Конспект

Т.Р. Жангиров

Нейронные сети

# Классификация изображений. Подход управления данными. K-n алгоритм

## Классификатор ближайших соседей

Одной из важнейших задач анализа данных является классификация — отнесение объектов предметной области к заранее определённым группам, называемым классами. При этом каждому классу должны принадлежать объекты, близкие по своим свойствам. Обобщая свойства известных объектов класса на новые, отнесённые к нему объекты, можно получать знания о них.

Задача классификации решается с помощью аналитических моделей, называемых классификаторами. Классифицировать объект означает предъявить набор его признаков (обычно представленных в виде вектора) на вход модели-классификатора, которая должна присвоить ему метку или номер класса.

Типичным представителем методов классификации, использующих эту логику, является метод k-ближайших соседей — **k-nearest neighbors algorithm** (KNN). Метод относится к классу непараметрических, т.е. не требует предположений о том, из какого статистического распределения была сформирована обучающее множество. Следовательно, классификационные модели, построенные с помощью метода KNN также будут непараметрическими. Это означает, что структура модели не задаётся жёстко изначально, а определяется данными.

### Алгоритм

Пусть имеется набор данных, состоящий из n наблюдений   
, для каждого из которых задан класс . Тогда на его основе может быть сформировано обучающее множество, все примеры которого представляют собой пары  ​.

Алгоритм KNN можно разделить на две простые фазы: обучения и классификации. При обучении алгоритм просто запоминает векторы признаков наблюдений и их метки классов (т.е. примеры). Также задаётся параметр алгоритма k, который задаёт число «соседей», которые будут использоваться при классификации.

На фазе классификации предъявляется новый объект, для которого метка класса не задана. Для него определяются k ближайших (в смысле некоторой метрики) предварительно классифицированных наблюдений. Затем выбирается класс, которому принадлежит большинство из k ближайших примеров-соседей, и к этому же классу относится классифицируемый объект.

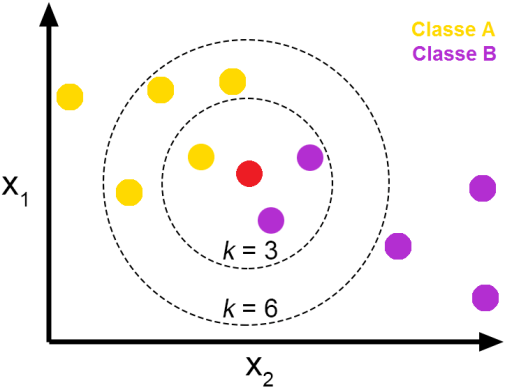


Рисунок 1. Пример работы KNN при k = 3 и 6

Кружком представлен объект, который требуется классифицировать, отнеся к одному из двух классов «треугольники» и «квадраты». Если выбрать k=3, то из трёх ближайших объектов два окажутся «треугольниками» и один «квадратом». Следовательно, новому объекту будет присвоен класс «треугольник». Если задать k=5, то из пяти «соседей» два будут «треугольники» и три «квадраты», в результате классифицируемый объект будет распознан как «квадрат».

### Особенности алгоритма

В обычном случае используют так называемое простое невзвешенное голосование (simple unweighted voting). При это предполагается, что все k примеров имеют одинаковое право «голоса» независимо от расстояния до классифицируемого объекта.

Однако, логично предположить, что чем дальше пример расположен от классифицируемого объекта в пространстве признаков, тем ниже его значимость для определения класса. Поэтому для улучшения результатов классификации вводят взвешивание примеров в зависимости от их удалённости. В этом случае используют взвешенное голосование (weighted voting).

В основе идеи взвешенного голосования лежит введение «штрафа» для класса, в зависимости от того, насколько относящиеся к нему примеры удалены от классифицируемого объекта. Такой «штраф» представляется как сумма величин, обратных квадрату расстояний от примера j-го класса до классифицируемого объекта (часто данное значение называют показателем близости):



где  - оператор вычисления расстояния,  - вектор признаков классифицируемого объекта,  - i-й пример j-го класса.

Таким образом, «побеждает» тот класс, для которого величина  окажется наибольшей. При этом также снижается вероятность того, что классы получат одинаковое число голосов.

