{\arg\max} \; p(\mathfrak{D}; \mathbf{W}, \mathbf{b}_{\mathrm{vis}}, \mathbf{b}_{\mathrm{hid}}) = \prod_{\mathbf{x} \in \mathfrak{D}} \sum_{\mathbf{h} \in \{0,1\}^{n'}} \frac{1}{J} \exp\big(-E(\mathbf{x}, \mathbf{h})\big). \tag{2.3}$$

В данной работе используется модифицированная версия ограниченной машины Больцмана, позволяющая работать с небинарными входными данными [115]. В этой модификации энергия E пары входного слоя $\mathbf{x}$ и скрытого слоя $\mathbf{h}$ выглядит следующим образом:

$$E(\mathbf{x}, \mathbf{h}) = \frac{(\mathbf{x} - \mathbf{b}_{\mathrm{vis}})^2}{2\boldsymbol{\sigma}^2} - \mathbf{h}^{\mathrm{T}} \cdot \mathbf{b}_{\mathrm{hid}} - \frac{\mathbf{h}}{\boldsymbol{\sigma}}^{\mathrm{T}} \mathbf{w}_{\mathrm{RBM}} \mathbf{x},$$

где $\boldsymbol{\sigma}$ — оценка дисперсии объектов выборки $\mathfrak{D}$, деление производится покомпонентно.

Для решения задачи оптимизации (2.3) используется алгоритм, описанный в [94].

Автокодировщик. Автокодировщик предназначен для снижения размерности исходного пространства признаков. Автокодировщик представляет собой суперпозицию кодирующего и декодирующего блока:

$$\mathbf{f}'_{\mathrm{AE}} = \mathbf{f}_{\mathrm{enc}}(\mathbf{f}_{\mathrm{dec}}(\mathbf{x})),$$

где

$$\mathbf{f}_{\text{enc}}(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{w_e}\mathbf{x} + \mathbf{b_e}) \text{ — кодирующий блок,}$$
$$\mathbf{f}_{\text{dec}}(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{w_d}\mathbf{g}(\mathbf{x}) + \mathbf{b_d}) \text{ — декодирующий блок,}$$
$$\boldsymbol{\sigma}(x) = (1 + \mathbf{exp}(-\mathbf{x}))^{-1} \text{ — сигмоидная функция,}$$

$\mathbf{w_e}, \mathbf{w_d}, \mathbf{b_e}, \mathbf{b_d}$ — параметры модели.

Введем дополнительное ограничение на матрицы $\mathbf{w_e}, \mathbf{w_d}$:

$$\mathbf{w_e} = \mathbf{w_d}^{\text{T}}.$$

Оптимизацию параметров модели $\mathbf{w}_e$ будем проводить таким образом, чтобы по образу вектора $\mathbf{x}$, получаемому с помощью кодирующего блока, можно было получить вектор $\mathbf{f}_{\text{AE}}$, близкий к исходному входному $\mathbf{x}$, при помощи преобразования декодирующего блока:

$$\mathbf{w_e^*}, \mathbf{w_d^*}, \mathbf{b_e^*}, \mathbf{b_d^*} = \underset{\mathbf{w_e}, \mathbf{w_d}, \mathbf{b_e}, \mathbf{b_d}}{\arg\min} \frac{1}{|\mathfrak{D}|} \sum_{\mathbf{x} \in \mathfrak{D}} ||\mathbf{f}_{\text{AE}}(\mathbf{x}) - \mathbf{x}||_2^2. \tag{2.4}$$

Декодирующий блок $\mathbf{f}_{\text{dec}}$ требуется только для решения задачи оптимизации (2.4) и не используется в суперпозиции (2.1). Таким образом, элемент суперпозиции (2.1) определен как

$$\mathbf{f}_{\text{AE}} = \mathbf{f}_{\text{enc}}(\mathbf{x}).$$

Двуслойная нейросеть. Двуслойная сеть представляет собой логистическую вектор-функцию:

$$\mathbf{f}_{\text{hidden}}(\mathbf{x}) = \mathbf{w}_2^{\text{T}} \mathbf{tanh}(\mathbf{w}_1^{\text{T}} \mathbf{x}), \tag{2.5}$$
$$\mathbf{f}_{\text{SM}}(\mathbf{x}) = \frac{\mathbf{exp}\big(\mathbf{f}_{\text{hidden}}(\mathbf{x})\big)}{\sum_{j=1}^{Z} \exp\big(\mathbf{f}_{\text{hidden}}^{j}(\mathbf{x})\big)},$$

где r-я компонента вектора $\mathbf{f}_{\text{SM}}(\mathbf{x})$ интерпретируется как вероятность принадлежности объекта $\mathbf{x}$ классу r. Итоговая функция классификации (2.1) ставит в соответствие объекту $\mathbf{x}$ метку класса y, где y — класс, к которому принадлежит $\mathbf{x}$ с наибольшей вероятностью:

$$f(\mathbf{w}, \mathbf{x})_r = \begin{cases} 1, \text{ если } r = \arg\max_{r'} f_{\text{SM}}(\mathbf{f}_{\text{AE}}(\mathbf{f}_{\text{RBM}}(\mathbf{x}))_{r'}, \\ 0 \text{ иначе.} \end{cases}$$

Здесь $\mathbf{f}_{\text{AE}}, \mathbf{f}_{\text{RBM}}$ — автокодировщик (2.4) и ограниченная машина Больцмана (2.3) соответственно, $f_{\text{SM}}(\mathbf{x})_r$ — r-я компонента вектора $\mathbf{f}_{\text{SM}}$, $f(\mathbf{w}, \mathbf{x})_r$ — r-я компонента вектор-функции $\mathbf{f}$.

Итоговая задача оптимизации выглядит следующим образом:

$$\boldsymbol{\theta}^* = \arg\min \sum_{\mathbf{x}, y \in \mathfrak{D}} \sum_{r=1}^{Z} [y = r] \log(f_{\text{SM}}^r(\mathbf{f}_{\text{AE}}(\mathbf{f}_{\text{RBM}}(\mathbf{x})))),$$

где $\boldsymbol{\theta}^* = [\mathbf{w}^*_{\text{RBM}}, \mathbf{b}^*_{\text{vis}}, \mathbf{b}^*_{\text{hid}}, \mathbf{w}^*_{\mathbf{e}}, \mathbf{b}^*_{\mathbf{e}}, \mathbf{w}^{*\mathsf{T}}_2, \mathbf{w}^{*\mathsf{T}}_1]$ — параметры ограниченной машины Больцмана (2.3), автокодировщика (2.4) и двуслойной сети (2.5).

Результаты вычислительного эксперимента. В качестве данных для проведения вычислительного эксперимента использовались данные WISDM [113], представляющие собой набор записей акселерометра мобильного телефона. Каждой записи соответствуют три координаты по осям акселерометра. Набор данных содержит записи движений для 6 классов переменной длины. При проведении вычислительного эксперимента из каждой записи использовались первые 200 сегментов. Т. к. выборка не сбалансирована, в нее добавлялись повторы записей классов, содержащих количество записей, меньшее чем у большего класса.

Основные эксперименты — исследование зависимости ошибки классификации от числа параметров и размера выборки — были проведены как с использованием инструментария на базе библиотеки Theano, так и с использованием инструментария на языке Matlab. Для оценки качества классификации была проведена процедура скользящего контроля [23] при соотношении числа объектов обучающей и контрольной выборки 3:1. Число нейронов на каждом слое задавалось из соотношения 10:6:3. При проведении процедуры скользящего контроля для каждого отсчета количества нейронов было произведено пять запусков. В эксперименте с использованием инструментария на базе Theano при обучении двуслойной нейронной сети проводился мультистарт [116], т. е. одновременный запуск обучения сети с 8 разными стартовыми значениями параметров для предотвращения возможного застревания алгоритма обучения в локальном минимуме. При оценке качества классификации выбиралась модель с наилучшими результатами. График зависимости ошибки классификации от числа используемых нейронов изображен на рис. 2.1.

Для оценки зависимости качества классификации от размера обучающей выборки была проведена кроссвалидация с фиксированным количеством объектов в обучающей выборке (25% исходной выборки) и переменным размером обучающей выборки. Число нейронов было установлено как 364:224:112. При проведении процедуры скользящего контроля для каждого отсчета было произведено пять запусков. График зависимости ошибки классификации от размера обучающей выборки представлен на рис. 2.2.

Для исследования скорости оптимизации нейросети в зависимости от конфигурации Theano был сделан следующий эксперимент: проводилось обучение двуслойной нейросети на основе подсчитанных заранее параметров ограниченной машины Больцмана (2.3) и автокодировщика (2.4). Обучение проходило за 100 итераций. При обучении алгоритм запускался параллельно с r разными стартовыми позициями, $r \in \{1, \ldots, 4\}$. Число нейронов было установлено как 300:200:100. Запуск осуществлялся со следующими конфигурациями Theano:

- вычисление на центральном процессоре, задействовано одно ядро;
- вычисление на центральном процессоре, задействовано четыре ядра;
- вычисление на центральном процессоре, задействовано восемь ядер;