### Выбор значения параметра k

Выбор параметра k является важным для получения корректных результатов классификации. Если значение параметра мало, то возникает эффект переобучения, когда решение по классификации принимается на основе малого числа примеров и имеет низкую значимость. Если установить k=1k=1, то алгоритм будет просто присваивать любому новому наблюдению метку класса ближайшего объекта. Кроме этого, следует учитывать, что использование небольших значений k увеличивает влияние шумов на результаты классификации, когда небольшие изменения в данных приводят к большим изменениям в результатах классификации. Но при этом границы классов оказываются более выраженными (класс при голосовании побеждает с большим счётом).

Напротив, если значение параметра слишком велико, то в процессе классификации принимает участие много объектов, относящихся к разным классам. Такая классификация оказывается слишком грубой и плохо отражает локальные особенности набора данных. Таким образом, выбор параметра k является компромиссом между точностью и обобщающей способностью модели. При больших значениях параметра k уменьшается зашумленность результатов классификации, но снижается выраженность границ классов.

В задачах бинарной классификации бывает целесообразно выбрать k как нечетное число, так как это позволяет избежать равенства «голосов» при определении класса для нового наблюдения.

## Наборы проверки для настройки гиперпараметров

Оптимизация гиперпараметров — задача машинного обучения по выбору набора оптимальных гиперпараметров для обучающего алгоритма.

Одни и те же виды моделей машинного обучения могут требовать различные предположения, веса, скорости обучения и других параметров для различных видов данных. Эти параметры называются гиперпараметрами и их следует настраивать так, чтобы модель могла оптимально решить задачу обучения. Для этого находится кортеж гиперпараметров, который даёт оптимальную модель, оптимизирующую заданную функцию потерь на заданных независимых данных. Целевая функция берёт кортеж гиперпараметров и возвращает связанные с ними потери.

Для поиска оптимальных гиперпараметров данные разбивают две выборки: обучающая и тестовая. На обучающей выборке происходит обучение выбранной модели, то есть нахождение оптимальных параметров модели. Тестовая выборка используется для оценки уже обученной модели на таких данных, которые не участвовали в обучении. Часто используется перекрёстная проверка для оценки этой обобщающей способности, которая будет рассмотрена далее.

## Применение KNN на практике

Пример бинарной классификации изображений с использованием метода KNN: https://russianblogs.com/article/9582214954/

# Линейная классификация: SVM/Sotfmax

## Введение в линейную классификацию

Линейный классификатор — способ решения задач классификации, когда решение принимается на основании линейного оператора над входными данными. Класс задач, которые можно решать с помощью линейных классификаторов, обладают, соответственно, свойством линейной разделимостью. Два множества точек в двумерном пространстве называются линейной разделимыми, если они могут быть полностью отделены друг от друга единственной прямой. Для n-мерного пространства два набора точек линейно разделимы, если они могут быть отделены (n-1)-мерной гиперплоскостью.

Пусть объекты описываются n числовыми признаками . Тогда пространство признаковых описаний объектов есть . Пусть Y – конечное множество номеров (имён, меток) классов.

### Случай двух классов

Допустим, что . Тогда линейным классификатором называется алгоритм классификации  вида:



где  - вес j-го признака,  - порог принятия решения,  - вектор весов,  - скалярное произведение признакового описания объекта на вектор его весов.

Предполагается, что искусственно введён «константный» нулевой признак .

### Случай произвольного числа классов

В данном случае линейный классификатор определяется выражанием:



где каждому классу соответствует свой вектор весов .

В случае двух классов во время обучения классификатора идет поиск такой гиперплоскости, что один класс лежит «ниже» плоскости, а другой «выше» плоскости. Отсюда следует, что вектор весов можно интерпретировать как вектор нормали, задающий плоскость, а значит можно интерпретировать важность тех или иных признаков для задачи классификации.

В случае нескольких классов классификатор обучается так, чтобы для каждого класса найти такие вектора, которые лучше всего описывают влияние признаков для этого класса. Это также можно интерпретировать так, что вектор каждого класса описывает нормаль плоскости отделяющий класс от всех остальных классов.

## Функция потерь

Для обучения модели необходимо иметь возможность ее оценить, то есть количественно рассчитать, насколько хорошо или плохо модель решает поставленную задачу. Для оценки такого качества используются функции потери. Функция потерь на вход принимает два параметра: прогнозируемые выходные данные (результат модели) и истинные выходные данные. На основе этих двух параметров, функция потери отображает результат работы алгоритмы в пространство , показывая «стоимость» ошибки. Так как функция потери показывает стоимость ошибки то, чем лучше полученная модель, тем меньше значение этой функции.

Существует большое количество различных функций потерь используемых в различных типах задач и для разных форматов данных. Далее представлены наиболее часто-встречающиеся функции потерь:

* 0-1 функция -  . Равна 0 только при полном совпадении результатов
* Квадратичная функция - 
* Абсолютная функция - 
* Потеря шарнира - . Применяется, когда метки класса равны -1 и +1. В основном используется в SVM, и показывает не только неправильные прогнозы, но и отображает «неуверенные» правильные прогнозы.
* Логарифмическая - 

Также, с понятие функции связано понятие эмпирического риска. Эмпирический риск – это средняя величина ошибки на обучающем наборе:



Отсюда следует предположение, что если модель хорошо показывает себя на обучающей выборке, то и на реальных данных он будет показывать результат близкий к хорошему. Это приводит к тому, что один из методов обучения моделей (обучение с учителем) это минимизация эмпирического риска .

## Мультиклассовый SVM

Метод опорных векторов (англ. SVM, support vector machine) - набор схожих алгоритмов обучения с учителем, использующихся для задач классификации и регрессионного анализа. Принадлежит семейству линейных классификаторов.

Основная идея метода — перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с наибольшим зазором в этом пространстве. Две параллельных гиперплоскости строятся по обеим сторонам гиперплоскости, разделяющей классы. Разделяющей гиперплоскостью будет гиперплоскость, создающая наибольшее расстояние до двух параллельных гиперплоскостей. Алгоритм основан на допущении, что чем больше разница или расстояние между этими параллельными гиперплоскостями, тем меньше будет средняя ошибка классификатора.

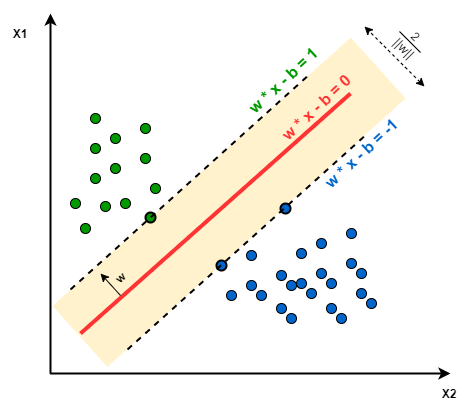


Рисунок 2 – Метод опорных векторов для двух классов в двумерном пространстве

На рисунке 2 изображен пример применения SVM для классификации двух классов (зеленые и синие точки). Штрихованные линии отображают границы классов, на которых лежат точки из классов. Эти границы задают область неопределенности, то есть точки из данной области относятся к классу уже с некоторой вероятностью. Красная линия показывает вектор, который является границей принятия решения о принадлежности к классу.

В случае количества классов большем чем 2 (многоклассовая ситуация), SVM можно применить двумя разными способами: разделение классов один к одному и разделение классов один к остальным.

В случае разделения один к одному, строится векторов, где m - количество классов. Эти вектора строятся между каждой парой классов, без учета точек остальных классов. На рисунке 3 изображен пример полученных векторов для трех классов в двумерном пространстве.

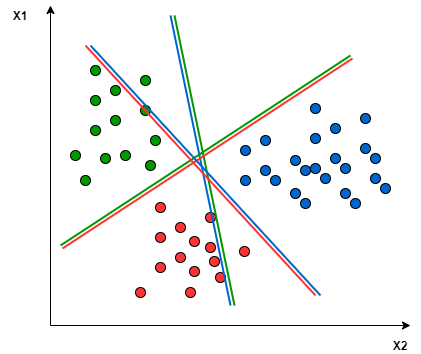


Рисунок 3 – SVM при разделении один к одному

В случае один к остальным вектора строятся так, чтобы отделить один класс от всех остальных. В данном случае будет построено m векторов, где m – количество классов. Данный случай изображен на рисунке 4