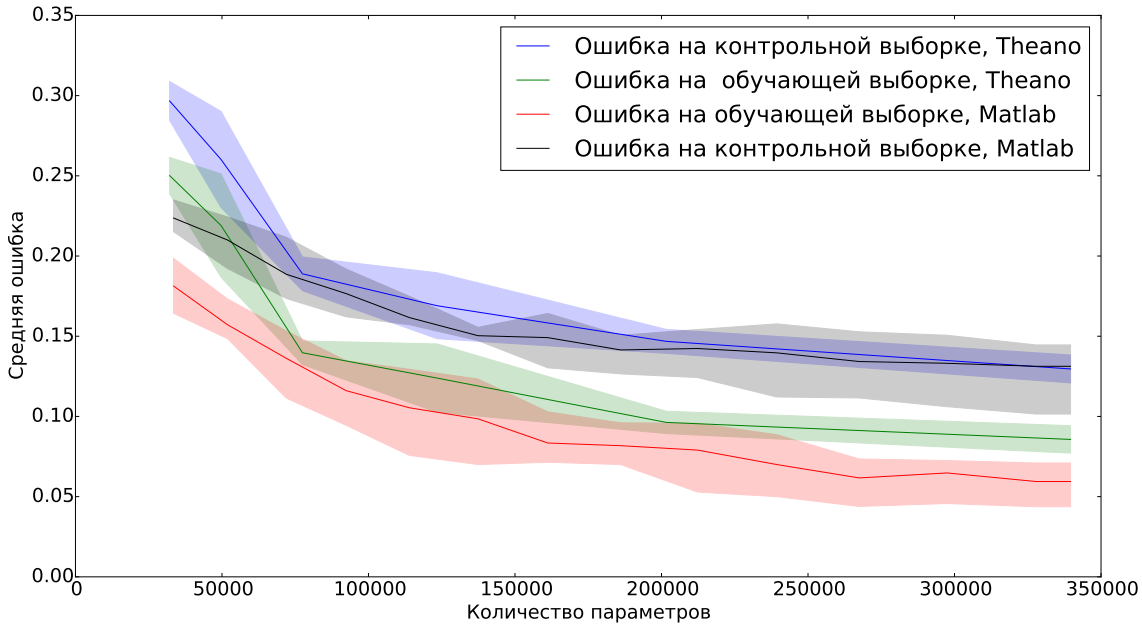


Рис. 2.1. Зависимость ошибки от числа нейронов

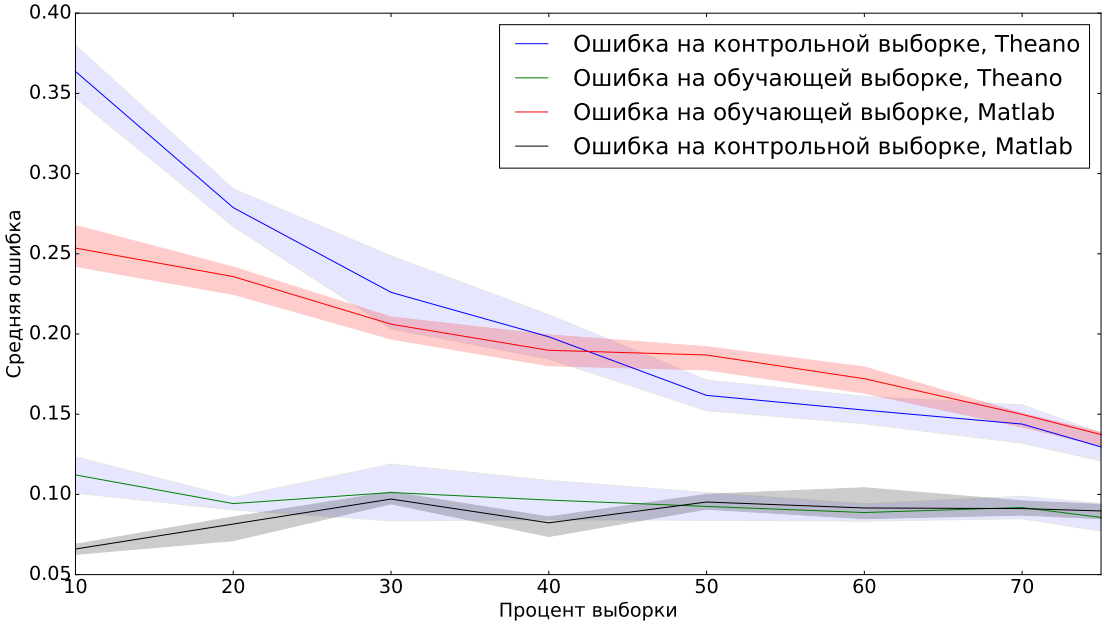


Рис. 2.2. Зависимость ошибки от размера обучающей выборки

- вычисление на графическом процессоре.

Результаты эксперимента приведены на рис. 2.3. Как видно из графика, вычисление с использованием CUDA показывает значительное ускорение по сравнению с вычислением на центральном процессоре.

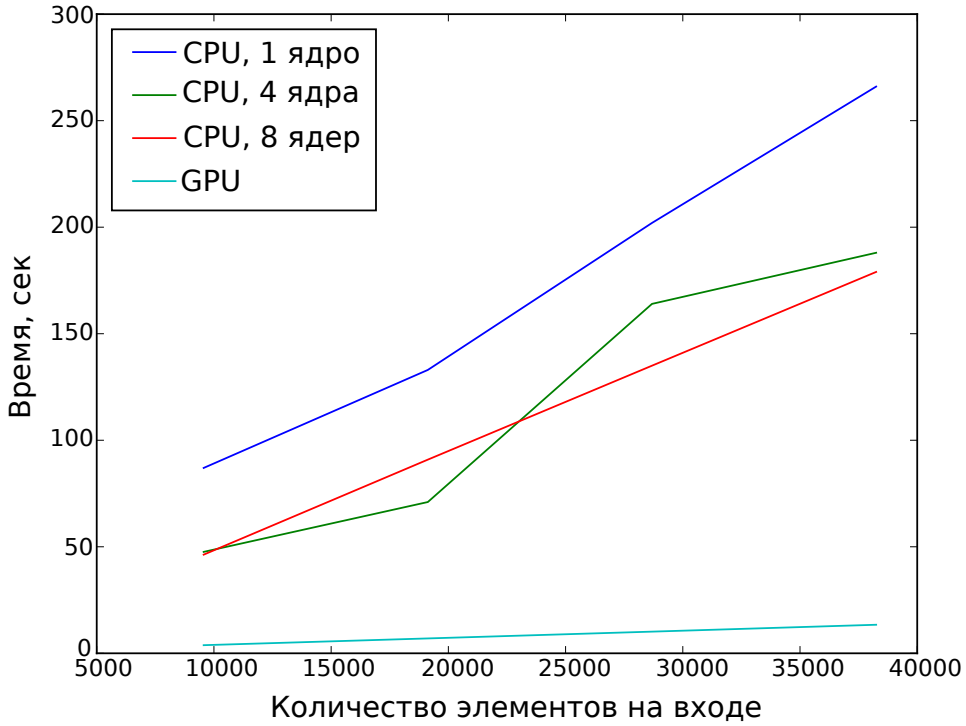


Рис. 2.3. Результаты эксперимента по исследованию скорости процесса обучения

2.2. Выбор модели обнаружения перефраза в тексте

В данном разделе решается задача выбора оптимальной нейросетевой модели из класса рекуррентных нейронных сетей. Рекуррентной нейросетью называется нейросеть со связью между нейронами одного слоя. В качестве критерия оптимальности используется нижняя оценка правдоподобия модели.

Для построения модели рекуррентной сети рассматривается модель из [?], решающая задачу определения сходства предложений. Модель принимает на вход векторизованные представления слов. Векторизация выполняется с помощью алгоритма GloVe [?], основанного на факторизации матрицы слов-контекстов и использовании весовой функции для уменьшения значимости редких слов. Альтернативой этому алгоритму выступает линейная модель Word2vec, комбинирующая в себе Continuous Bag-of-Words, skip-gram, negative sampling [?]. Несмотря на разные подходы к проблеме, GloVe и Word2vec оптимизируют схожие функционалы. Упрощенной линейной моделью Word2vec, предназначенной для классификации документов, является fastText — метод, работающий на символьных n-граммах [?].

Предлагается подход, основанный на получении вариационной нижней оценки правдоподобия модели. Предлагаемый подход сравнивается с методом удаления параметров Optimal Brain Damage, базирующимся на анализе функции ошибки (1.8).

Вычислительный эксперимент проводится на выборке размеченных пар

предложений SemEval 2015. Для каждой пары предложений из корпуса дана экспертная оценка их семантической близости. Требуется построить модель, оценивающую семантическую близость двух предложений. Проблема рассматривается как задача многоклассовой классификации, аналогично [?]. Критерием качества служит F1-мера, учитывающая как полноту, так и точность предсказаний. В качестве базовой модели рассматривается пара соединенных рекуррентных сетей с общим вектором параметров и softmax-классификатором на выходе.

Постаноквка задачи. Для построения выборки используем набор пар предложений SemEval 2015 [?].

Каждому слову сопоставим вектор размерности n. Обозначим через l число слов в самом длинном предложении. Предложения длины, меньше l, дополним нулевыми векторами. Построим выборку

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

где $\mathbf{x}_i = [\mathbf{x}_i^1, \mathbf{x}_i^2]$ — пары последовательностей векторов слов, соответствующих i-й паре предложений, $\mathbf{x}_i^1, \mathbf{x}_i^2 \in \mathbb{R}^{n \times l}$; $y_i \in Y = \{0, \ldots, 5\}$ — экспертная оценка семантической близости.

Требуется построить модель $f(\mathbf{w}) : \mathbb{R}^{n \times l} \times \mathbb{R}^{n \times l} \to Y$, сопоставляющую паре предложений $\mathbf{x}_i^1$ и $\mathbf{x}_i^2$ класс семантической близости, где $\mathbf{w} \in \mathbb{W} \subseteq \mathbb{R}^s$ — пространство параметров модели. Искомая модель выбирается из множества M рекуррентных нейронных сетей с функцией активации tanh. Модель

$$f(\mathbf{w}) : \mathbb{R}^{n \times l} \times \mathbb{R}^{n \times l} \to Y$$

принадлежит искомому множеству моделей M, если существуют такие матрицы перехода $\mathbf{W} \in \mathbb{R}^{n \times m}, \mathbf{U} \in \mathbb{R}^{n \times n}, \mathbf{V} \in \mathbb{R}^{(|Y| \times 2n)}$ и вектор смещения $\mathbf{b} \in \mathbb{R}^n$, что для j-х элементов $\mathbf{x}_{ij}^1, \mathbf{x}_{ij}^2 \in \mathbb{R}^m$ последовательностей $\mathbf{x}_i^1$ и $\mathbf{x}_i^2$ определены векторы скрытого слоя $\mathbf{h}_{ij}^1, \mathbf{h}_{ij}^2 \in \mathbb{R}^n$:

$$\mathbf{h}_{ij}^1 = \tanh(\mathbf{W} \cdot \mathbf{x}_{ij}^1 + \mathbf{U} \cdot \mathbf{h}_{i,j-1}^1 + \mathbf{b}), \tag{2.6}$$

$$\mathbf{h}_{ij}^2 = \tanh(\mathbf{W} \cdot \mathbf{x}_{ij}^2 + \mathbf{U} \cdot \mathbf{h}_{i,j-1}^2 + \mathbf{b}). \tag{2.7}$$

Для определения класса семантической близости используются последние значения скрытого слоя $\mathbf{h}_{il}^1$ и $\mathbf{h}_{il}^2$, сконкатенированные в один вектор. После l-й итерации пару предложений будем относить к классу с наибольшим значением, полученным после l-й итерации, $j = 1, \ldots, l$:

$$y = \arg\max_{k \in Y} \left(\mathbf{V} \begin{bmatrix} \mathbf{h}_{il}^1 \\ \mathbf{h}_{il}^2 \end{bmatrix} \right)_k, \tag{2.8}$$

где $(\cdot)_k$ — k-я компонента вектора.

В качестве оптимизируемой функции потерь L выступает вариационная оценка правдоподобия модели (**??**): TODO.

В качестве вариационного распределения выберем нормальное распределение:

$$q \sim \mathcal{N}(\boldsymbol{\mu}_q, \mathbf{A}_q^{-1}),$$

где $\boldsymbol{\mu}_q, \mathbf{A}_q^{-1}$ — вектор средних и матрица ковариации этого распределения. Априорное распределение $p(\mathbf{w}|\mathbf{f})$ вектора параметров $\mathbf{w}$ будем считать нормальным с параметрами $\boldsymbol{\mu}$ и $\mathbf{A}$:

$$p(\mathbf{w}|\mathbf{h}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{A}^{-1}),$$

где $\boldsymbol{\mu}$ — вектор средних, $\mathbf{A}^{-1}$ — матрица ковариаций.

Расстояние Кульбака–Лейблера между нормальными распределениями $\mathcal{N}(\boldsymbol{\mu}, \mathbf{A}_{\text{pr}}^{-1})$ и $\mathcal{N}(\mathbf{m}, \mathbf{A}_{\text{ps}}^{-1})$ вычисляется по формуле

$$TODO D_{\text{KL}}\big(q(\boldsymbol{\mu}_q, \mathbf{A}_q^{-1})||p(\boldsymbol{\mu}, \mathbf{A}^{-1})\big) = \frac{1}{2}\big(\log\frac{|\mathbf{A}_{\text{ps}}^{-1}|}{|\mathbf{A}_{\text{pr}}^{-1}|} - d + \text{tr}(\mathbf{A}_2\mathbf{A}_{\text{pr}}^{-1}) + (\boldsymbol{\mu} - \mathbf{m})^T\mathbf{A}_2(\boldsymbol{\mu} - \tag{2.9}$$

Рассмотрим частные случаи вида матриц ковариаций $\mathbf{A}_q^{-1}$ и $\mathbf{A}^{-1}$. Так как априори нет предпочтений при выборе параметров, то априорное распределение для всех параметров считаем одинаковым, т. е. вектор средних $\boldsymbol{\mu} = \mu$, матрица ковариаций скалярна: $\mathbf{A}^{-1} = \alpha\mathbf{I}$.

Будем уточнять априорное распределение после каждого шага оптимизации вариационных параметров. Алгоритм решения оптимизационной задачи заключается в выполнении градиентного шага при заданном априорном распределении, вычислении апостериорного распределения и аппроксимации нового априорного распределения полученным апостериорным.

Рассмотрим различные виды матрицы ковариаций $\mathbf{A}_q^{-1}$ вариационного распределения q.

1. Матрица ковариаций скалярна: $\mathbf{A}_q^{-1} = \alpha_q\mathbf{I}$. В этом случае TODO

$$D_{\text{KL}}\big(\mathcal{N}(\boldsymbol{\mu}, \mathbf{A}_{\text{pr}}^{-1})||\mathcal{N}(\mathbf{m}, \mathbf{A}_{\text{ps}}^{-1})\big) = \sum_{i=1}^{W}\big(\log\frac{\sigma}{\alpha} + \frac{(\mu - m_i)^2 + \alpha^2 + \sigma^2}{2\sigma^2}\big).$$

   По значениям параметров a_q и $\boldsymbol{\mu}$ апостериорного распределения вычислим параметры априорного. TODO Из условия $\frac{\partial}{\partial\mu}D_{\text{KL}} = \sum_{i=1}^{W}\frac{\mu - m_i}{\sigma^2} = 0$ получаем выражения для μ на следующей итерации $\hat{\mu} = \frac{1}{W}\sum_{i=1}^{W} m_i$. Аналогично $\frac{\partial}{\partial\sigma^2}D_{\text{KL}} = \sum_{i=1}^{W}\frac{1}{2\sigma^2} - \frac{(\mu - m_i)^2 + \alpha^2}{2\sigma^4} = 0 \;\Rightarrow\; \hat{\sigma}^2 = \frac{1}{W}\sum_{i=1}^{W}(\mu - m_i)^2 + \alpha^2$.

2. Матрица ковариаций диагональна: $\mathbf{A}_q^{-1} = \text{diag}(\mathbf{a}_q^2)$. TODO В этом случае

$$D_{\text{KL}}\big(\mathcal{N}(\boldsymbol{\mu}, \mathbf{A}_{\text{pr}}^{-1})||\mathcal{N}(\mathbf{m}, \mathbf{A}_{\text{ps}}^{-1})\big) = \sum_{i=1}^{W}(\log\frac{\sigma}{\sigma_i} + \frac{(\mu - m_i)^2 + \sigma_i^2 + \sigma^2}{2\sigma^2}).$$

Значения параметров априорного распределения для следующей итерации вычисляются следующим образом:

$$\text{из } \frac{\partial}{\partial \mu} D_{\mathrm{KL}} = \sum_{i=1}^{W} \frac{\mu - m_i}{\sigma^2} = 0 \text{ получаем } \hat{\mu} = \frac{1}{W} \sum_{i=1}^{W} m_i,$$

$$\text{из } \frac{\partial}{\partial \sigma^2} D_{\mathrm{KL}} = \sum_{i=1}^{W} \frac{1}{2\sigma^2} - \frac{(\mu - m_i)^2 + \sigma_i^2}{2\sigma^4} = 0 \text{ получаем } \hat{\sigma}^2 = \frac{1}{W} \sum_{i=1}^{W} (\mu - m_i)^2 + \sigma_i^2.$$

Оптимизация параметров сводится к следующему алгоритму:

Инициализировать $\mathbf{a}_q = \mathbf{1},\ \boldsymbol{\mu}_q = \mathbf{0}, \mu = 0,\ a^2 = 1.$

Повторять:

Сделать градиентный шаг по вариационным параметрам.

Обновить параметры априорного распределения.

Пока значение L не стабилизируется.

Удаление параметров из сети. Введем множество индексов активных параметров модели $\mathcal{A} = \{i | w_i \neq 0\}$. Для увеличения правдоподобия модели предлагается уменьшить ее сложность TODO, т. е. уменьшить число параметров $|\mathcal{A}|$. Для удаления выберем параметры, имеющие наибольшую плотность вариационной вероятности ρ в нуле. Если апостериорная матрица ковариаций скалярна, то

$$\rho_i = \exp\left(-\frac{\mu_{q,i}^2}{a_{q,i}^2}\right). \tag{2.10}$$

Чем больше ρ, тем меньше $|\frac{\mu_{q,i}}{a_{q,i}}|$, поэтому удаляются параметры со значением $|\frac{\mu_{q,i}}{a_{q,i}}| < \lambda$, где λ — пороговое значение. Варьируя пороговое значение λ, выбираем оптимальное число неудаленных параметров. Для диагонального вида матрицы ковариаций критерий удаления параметров записывается как $|\frac{\mu_{q,i}}{a_{q,i}}| < \lambda$.

Вычислительный эксперимент. Цель эксперимента — проверка работоспособности предложенного алгоритма и сравнение результатов с ранее полученными. В качестве данных использовалась выборка SemEval 2015, состоящая из 8331 пары схожих и несхожих предложений. Слова преобразовывались в векторы размерности 50 при помощи алгоритма GloVe [**?**].

Для базовых алгоритмов тренировочная, валидационная и тестовая выборки составили 70%, 15% и 15% соответственно. Для рекуррентной нейронной сети, полученной вариационным методом, валидационная выборка отсутствовала, а тренировочная и тестовая выборки составили 85% и 15% соответственно. Критерием качества была выбрана F1-мера. В качестве базовых алгоритмов использовались линейная регрессия, метод ближайших соседей, решающее дерево и модификация метода опорных векторов SVC. Базовые алгоритмы взяты из библиотеки sklearn [54]. Дополнительно были построены рекуррентная

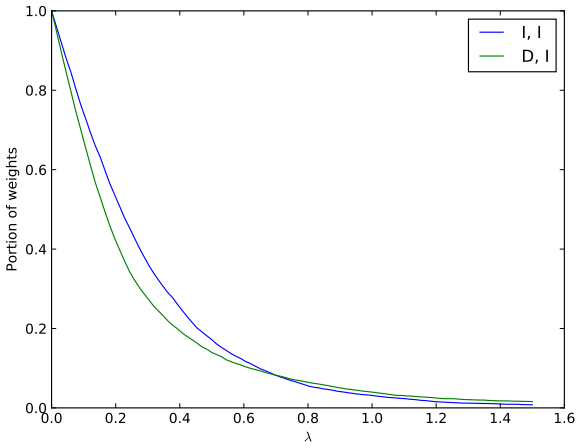


Рис. 2.4. Доля неудаленных параметров сети в зависимости от порогового значения λ для скалярного (I) и диагонального (D) вида апостериорной матрицы ковариаций

нейросеть с одним скрытым слоем [**?**] и нейросеть с одним скрытым слоем и вариационной оптимизацией параметров [**?**, **?**].