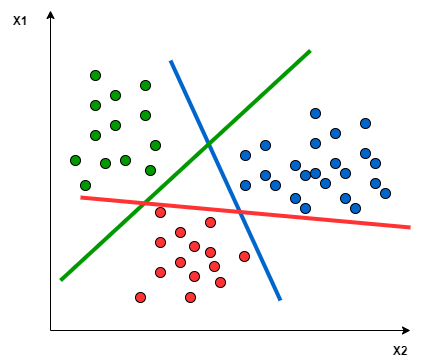


Рисунок 4 – SVM при разделении один к остальным

## Softmax классификатор

Softmax — это обобщение логистической функции для многомерного случая. Функция преобразует вектор размерности  в той же размерности, где каждая координата  полученного вектора представлена вещественным числом в интервале (0,1) и сумма координат равна 1.

Координаты  вычисляются следующим образом:



Если в случае SVM классификатор подсчитывает корректные классы, которые больше некорректных более чем на заданное значение, то в случае Softmax классификатора применяется другая функция потерь:



Внутри логарифма находится softmax-функция, которую можно интерпретировать как вектор вероятностей корректных оценок в диапазоне от 0 до 1.

При обучении, softmax-классификатор сводит к минимум кросс-энтропию между распределением вероятностей верного предсказания классов по всем исследуемым объектам и истинным распределением, в котором вероятность, равная 1, очевидно соответствует только одному из объектов, а вероятности остальных равны 0. То есть, задача минимизации функции потерь сводится к тому, чтобы максимизировать вероятность правильной классификации, или минимизации расстояния между двумя распределениями.

За счет этого основное различие между SVM и Softmax заключается в том, что softmax проще интерпретировать с точки зрения вероятности, а также можно применять в нечеткой классификации. В остальном, на практике оба метода дают примерно одинаковый результат и не сильно отличаются по производительности.

# Оптимизация, Высокоуровневое представление

### Визуализация функции потерь

При построении моделей машинного обучения необходимо искать оптимальные параметры модели. Для облегчения поиска необходимых значений параметров обычно проводят визуализацию значений функции потерь в зависимости от группы параметров.

В случае, когда зависимость строится только от одного параметра, то достаточно обычного графика зависимости функции потерь (ось ординат) на некотором диапазон значений параметра (ось абсцисс). Но некоторые модели могут обладать достаточно большим количеством параметров, и в данном случае получается многомерное пространство поиска, которое сложно визуализировать.

В таком случае, можно визуализировать следующем путем. Допустим, что есть два набора параметров  и . Обычно  случайно выбранный набор, а  уже найденный набор параметров характеризующий локальный минимум (в некоторых случаях даже глобальный). Имея таких два набора, можно провести линейную интерполяцию между этими наборами:



Таким образом, изменяя значение от 0 до 1, можно получать новые наборы параметров, а затем построить график зависимости функции потерь от  (рисунок 5.). Как видно, при  будет получаться набор  , а при  набор .

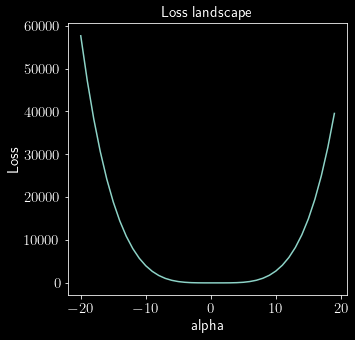


Рисунок 5. Пример графика зависимости потерь от параметра 

Другой подход, это визуализация в виде карты высот. В данном случае изначально необходим один набор известных параметров , а также вектора направления  и , размерность которых равна размерности вектора параметров. В таком случае, можно получать новые наборы параметров изменяя два коэффициента  и  согласно формуле:



Пример данной визуализации можно увидеть на рисунке 6.

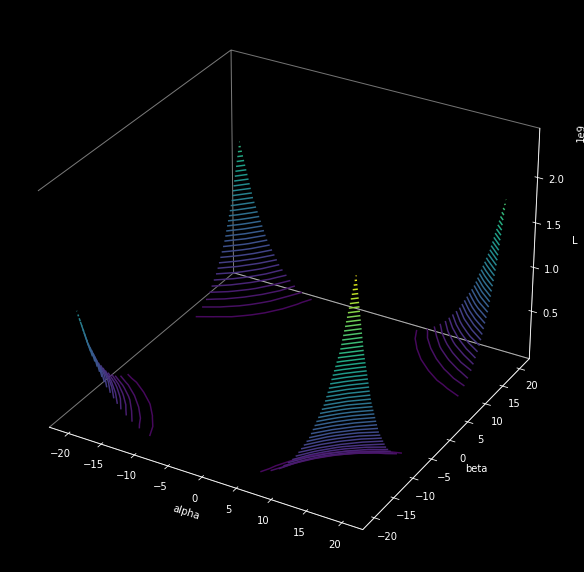


Рисунок 6. Пример карты высот потерь в зависимости параметров  и , изменяющихся в диапазоне [-20;20]

## Оптимизация

Оптимизация – это процесс максимизации «выгодных» характеристик или минимизации «не выгодных». В случае моделей машинного обучения, оптимизация — это минимизация значения функции потерь за счет изменения варьирующихся параметров.

### Случайный поиск

Алгоритм случайного поиска является самым примитивным способом оптимизации. Алгоритмы выглядит следующим образом:

1. Выбирается начальная точка поиска  в которой считается значение целевой функции. Данная точка является текущей наилучшей.
2. Изменяем текущую точку на случайный вектор  и вычисляется значение целевой функции в полученной точке.
3. Если значение целевой функции улучшилось, то данная точка становится текущей наилучшей.
4. Проверяется условие остановки поиска. Если оно выполняется, то оптимальная точка найдена, иначе происходит переход на шаг 2.

Данный алгоритм можно отобразить в виде рисунка 7.

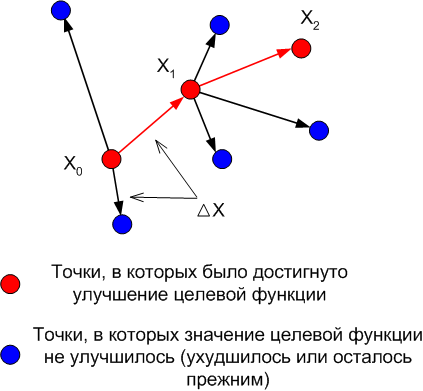


Рисунок 7. Пример случайного поиска

Гораздо более эффективными и хорошо зарекомендовавшими себя практике являются адаптивные алгоритмы случайного поиска глобального экстремума. Их основная идея заключается в том, что поиск ведется не из какой-то одной начальной точки, а по всей области, и в процессе его выполнения изменяется закон распределения генерации вектора рабочих параметров (точек, в которых вычисляется значений целевой функции). Обычно на начальных этапах распределение является равномерным, а затем плотность вероятности увеличивается в районе предполагаемого оптимума. Следует заметить, что многие из этих алгоритмов хорошо зарекомендовали себя при решении задач как непрерывной, так и дискретной и дискретно-непрерывной оптимизации, а, следовательно, может использоваться при параметрическом, структурном и структурно-параметрическом синтезе объектов.

В случае, когда целевая функция представлена в таком виде, что она дифференцируема в области поиска, то случайный поиск преобразуется к такому виде, что вместо изменения текущей точки на случайный вектор, считается значение градиента в текущей точки, и следующая точка выбирается вдоль направления антиградиента.

### Метод градиентного спуска

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Раздел 4.3 стр. 84

# Введение в нейронные сети, метод обратного распространения ошибки

## Простая нейронная сеть

Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. – " Издательский дом"" Питер""", 2017. Раздел 3.2 стр. 97

## Обратное распространение ошибки

Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. – " Издательский дом"" Питер""", 2017. Раздел 2.5 стр. 75

# Работа нейронных сетей: процесс кросс­валидации, оптимизации, поиска ошибок

## Настройка данных и модели

Все современные системы машинного обучения используют тензоры в качестве основной структуры данных. Тензоры являются фундаментальной структурой данных — настолько фундаментальной, что это отразилось на названии библиотеки Google TensorFlow.

Фактически тензор — это контейнер для данных, практически всегда числовых. Другими словами, это контейнер для чисел. Например, матрица является двумерным тензором: тензоры — это обобщение матриц с произвольным количеством измерений.