На рис. 2.5а и 2.5б представлена зависимость оценки правдоподобия L (**??**) от параметра λ. Для обоих случаев существует оптимальное значение λ, минимизирующее L; модели с таким параметром будут оптимальными. На рис. 2.5в, 2.5г, 2.5д и 2.5е отображены зависимости качества модели от λ и доли выброшенных параметров. Видно, что даже при удалении большинства параметров из сети качество предсказаний меняется несущественно, что говорит о слишком большом числе параметров исходной модели.

Из рис. 2.4 видно, что при малых λ из сети с диагональной апостериорной матрицей ковариаций удаляется больше весов, а при больших λ — меньше, что говорит о лучшем отборе параметров такой моделью.

Таблица 2.1. Результаты вычислительного эксперимента

Classificator	F1-measure, валидация	F1-measure, test
Logistic Regression	0,286	0,286
SVC	0,290	0,290
DecisionTreeClassifier	0,316	0,316
KNeighborsClassifier	0,322	0,322
RNN	0,393	0,362
RNN+variational, I, I	—	0,311
RNN+variational, D, I	—	0,330

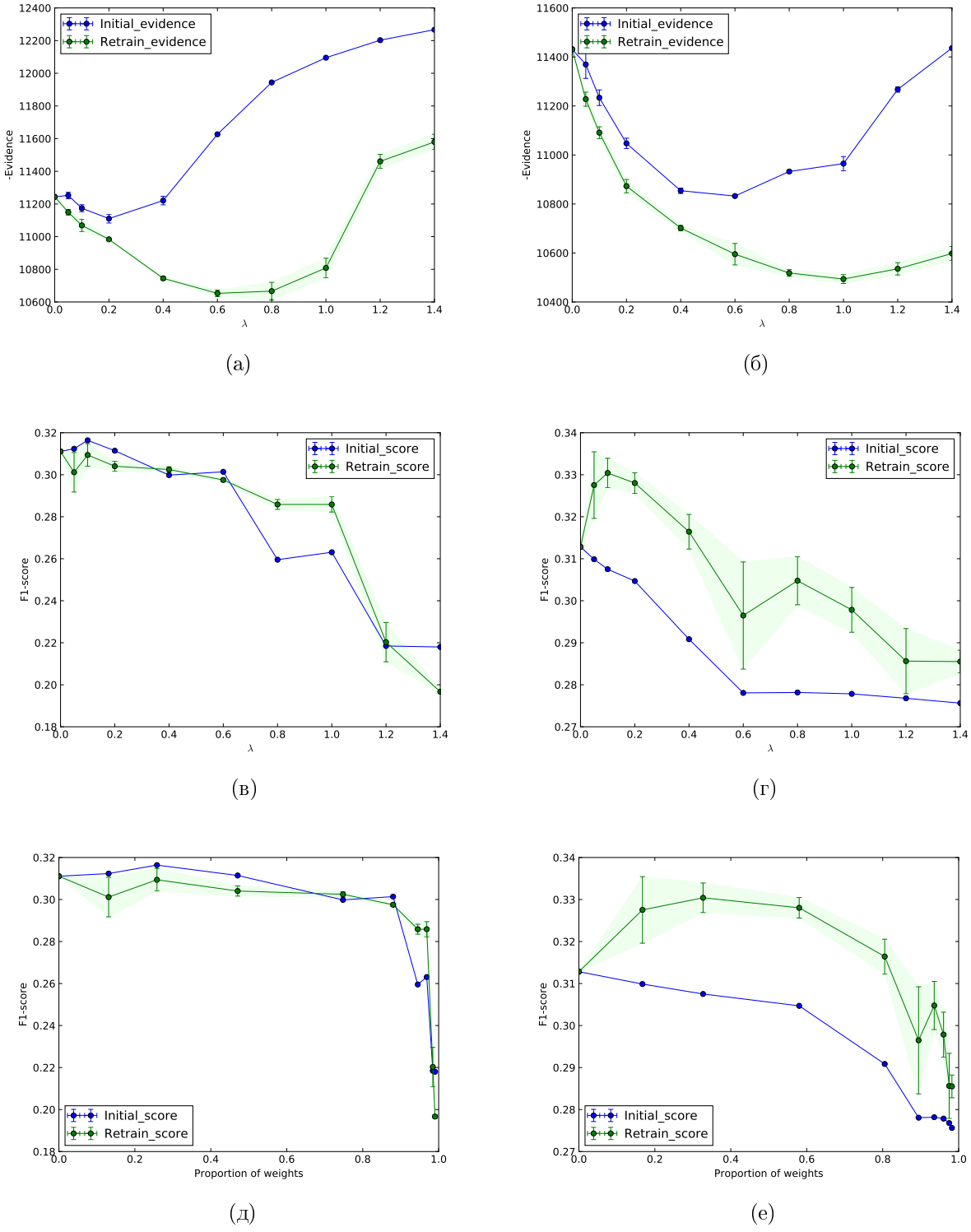


Рис. 2.5. Зависимость нижней оценки правдоподобия модели и F1-меры от λ для скалярной (а, б, в) и диагональной(г, д, е) матриц.

2.3. Определение релевантности параметров модели глубокого обучения

В данном разделе решается задача выбора субоптимальной структуры нейронной сети. Предлагается удалять наименее релевантные параметры модели. Под релевантностью [?] подразумевается то, насколько параметр влияет на функцию ошибки. Малая релевантность указывает на то, что удаление этого параметра не влечет значимого изменения функции ошибки. Метод предлагает построение исходной избыточной сложности нейросети с большим числом избыточных параметров. Для определения релевантности параметров предлагается оптимизацировать параметры и гиперпараметры в единой процедуре. Для удаления параметров предлагается использовать метод Белсли [?] .

Проверка и анализ метода проводится на выборке Boston Housing [?], Wine [?] и синтетических данных. Результат сравнивается с моделью, полученной при помощи базовых алгоритмов.

Постановка задачи

Задана выборка

$$\mathfrak{D} = \{\mathbf{x}_i, y_i\},\ i = 1, ..., N, \tag{2.1}$$

где $\mathbf{x}_i \in \mathbb{R}^m$, $y_i \in \{1, \ldots, Y\}$, Y — число классов. Рассмотрим модель

$$\mathbf{f}(\mathbf{x}, \mathbf{w}) : \mathbb{R}^m \times \mathbb{R}^n \to \{1, \ldots, Y\}$$

, где $\mathbf{w} \in \mathbb{R}^n$ — пространство параметров модели,

$$\mathbf{f}((\mathbf{x}, \mathbf{w}) = \mathrm{softmax}\big(f_1(f_2(...(f_l(\mathbf{x}, \mathbf{w})\big), \tag{2.2}$$

где $f_k(\mathbf{x}, \mathbf{w}) = \tanh(\mathbf{w}^\mathsf{T}\mathbf{x})$, l — число слоев нейронной сети, $k \in \{1 \ldots l\}$. Параметр w_j модели f называется активным, если $w_j \neq 0$. Множество индексов активных параметров обозначим $\mathcal{A} \subset \mathcal{J} = \{1, ..., n\}$. Задано пространство параметров модели:

$$\mathbb{W}_\mathcal{A} = \{\mathbf{w} \in \mathbb{R}^n \mid w_j \neq 0,\ j \in \mathcal{A}\}, \tag{2.3}$$

Для модели $\mathbf{f}$ с множеством индексов активных параметров $\mathcal{A}$ и соответствующего ей вектора параметров $\mathbf{w} \in \mathbb{W}_\mathcal{A}$ определим логарифмическую функцию правдоподобия выборки:

$$\mathcal{L}_\mathfrak{D}(\mathfrak{D}, \mathcal{A}, \mathbf{w}) = \log p(\mathfrak{D}|\mathcal{A}, \mathbf{w}), \tag{2.4}$$

где $p(\mathfrak{D}|\mathcal{A}, \mathbf{w})$ — апостериорная вероятность выборки $\mathfrak{D}$ при заданных $\mathbf{w}, \mathcal{A}$.

TODO: здесь абсолютно аналогично работе Смердова, вводим вар. вывод, вводим q и пр.

Базовые методы прореживания нейросети Случайное удаление Метод случайного удаления заключается в том, что случайным образом удаляется некоторый параметр w_ξ из множества активных параметров сети. Индекс

параметра ξ из равномерного распределения случайная величина, предположительно доставляющая оптимум в (2.11).

$$\xi \sim \mathcal{U}(\mathcal{A}). \tag{3.1.1}$$

Оптимальное прореживание Метод оптимального прореживания [?] использует вторую производную целевой функции (2.4) по параметрам для определения нерелевантных параметров. Рассмотрим функцию потерь $\mathcal{L}$ (2.4) разложенную в ряд Тейлора в некоторой окрестности вектора параметров $\mathbf{w}$:

$$\delta\mathcal{L} = \sum_{j\in\mathcal{A}} g_j \delta w_j + \frac{1}{2}\sum_{i,j\in\mathcal{A}} h_{ij}\delta w_i \delta w_j + O(||\delta\mathbf{w}||^3), \tag{3.2.1}$$

где δw_j — компоненты вектора $\delta\mathbf{w}$, g_j — компоненты вектора градиента $\nabla\mathcal{L}$, а h_{ij} — компоненты гессиана $\mathbf{H}$:

$$g_j = \frac{\partial\mathcal{L}}{\partial w_j}, \qquad h_{ij} = \frac{\partial^2\mathcal{L}}{\partial w_i \partial w_j}. \tag{3.2.2}$$

Задача является вычислительно сложной в силу высокой размерности матрицы $\mathbf{H}$. Введем предположение [?], о том что удаление нескольких параметров приводит к такому же изменению функции потерь $\mathcal{L}$, как и суммарное изменение при индивидуальном удалении:

$$\delta\mathcal{L} = \sum_{j\in\mathcal{A}} \delta\mathcal{L}_j, \tag{3.2.3}$$

где $\mathcal{A}$ — множество активных параметров, $\delta\mathcal{L}_j$ — изменение функции потерь при удалении одного параметра $\mathbf{w}_j$.

В силу данного предположения будем рассматривать только диагональные элементы матрицы $\mathbf{H}$. После введенного предположения, выражение (3.2.1) принимает вид

$$\delta\mathcal{L} = \frac{1}{2}\sum_{j\in\mathcal{A}} h_{jj}\delta w_j^2, \tag{3.2.4}$$

Получаем следующую задачу оптимизации:

$$\xi = \arg\min_{j\in\mathcal{A}} h_{jj}\frac{w_j^2}{2}, \tag{3.2.5}$$

где ξ — индекс наименее релевантного, удаляемого параметра, предположительно доставляющая оптимум в (2.11).

Удаление неинформативных параметров с помощью вариационного вывода. Для удаления параметров в работе [?] предлагается удалить параметры, которые имеют максимальное отношение плотности $p(\mathbf{w}|\mathcal{A})$ априорной вероятности в нуле к плотности вероятности априорной вероятности в

математическом ожидании μ_j параметра w_j.
Для гауссовского распределения с диагональной матрицей ковариации получаем:

$$p_j(\mathbf{w}|\mathcal{A})(w) = \frac{1}{\sqrt{2\sigma_j^2}} \exp(-\frac{(w-\mu_j)^2}{2\sigma_j^2}), \tag{3.3.1}$$

где w — значение носителя распределенного параметра. Разделим плотность вероятности в нуле к плотности в математическом ожидании

$$\frac{p_j(\mathbf{w}|\mathcal{A})(0)}{p_j(\mathbf{w}|\mathcal{A})(\mu_j)} = \exp(-\frac{\mu_j^2}{2\sigma_j^2}), \tag{3.3.2}$$

и поставим следующую задачу оптимизации:

$$\xi = \arg\min_{j\in\mathcal{A}} \left|\frac{\mu_j}{\sigma_j}\right|, \tag{3.3.3}$$

где ξ — индекс наименее релевантного, удаляемого параметра.

Предлагаемый метод определения релевантности параметров нейросети. Предлагается метод основанный, на модификации метода Белсли. Пусть $\mathbf{w}$ — вектор параметров, доставляющий минимум функционалу потерь $\mathcal{L}_\mathcal{A}$ из (2.8) на множестве $\mathbb{W}_\mathcal{A}$, а $\mathbf{A}_{\text{ps}}$ соответствующая ему ковариационная матрица.

Выполним сингулярное разложение матрицы

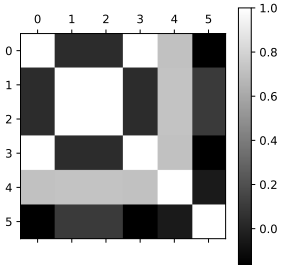
$$\mathbf{A}_{\text{ps}} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\mathsf{T}. \tag{4.1}$$

Индекс обусловленности η_j определим как отношение максимального элемента к j-му элементу матрицы $\mathbf{\Lambda}$. Для нахождения мультикоррелирующих признаков требуется найти индекс ξ вида:

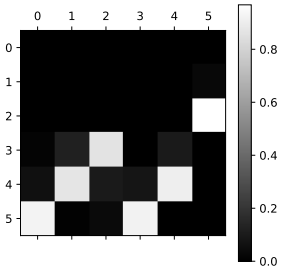
$$\xi = \arg\max_{j\in\mathcal{A}} \eta_j. \tag{4.2}$$

Таблица 2.2. Илюстрация метода Белсли

η	q_1	q_2	q_3	q_4	q_5	q_6
1.0	$2\cdot10^{-17}$	$4\cdot10^{-17}$	$1\cdot10^{-16}$	$2\cdot10^{-17}$	$6\cdot10^{-17}$	$3\cdot10^{-4}$
1.5	$5\cdot10^{-17}$	$9\cdot10^{-17}$	$2\cdot10^{-16}$	$5\cdot10^{-17}$	$3\cdot10^{-20}$	$3\cdot10^{-2}$
3.3	$9\cdot10^{-18}$	$1\cdot10^{-17}$	$2\cdot10^{-17}$	$9\cdot10^{-18}$	$2\cdot10^{-19}$	$9\cdot10^{-1}$
$2\cdot10^{15}$	$1\cdot10^{-2}$	$1\cdot10^{-1}$	$8\cdot10^{-1}$	$2\cdot10^{-3}$	$9\cdot10^{-2}$	$1\cdot10^{17}$
$8\cdot10^{15}$	$6\cdot10^{-2}$	$8\cdot10^{-1}$	$9\cdot10^{-2}$	$8\cdot10^{-2}$	$9\cdot10^{-1}$	$2\cdot10^{17}$
$1\cdot10^{16}$	$\mathbf{9\cdot10^{-1}}$	$1\cdot10^{-2}$	$4\cdot10^{-2}$	$\mathbf{9\cdot10^{-1}}$	$1\cdot10^{-3}$	$5\cdot10^{-21}$



(а) Матрица ковариации



(б) Дисперсионные доли

Рис. 2.6. Илюстрация метода Белсли

Дисперсионный долевой коэффициент q_{ij} определим как вклад j-го признака в дисперсию i-го элемента вектора параметра $\mathbf{w}$:

$$q_{ij} = \frac{u_{ij}^2/\lambda_{jj}}{\sum_{j=1}^{n} u_{ij}^2/\lambda_{jj}}. \tag{4.3}$$

Большие значение дисперсионных долей указывают на наличие зависимости между параметрами. Находим долевые коэффициенты, которые вносят максимальный вклад в дисперсию параметра w_ξ:

$$\zeta = \arg\max_{j \in \mathcal{A}} q_{\xi j}. \tag{4.4}$$

Параметр с индексом ζ определим как наименее релевантный параметр нейросети.

Проиллюстрируем принцип работы метода Белсли на примере. Рассмотрим данные порожденные следующим образом:

$$\mathbf{w} = \begin{bmatrix} \sin(x) \\ \cos(x) \\ 2{+}\cos(x) \\ 2{+}\sin(x) \\ \cos(x) + \sin(x) \\ x \end{bmatrix}$$

с матрицей ковариации на рис. 2.6.a, где $x \in [0.0, 0.02, ..., 20.0]$.

В табл. 2.2 приведены индексы обусловленности и соответствующие им дисперсионные доли, которые также изображены на рис. 2.6.b. Согласно этим данным, максимальный индекс обусловленности $\eta_6 = 1.2 \cdot 10^{16}$. Ему соответствуют максимальные дисперсионные доли признаков с индексами 1 и 4, которые, как видно из построения выборки, коррелируют между собой.

Вычислительный эксперимент. Для анализа свойств предложенного алгоритма и сравнения его с существующими был проведен вычислительный эксперимент в котором параметры нейросети удалялись методами, которые были описаны в разделах 3.1—3.3 и методом Белсли.

В качестве данных использовались три выборки. Выборки Wine [?] и Boston Housing [?] — это реальные данные. Синтетические данные сгенерированы таким образом, чтобы параметры сети были мультикоррелируемые. Генерация данных состояла из двух этапов. На первом этапе генерировался вектор параметров $\mathbf{w}_{\text{synthetic}}$:

$$\mathbf{w}_{\text{synthetic}} \sim \mathcal{N}(\mathbf{m}_{\text{synthetic}}, \mathbf{A}_{\text{synthetic}}), \tag{5.1}$$

где $\mathbf{m}_{\text{synthetic}} = \begin{bmatrix} 1.0 \\ 0.0025 \\ \cdots \\ 0.0025 \end{bmatrix}$, $\mathbf{A}_{\text{synthetic}} = \begin{bmatrix} 1.0 & 10^{-3} & \cdots & 10^{-3} & 10^{-3} \\ 10^{-3} & 1.0 & \cdots & 0.95 & 0.95 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 10^{-3} & 0.95 & \cdots & 0.95 & 1.0 \end{bmatrix}$.

На втором этапе генерировалась выборка $\mathfrak{D}_{\text{synthetic}}$:

$$\mathfrak{D}_{\text{synthetic}} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \sim \mathcal{N}(\mathbf{1}, \mathbf{I}), y_i = x_{i0}, i = 1 \cdots 10000\}. \tag{5.2}$$

В приведенном выше векторе параметров $\mathbf{w}_{\text{synthetic}}$ для выборки $\mathfrak{D}_{\text{synthetic}}$, наиболее релевантным является первый параметр, а все остальные параметры являются нерелевантными. Матрица ковариации была выбрана таким образом, чтобы все нерелевантные параметры были зависимы и метод Белсли был максимально эффективен.

Таблица 2.3. Описание выборок

Выборка	Тип задачи	Размер выборки	Число признаков
Wine	класификация	178	13
Boston Housing	регресия	506	13
Synthetic data	регресия	10000	100

Для алгоритмов тренировочная и тестовая выборки составили 80% и 20% соответсвенно. Критерием качества прореживания служит процент параметров нейросети, удаление которого не влечет значимой потери качества прогноза. Также критерием качества служит устойчивость нейросети к зашумленности данных.

Качеством прогноза R_{cl} модели для задачи классификации является точность прогноза модели:

$$R_{\text{cl}} = \frac{\sum_{(\mathbf{x},y)\in\mathfrak{D}}[f(\mathbf{x},\mathbf{w}) = y]}{|\mathfrak{D}|}, \tag{5.3}$$

Качеством прогноза R_{rg} модели для задачи регрессии является среднеквадратическое отклонение результата модели от точного:

$$R_{\text{rg}} = \frac{\sum_{(\mathbf{x},y)\in\mathfrak{D}}\left(f(\mathbf{x},\mathbf{w}) - y\right)^2}{|\mathfrak{D}|}, \tag{5.4}$$

Wine. Рассмотрим нейроную сеть с 13 нейронами на входе, 13 нейронами в скрытом слое и 3 нейронами на выходе.