Тензор, содержащий единственное число, называется скаляром (скалярным, или тензором нулевого ранга).

Одномерный массив чисел называют вектором, или тензором первого ранга. Тензор первого ранга имеет единственную ось. Например, пятимерный вектор имеет только одну ось и пять значений на этой оси, тогда как пятимерный тензор имеет пять осей (и может иметь любое количество значений на каждой из них).

Массив векторов — это матрица, или двумерный тензор. Матрица имеет две оси (часто их называют строками и столбцами).

Если упаковать такие матрицы в новый массив, получится трехмерный тензор, который можно представить как числовой куб. Упаковав трехмерные тензоры в массив, вы получите четырехмерный тензор и т. д. В глубоком обучении чаще всего используются тензоры от нулевого ранга до четырехмерных, но иногда, например при обработке видеоданных, дело может дойти и до пятимерных тензоров.

При разработке модели машинного обучения необходимо обеспечить, чтобы модель могла на вход принимать тензор нужного ранга и формы, и выдаваемый результат также представлялся в виде необходимого тензора (например, в случае бинарное классификации это может быть один скаляр, а в случае нескольких классов вектор, размер которого соответствует количеству классов).

При подготовке данных, обычно выполняют следующий набор действий:

* данные должны быть представлены в виде тензора
* значения, помещаемые в тензоры, обычно требуют масштабирования и приведения к меньшим величинам: например, в диапазоне [–1, 1] или [0, 1]
* если значения разных признаков находятся в разных диапазонах (разнородные данные), их следует нормализовать
* возможно, понадобится выполнить конструирование признаков, особенно при небольшом объеме исходных данных. То есть из исходных признаков сформировать новые в пространстве меньшей размерности. Например, можно использовать метод главных компонент.

## Веса при инициализации

Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. – " Издательский дом"" Питер""", 2017. Раздел 4.2 стр. 142

## Пакетная нормализация

Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. – " Издательский дом"" Питер""", 2017. Раздел 4.3 стр. 153

## Регуляризация (L2 / L1 / Maxnorm /Dropout)

Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. – " Издательский дом"" Питер""", 2017. Раздел 4.1 стр. 138

## Проверка вменяемости (кросс-валидация)

### выделение из общей выборки отдельного проверочного набора данных

Смысл состоит в следующем. Некоторая часть данных выделяется в контрольный набор. Обучение производится на оставшихся данных, а оценка качества — на контрольных. Как уже говорилось, для предотвращения утечек информации модель не должна настраиваться по результатам прогнозирования на контрольных данных, поэтому требуется также зарезервировать отдельный проверочный набор. Это самый простой протокол оценки, страдающий одним существенным недостатком: при небольшом объеме доступных данных проверочный и контрольный наборы могут содержать слишком мало образцов, чтобы считаться статистически репрезентативными.

### Перекрестная проверка по K блокам

При использовании этого подхода данные разбиваются на K блоков равного размера. Для каждого блока i производится обучение модели на остальных K—1 блоках и оценка на блоке i. Окончательная оценка рассчитывается как среднее K промежуточных оценок. Этот метод может пригодиться, когда качество модели слишком сильно зависит от деления данных на тренировочный/контрольный наборы. Подобно проверке с простым расщеплением выборки, этот метод не избавляет от необходимости использовать отдельный проверочный набор для калибровки модели.

### Итерационная проверка по K блокам с перемешивание

Этот метод подходит для ситуаций, когда имеется относительно небольшой набор данных и требуется оценить модель максимально точно. Суть его заключается в многократном применении перекрестной проверки по K блокам с перемешиванием данных перед каждым разделением на K блоков. Конечная оценка — среднее по оценкам, полученным в прогонах перекрестной проверки по K блокам. Обратите внимание: в конечном счете обучению и оценке подвергается P × K моделей (где P — количество итераций), что может быть очень затратным.