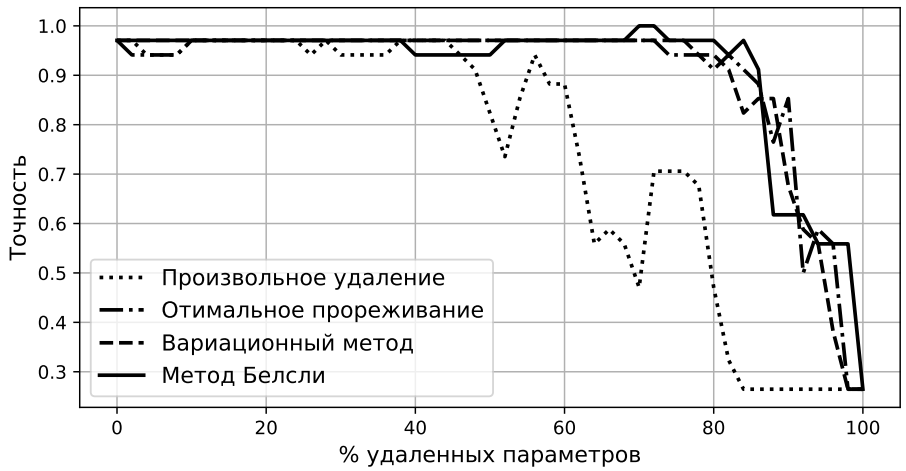
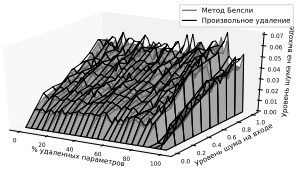


Рис. 2.7. Качество прогноза при удаление параметров на выборке Wine

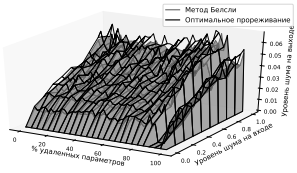
На рис. 2.7 показано как меняется точность прогноза R_{cl} при удалении параметров указанными методами. Из графика видно, что метод оптимального прореживания, вариационный метод и метод Белсли позволяют удалить $\approx 80\%$ параметров и качество всех этих методов падает при удалении $\approx 90\%$ параметров нейросети.

На рис. **??** показаны поверхности изменения уровня шума ответов нейросети при изменении процента удаленных параметров и уровня шума входных данных для разных методов прореживания. На графиках показано, что при удалении параметров нейросети методом Белсли шум меньше, чем при удалении параметров другими методами, на это указывает то что поверхность которая соответствует методу Белсли ниже других поверхностей.

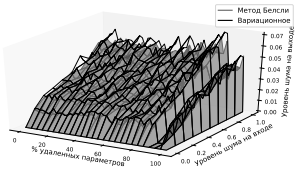
Boston Housing. Рассмотрим нейроную сеть с 13 нейронами на входе, 39 нейронами в скрытом слое и одним нейроном на выходе.



(а) Произвольное удаление параметров



(б) Оптимальное прореживание



(в) Вариационный метод

Рис. 2.8. Влияние шума в начальных данных на шум выхода нейросети на выборке Wine

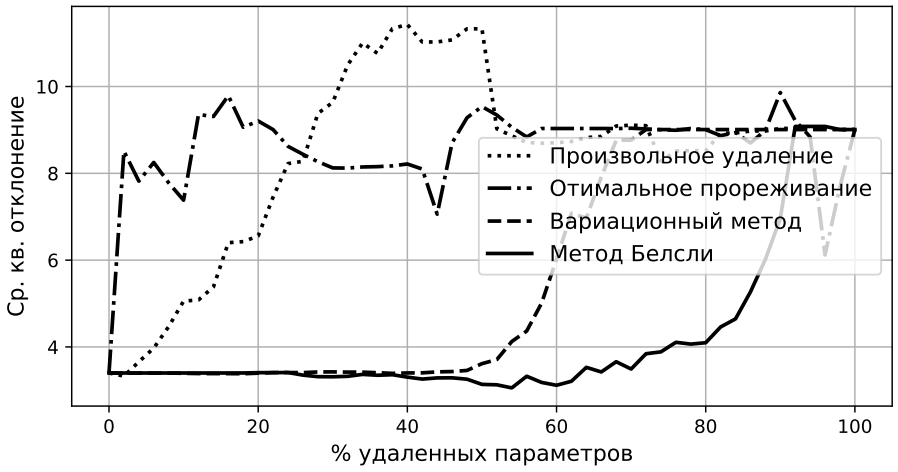
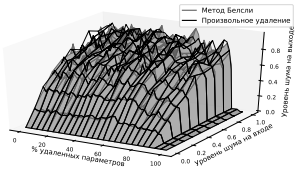


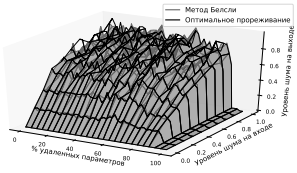
Рис. 2.9. Качество прогноза при удаление параметров на выборке Boston

На рис. 2.9 показано как меняется среднеквадратическое отклонение прогноза R_{rg} от точного ответа при удалении параметров указанными методами. График показывает, что метод Белсли является более эффективным, чем другие методы, так-как позволяет удалить больше параметров нейросети без потери качества.

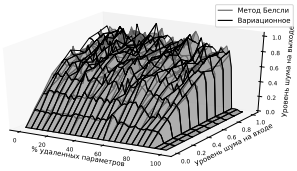
На рис. 2.10 показаны поверхности изменения уровня шума ответов нейросети при изменении процента удаленных параметров и уровня шума входных данных для разных методов прореживания. График показывает, что уровень



(а) Произвольное удаление параметров



(б) Оптимальное прореживание



(в) Вариационный метод

Рис. 2.10. Влияние шума в начальных данных на шум выхода нейросети на выборке Boston

шума всех методов одинаковый, так как поверхности всех методов находятся на одном уровне.

Синтетические данные. Рассмотрим нейроную сеть с 100 нейронами на входе и одним нейроном на выходе.

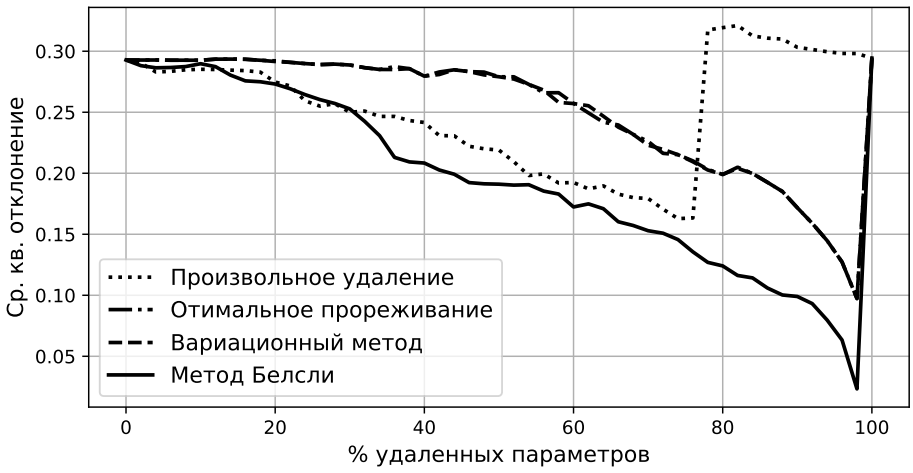
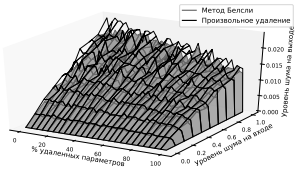
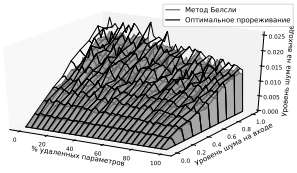


Рис. 2.11. Качество прогноза при удаление параметров на синтетической выборке

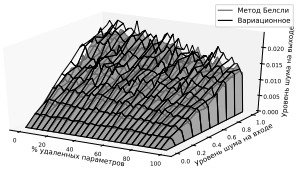
На рис. 2.11 показано как меняется среднеквадратическое отклонение прогноза от R_{rg} точного ответа при удалении параметров указанными методами.



(а) Произвольное удаление параметров



(б) Оптимальное прореживание



(в) Вариационный метод

Рис. 2.12. Влияние шума в начальных данных на шум выхода нейросети на синтетической выборке

График показывает, что удаление параметров методом Белсли является более эффективным чем другие методы прореживания, так-как качество прогноза нейросети улучшается при удалении шумовых параметров.

На рис. 2.12 показаны поверхности изменения уровня шума ответов нейросети при изменении процента удаленных параметров и уровня шума входных данных для разных методов прореживания. На графиках показано, что при удалении параметров нейросети методом Белсли шум меньше, чем при удалении параметров другими методами, так-как поверхность которая соответствует методу Белсли ниже других поверхностей.

Список использованных источников

1. *Liu Hanxiao, Simonyan Karen, Yang Yiming.* Darts: Differentiable architecture search // *arXiv preprint arXiv:1806.09055.* — 2018.
2. *Токмакова А. А., Стрижов В. В.* Оценивание гиперпараметров линейных и регрессионных моделей при отборе шумовых и коррелирующих признаков // *Информатика и её применения.* — 2012. — Т. 6(4). — С. 66–75. http://strijov.com/papers/Tokmakova2011HyperParJournal_Preprint.pdf.
3. *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.

4. *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.
5. *Li Jundong, Liu Huan.* Challenges of feature selection for big data analytics // *IEEE Intelligent Systems.* — 2017. — Vol. 32, no. 2. — Pp. 9–15.
6. *Grünwald Peter.* A Tutorial Introduction to the Minimum Description Length Principle // Advances in Minimum Description Length: Theory and Applications. — MIT Press, 2005.
7. *Bishop Christopher M.* Pattern Recognition and Machine Learning (Information Science and Statistics). — Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
8. AdaNet: Adaptive Structural Learning of Artificial Neural Networks / Corinna Cortes, Xavier Gonzalvo, Vitaly Kuznetsov et al. // International Conference on Machine Learning. — 2017. — Pp. 874–883.
9. *Перекрестенко Д.О.* Анализ структурной и статистической сложности суперпозиции нейронных сетей. — 2014. http://sourceforge.net/p/mlalgorithms/code/HEAD/tree/Group074/Perekrestenko2014Comple
10. *Vladislavleva E.* Other publications TiSEM: : Tilburg University, School of Economics and Management, 2008. http://EconPapers.repec.org/RePEc:tiu:tiutis:65a72d10-6b09-443f-8cb9-88f3bb3bc31b.
11. Predicting Parameters in Deep Learning / Misha Denil, Babak Shakibi, Laurent Dinh et al. // Advances in Neural Information Processing Systems 26 / Ed. by C.j.c. Burges, L. Bottou, M. Welling et al. — 2013. — Pp. 2148–2156. http://media.nips.cc/nipsbooks/nipspapers/paper_files/nips26/1053.pdf.
12. *Xu Huan, Mannor Shie.* Robustness and generalization // *Machine Learning.* — 2012. — Vol. 86, no. 3. — Pp. 391–423. http://dx.doi.org/10.1007/s10994-011-5268-1.
13. Intriguing properties of neural networks. / Christian Szegedy, Wojciech Zaremba, Ilya Sutskever et al. // *CoRR.* — 2013. — Vol. abs/1312.6199. http://dblp.uni-trier.de/db/journals/corr/corr1312.html#SzegedyZSBEGF13.
14. *MacKay David J. C.* Information Theory, Inference & Learning Algorithms. — New York, NY, USA: Cambridge University Press, 2002.
15. *Зайцев А. А., Стрижов В. В., Токмакова А. А.* Оценка гиперпараметров регрессионных моделей методом максимального правдоподобия // *Информационные технологии.* — 2013. — Vol. 2. — Pp. 11–15. http://strijov.com/papers/ZaytsevStrijovTokmakova2012Likelihood_Preprint.pdf.
16. *Strijov V., Weber Gerhard-Wilhelm.* NONLINEAR REGRESSION MODEL GENERATION USING HYPERPARAMETERS OPTIMIZATION: Preprint

2009-21. — Middle East Technical University, 06800 Ankara, Turkey: Institute of Applied Mathematics, 2009. — Октябрь. — Preprint No. 149.

17. *Стрижов В. В.* Порождение и выбор моделей в задачах регрессии и классификации: Ph.D. thesis / Вычислительный центр РАН. — 2014. http://strijov.com/papers/Strijov2015ModelSelectionRu.pdf.
18. Stochastic Variational Inference / Matthew D. Hoffman, David M. Blei, Chong Wang, John Paisley // *J. Mach. Learn. Res.* — 2013. — Май. — Vol. 14, no. 1. — Pp. 1303–1347. http://dl.acm.org/citation.cfm?id=2502581.2502622.
19. *Salimans Tim, Kingma Diederik P., Welling Max.* Markov Chain Monte Carlo and Variational Inference: Bridging the Gap. // ICML / Ed. by Francis R. Bach, David M. Blei. — Vol. 37 of *JMLR Proceedings*. — JMLR.org, 2015. — Pp. 1218–1226. http://dblp.uni-trier.de/db/conf/icml/icml2015.html#SalimansKW15.
20. *Maclaurin Dougal, Duvenaud David K., Adams Ryan P.* Early Stopping is Nonparametric Variational Inference // *CoRR*. — 2015. — Vol. abs/1504.01344. http://arxiv.org/abs/1504.01344.
21. *Mandt Stephan, Hoffman Matthew D, Blei David M.* Continuous-Time Limit of Stochastic Gradient Descent Revisited.
22. *Welling Max, Teh Yee Whye.* Bayesian Learning via Stochastic Gradient Langevin Dynamics // Proceedings of the 28th International Conference on Machine Learning (ICML-11) / Ed. by Lise Getoor, Tobias Scheffer. — ICML '11. — New York, NY, USA: ACM, 2011. — June. — Pp. 681–688.
23. *Arlot Sylvain, Celisse Alain.* A survey of cross-validation procedures for model selection // *Statist. Surv.* — 2010. — Vol. 4. — Pp. 40–79. http://dx.doi.org/10.1214/09-SS054.
24. Fast and Accurate Support Vector Machines on Large Scale Systems / Abhinav Vishnu, Jeyanthi Narasimhan, Lawrence Holder et al. // 2015 IEEE International Conference on Cluster Computing, CLUSTER 2015, Chicago, IL, USA, September 8-11, 2015. — 2015. — Pp. 110–119. http://dx.doi.org/10.1109/CLUSTER.2015.26.
25. Cross-validation pitfalls when selecting and assessing regression and classification models / Damjan Krstajic, Ljubomir J. Buturovic, David E. Leahy, Simon Thomas // *Journal of Cheminformatics*. — 2014. — Vol. 6, no. 1. — Pp. 1–15. http://dx.doi.org/10.1186/1758-2946-6-10.
26. *Hornung Roman, Bernau Christoph, Truntzer Caroline et al.* Full versus incomplete cross-validation: measuring the impact of imperfect separation between training and test sets in prediction error estimation. — 2014. http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-20682-6.
27. *Bengio Yoshua, Grandvalet Yves.* No Unbiased Estimator of the Variance of K-Fold Cross-Validation // *J. Mach. Learn. Res.* — 2004. — Декабрь. — Vol. 5. — Pp. 1089–1105. http://dl.acm.org/citation.cfm?id=1005332.1044695.

28. *Cun Yann Le, Denker John S., Solla Sara A.* Optimal Brain Damage // Advances in Neural Information Processing Systems. — Morgan Kaufmann, 1990. — Pp. 598–605.
29. *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.
30. Incremental network quantization: Towards lossless cnns with low-precision weights / Aojun Zhou, Anbang Yao, Yiwen Guo et al. // *arXiv preprint arXiv:1702.03044*. — 2017.
31. *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.
32. 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.
33. 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.
34. *Louizos Christos, Ullrich Karen, Welling Max.* Bayesian compression for deep learning // Advances in Neural Information Processing Systems. — 2017. — Pp. 3290–3300.
35. *Bergstra James, Bengio Yoshua.* Random search for hyper-parameter optimization // *Journal of Machine Learning Research*. — 2012. — Vol. 13, no. Feb. — Pp. 281–305.
36. Algorithms for hyper-parameter optimization / James S Bergstra, Rémi Bardenet, Yoshua Bengio, Balázs Kégl // Advances in Neural Information Processing Systems. — 2011. — Pp. 2546–2554.
37. *Bengio Yoshua.* Gradient-based optimization of hyperparameters // *Neural computation*. — 2000. — Vol. 12, no. 8. — Pp. 1889–1900.
38. DrMAD: Distilling Reverse-Mode Automatic Differentiation for Optimizing Hyperparameters of Deep Neural Networks / Jie Fu, Hongyin Luo, Jiashi Feng et al. // *arXiv preprint arXiv:1601.00917*. — 2016.
39. *Pedregosa Fabian.* Hyperparameter optimization with approximate gradient // Proceedings of the 33rd International Conference on Machine Learning. — 2016.
40. Scalable Gradient-Based Tuning of Continuous Regularization Hyperparameters / Jelena Luketina, Tapani Raiko, Mathias Berglund, Klaus Greff // Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 / Ed. by

Maria-Florina Balcan, Kilian Q. Weinberger. — Vol. 48 of *JMLR Workshop and Conference Proceedings*. — JMLR.org, 2016. — Pp. 2952–2960.

41. *Snoek Jasper, Larochelle Hugo, Adams Ryan P.* Practical bayesian optimization of machine learning algorithms // Advances in neural information processing systems. — 2012. — Pp. 2951–2959.
42. Bayesian Optimization in High Dimensions via Random Embeddings. / Ziyu Wang, Masrour Zoghi, Frank Hutter et al. // IJCAI. — 2013. — Pp. 1778–1784.
43. Bayesian Optimization with Tree-structured Dependencies / Rodolphe Jenatton, Cedric Archambeau, Javier González, Matthias Seeger // International Conference on Machine Learning. — 2017. — Pp. 1655–1664.
44. 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.
45. 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.
46. 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.
47. 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.
48. *Воронцов Константин Вячеславович.* Локальные базисы в алгебраическом подходе к проблеме распознавания: Ph.D. thesis. — Graz, 1999.
49. *Abadi Martín, Agarwal Ashish, Barham Paul et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. — 2015. — Software available from tensorflow.org. http://tensorflow.org/.
50. *Theano Development Team.* Theano: A Python framework for fast computation of mathematical expressions // *arXiv e-prints*. — 2016. — may. — Vol. abs/1605.02688. http://arxiv.org/abs/1605.02688.
51. Automatic differentiation in PyTorch / Adam Paszke, Sam Gross, Soumith Chintala et al. — 2017.
52. *Eibe Frank, Hall MA, Witten IH.* The WEKA Workbench. Online Appendix for"Data Mining: Practical Machine Learning Tools and Techniques // *Morgan Kaufmann*. — 2016.
53. *Hofmann Markus, Klinkenberg Ralf.* RapidMiner: Data mining use cases and business analytics applications. — CRC Press, 2013.

54. Scikit-learn: Machine learning in Python / Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort et al. // *Journal of machine learning research.* — 2011. — Vol. 12, no. Oct. — Pp. 2825–2830.
55. Relational inductive biases, deep learning, and graph networks / Peter W Battaglia, Jessica B Hamrick, Victor Bapst et al. // *arXiv preprint arXiv:1806.01261.* — 2018.
56. *Negrinho Renato, Gordon Geoff.* Deeparchitect: Automatically designing and training deep architectures // *arXiv preprint arXiv:1704.08792.* — 2017.
57. 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.
58. *Alvarez-Melis David, Jaakkola Tommi S.* Tree-structured decoding with doubly-recurrent neural networks. — 2016.
59. *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.
60. *Feng Jiashi, Darrell Trevor.* Learning the structure of deep convolutional networks // Proceedings of the IEEE international conference on computer vision. — 2015. — Pp. 2749–2757.
61. *Shirakawa Shinichi, Iwata Yasushi, Akimoto Youhei.* Dynamic Optimization of Neural Network Structures Using Probabilistic Modeling // *arXiv preprint arXiv:1801.07650.* — 2018.
62. Toward Optimal Run Racing: Application to Deep Learning Calibration / Olivier Bousquet, Sylvain Gelly, Karol Kurach et al. // *arXiv preprint arXiv:1706.03199.* — 2017.
63. Approximation and learning by greedy algorithms / Andrew R. Barron, Albert Cohen, Wolfgang Dahmen, Ronald A. DeVore // *Ann. Statist.* — 2008. — 02. — Vol. 36, no. 1. — Pp. 64–94. http://dx.doi.org/10.1214/009053607000000631.
64. *Tzikas Dimitris, Likas Aristidis.* An Incremental Bayesian Approach for Training Multilayer Perceptrons // Artificial Neural Networks – ICANN 2010: 20th International Conference, Thessaloniki, Greece, September 15-18, 2010, Proceedings, Part I / Ed. by Konstantinos Diamantaras, Wlodek Duch, Lazaros S. Iliadis. — Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. — Pp. 87–96. http://dx.doi.org/10.1007/978-3-642-15819-3_12.
65. *Tipping Michael E.* Sparse Bayesian Learning and the Relevance Vector Machine // *J. Mach. Learn. Res.* — 2001. — Сентябрь. — Vol. 1. — Pp. 211–244. http://dx.doi.org/10.1162/15324430152748236.
66. Greedy Layer-Wise Training of Deep Networks / Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle // Advances in Neural Information Processing Systems 19 / Ed. by B. Schölkopf,

J. C. Platt, T. Hoffman. — MIT Press, 2007. — Pp. 153–160. http://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.pdf.

67. *Hinton Geoffrey E., Osindero Simon, Teh Yee-Whye.* A Fast Learning Algorithm for Deep Belief Nets // *Neural Comput.* — 2006. — Июль. — Vol. 18, no. 7. — Pp. 1527–1554. http://dx.doi.org/10.1162/neco.2006.18.7.1527.
68. 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.
69. *Li Yi, Shapiro L. O., Bilmes J. A.* A generative/discriminative learning algorithm for image classification // Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. — Vol. 2. — 2005. — Oct. — Pp. 1605–1612 Vol. 2.
70. *J. Lasserre.* Hybrid of generative and discriminative methods for machine learning: Ph.D. thesis / University of Cambridge. — 2008.
71. Learning deep resnet blocks sequentially using boosting theory / Furong Huang, Jordan Ash, John Langford, Robert Schapire // *arXiv preprint arXiv:1706.04964.* — 2017.
72. Progressive neural architecture search / Chenxi Liu, Barret Zoph, Jonathon Shlens et al. // *arXiv preprint arXiv:1712.00559.* — 2017.
73. *Alain Guillaume, Bengio Yoshua.* Understanding intermediate layers using linear classifier probes // *arXiv preprint arXiv:1610.01644.* — 2016.
74. *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.
75. Incremental Training of Deep Convolutional Neural Networks / R Istrate12, ACI Malossi, C Bekas, D Nikolopoulos.
76. *Zoph Barret, Le Quoc V.* Neural architecture search with reinforcement learning // *arXiv preprint arXiv:1611.01578.* — 2016.
77. Accelerating neural architecture search using performance prediction / Bowen Baker, Otkrist Gupta, Ramesh Raskar, Nikhil Naik // *CoRR, abs/1705.10823.* — 2017.
78. Learning transferable architectures for scalable image recognition / Barret Zoph, Vijay Vasudevan, Jonathon Shlens, Quoc V Le // *arXiv preprint arXiv:1707.07012.* — 2017.
79. Efficient Architecture Search by Network Transformation / Han Cai, Tianyao Chen, Weinan Zhang et al. — 2018.
80. *Chen Tianqi, Goodfellow Ian, Shlens Jonathon.* Net2net: Accelerating learning via knowledge transfer // *arXiv preprint arXiv:1511.05641.* — 2015.

81. 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.
82. *Miranda Conrado S, Von Zuben Fernando J.* Reducing the Training Time of Neural Networks by Partitioning // *arXiv preprint arXiv:1511.02954*. — 2015.
83. *Schmidhuber Juergen, Zhao Jieyu, Wiering MA.* Simple principles of metalearning // *Technical report IDSIA*. — 1996. — Vol. 69. — Pp. 1–23.
84. *Schmidhuber Jürgen.* A neural network that embeds its own meta-levels // Neural Networks, 1993., IEEE International Conference on / IEEE. — 1993. — Pp. 407–412.
85. Meta-SGD: Learning to Learn Quickly for Few Shot Learning / Zhenguo Li, Fengwei Zhou, Fei Chen, Hang Li // *arXiv preprint arXiv:1707.09835*. — 2017.
86. *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.
87. 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.
88. *Kinga D, Adam J Ba.* A method for stochastic optimization // International Conference on Learning Representations (ICLR). — Vol. 5. — 2015.
89. *Duchi John, Hazan Elad, Singer Yoram.* Adaptive subgradient methods for online learning and stochastic optimization // *Journal of Machine Learning Research*. — 2011. — Vol. 12, no. Jul. — Pp. 2121–2159.
90. *Friesen Abram L, Domingos Pedro.* Deep Learning as a Mixed Convex-Combinatorial Optimization Problem // *arXiv preprint arXiv:1710.11573*. — 2017.
91. *Kristiansen Gus, Gonzalvo Xavi.* EnergyNet: Energy-based Adaptive Structural Learning of Artificial Neural Network Architectures // *arXiv preprint arXiv:1711.03130*. — 2017.
92. Pathnet: Evolution channels gradient descent in super neural networks / Chrisantha Fernando, Dylan Banarse, Charles Blundell et al. // *arXiv preprint arXiv:1701.08734*. — 2017.
93. *Veniat Tom, Denoyer Ludovic.* Learning time-efficient deep architectures with budgeted super networks // *arXiv preprint arXiv:1706.00046*. — 2017.
94. *Salakhutdinov Ruslan, Hinton Geoffrey E.* Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure // Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07) / Ed. by Marina Meila, Xiaotong Shen. — Vol. 2. — Journal of Machine Learning Research - Proceedings Track, 2007. — Pp. 412–419. http://jmlr.csail.mit.edu/proceedings/papers/v2/salakhutdinov07a/salakhutdinov07a.pdf.

95. *Cho Kyunghyun.* Foundations and Advances in Deep Learning: G5 Artikkeliväitöskirja. — Aalto University; Aalto-yliopisto, 2014. — P. 277. http://urn.fi/URN:ISBN:978-952-60-5575-6.
96. *Alain Guillaume, Bengio Yoshua.* What regularized auto-encoders learn from the data-generating distribution // *Journal of Machine Learning Research.* — 2014. — Vol. 15, no. 1. — Pp. 3563–3593. http://dl.acm.org/citation.cfm?id=2750359.
97. *Kamyshanska Hanna, Memisevic Roland.* On autoencoder scoring // Proceedings of the 30th International Conference on Machine Learning (ICML-13) / Ed. by Sanjoy Dasgupta, David Mcallester. — Vol. 28. — JMLR Workshop and Conference Proceedings, 2013. — Май. — Pp. 720–728. http://jmlr.org/proceedings/papers/v28/kamyshanska13.pdf.
98. *D. Kingma M. Welling.* Auto-Encoding Variational Bayes // Proceedings of the International Conference on Learning Representations (ICLR). — 2014.
99. How to Train Deep Variational Autoencoders and Probabilistic Ladder Networks. / Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe et al. // *CoRR.* — 2016. — Vol. abs/1602.02282. http://dblp.uni-trier.de/db/journals/corr/corr1602.html#SonderbyRMSW16.
100. Semi-Supervised Learning with Ladder Network. / Antti Rasmus, Harri Valpola, Mikko Honkala et al. // *CoRR.* — 2015. — Vol. abs/1507.02672. http://dblp.uni-trier.de/db/journals/corr/corr1507.html#RasmusVHBR15.
101. 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.
102. *Nalisnick Eric, Smyth Padhraic.* Deep Generative Models with Stick-Breaking Priors // *arXiv preprint arXiv:1605.06197.* — 2016.
103. *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.
104. *Miller A. C., Foti N., Adams R. P.* Variational Boosting: Iteratively Refining Posterior Approximations // *ArXiv e-prints.* — 2016. — nov.
105. *Arnold Ludovic, Ollivier Yann.* Layer-wise learning of deep generative models // *arXiv preprint arXiv:1212.1524.* — 2012.
106. *Karaletsos Theofanis, Rätsch Gunnar.* Automatic Relevance Determination For Deep Generative Models // *arXiv preprint arXiv:1505.07765.* — 2015.
107. *Längkvist Martin, Karlsson Lars, Loutfi Amy.* A review of unsupervised feature learning and deep learning for time-series modeling // *Pattern Recognition Letters.* — 2014. — Vol. 42. — Pp. 11–24.

108. Evolving deep recurrent neural networks using ant colony optimization / Travis Desell, Sophine Clachar, James Higgins, Brandon Wild // European Conference on Evolutionary Computation in Combinatorial Optimization / Springer. — 2015. — Pp. 86–98.
109. *Popova Mariya Sergeevna, Strijov Vadim.* Building superposition of deep learning neural networks for solving the problem of time series classification // *Sistemy i Sredstva Informatiki [Systems and Means of Informatics].* — 2015. — Vol. 25, no. 3. — Pp. 60–77.
110. Intriguing properties of neural networks / Christian Szegedy, Wojciech Zaremba, Ilya Sutskever et al. // *arXiv preprint arXiv:1312.6199.* — 2013.
111. *Sutskever Ilya, Hinton Geoffrey E, Taylor Graham W.* The recurrent temporal restricted boltzmann machine // Advances in neural information processing systems. — 2009. — Pp. 1601–1608.
112. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection / Richard Socher, Eric H Huang, Jeffrey Pennin et al. // Advances in neural information processing systems. — 2011. — Pp. 801–809.
113. *Kwapisz Jennifer R, Weiss Gary M, Moore Samuel A.* Activity recognition using cell phone accelerometers // *ACM SigKDD Explorations Newsletter.* — 2011. — Vol. 12, no. 2. — Pp. 74–82.
114. *Hinton Geoffrey E.* Learning multiple layers of representation // *Trends in cognitive sciences.* — 2007. — Vol. 11, no. 10. — Pp. 428–434.
115. *Cho Kyung Hyun, Raiko Tapani, Ilin Alexander.* Gaussian-bernoulli deep boltzmann machine // Neural Networks (IJCNN), The 2013 International Joint Conference on / IEEE. — 2013. — Pp. 1–7.
116. *Shang Yi, Wah B. W.* Global optimization for neural network training // *Computer.* — 1996. — Mar. — Vol. 29, no. 3. — Pp. 45–54.