## Процесс обучения

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Раздел 6.5.6 стр. 188

## Прямой порядок (SGD), импульс, нестеровский импульс

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Раздел 8.3 стр. 253

## Методы второго порядка

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Раздел 8.6 стр. 267

## Адаптивные скорости обучения по параметрам (Adagrad, RMSProp).

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Раздел 8.5 стр. 263

# Сверточные нейронные сети: архитектура, сверточные / объединяющие слои

## Сверточные сети

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Глава 9 стр. 282

## Архитектуры сверточных сетей

Существует несколько архитектур свёрточных нейронных сетей у которых есть собственные имена. Вот некоторые из них:

* **LeNet**. Первая успешная реализация свёрточной нейронной сети была разработана Yann LeCun в 1990. Из них самой известной стала архитектура LeNet, которая использовалась для чтения ZIP-кодов, цифры и др.
* **AlexNet**. Первая работа, которая популяризовала свёрточные нейронные сети в компьютерном зрении, разработана Alex Krizhevsky, Ilya Sutskever и Geoff Hinton. AlexNet была протестирована на соревновании ImageNet ILSVRC в 2012 году и значительно опередила конкурента на втором месте (процент ошибок: 16% против 26%). Архитектура сети была очень похожа на архитектуру LeNet, но была глубже, больше и использовала последовательности свёрточных слоёв (до этого было стандартной практикой использоватье только комбинацию свёрточного слоя сразу за которым следовал слой подвыборки).
* **ZFNet**. Победителем ILSVRC 2013 года стала свёрточная нейронная сеть от Matthew Zeiler и Rob Fergus. Стала она известна под названием ZFNet. Это была усовершенствованная версия AlexNet, которая использовала другие значения гипер-параметров, в частности увеличивавшая размер среднего свёрточного слоя и уменьшавшая шаг и размер фильтра на первом свёрточном слое.
* **GoogLeNet**. Победителем ILSVRC 2014 года стала свёрточная нейронная сеть от Szegedy et al. из Google. Основным изменением в архитектура стал Inception-модуль, который позволял значительно снизить количество параметров в сети (4 Мб против 60 Мб в AlexNet). В добавок к этому сеть использует подвыборку по среднему значению вместо полносвязных слоёв к концу сети, исключая тем самым большое количество параметров, которые не имеют большого значения. У данной версии сети есть несколько модификаций, одна из последних — Inveption-v4.
* **VGGNet**. Вслед за победителем 2014 ILSVRC была сеть от Karen Simonyan и Andrew Zisserman, которая стала так же известна как VGGNet. Основным вкладом этой нейронной сети в развития представлений о свёрточных нейронных сетях был тот факт, что глубина сети является ключевым компонентом хорошей производительности. Финальная версия их сети содержала 16 пар свёрточныйх + полносвязных слоёв и использовала единые размеры фильтров (3х3 для свёртки и 2х2 для операции подвыборки). Их предобученная модель находится в публичном доступе и с ней можно поэкспериментировать. Обратная сторона медали VGGNet — стоимость вычислений и использование большого количества памяти (140Мб). Большинство из этих параметров находятся на первом полносвязном слое и, как было в дальнейшем изучено и протестировано, эти полносвязные слои могут быть исключены без влияния на производительность, но с колоссальным уменьшением количества требуемых параметров.

# Понимание и визуализация сверточных нейронных сетей

## Визуализация нейронных сетей

<https://medium.com/@balovbohdan/%D0%B3%D0%BB%D1%83%D0%B1%D0%BE%D0%BA%D0%BE%D0%B5-%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5-%D1%80%D0%B0%D0%B7%D0%B1%D0%B8%D1%80%D0%B0%D0%B5%D0%BC%D1%81%D1%8F-%D1%81%D0%BE-%D1%81%D0%B2%D0%B5%D1%80%D1%82%D0%BA%D0%B0%D0%BC%D0%B8-6e47bfc27792>

## Обучение через перемещение.

https://neurohive.io/ru/novosti/kak-transfer-learning-ispolzujut-dlya-zadach-s-medicinskimi-snimkami/

## Тренировка сверточных сетей на практике

https://habr.com/ru/post/456740/

# Классификации изображений: локализация, детектирование, сегментация

## Рекурсивные нейронные сети

Гудфеллоу Я., Иошуа Б., Курвилль А. Глубокое обучение. – Litres, 2018. Раздел 10.6 стр. 337

## Сегментация изображений

Chen T. et al. Learning to segment object candidates via recursive neural networks //IEEE Transactions on Image Processing. – 2018. – Т. 27. – №. 12. – С. 5827-5839.

Socher R. et al. Parsing natural scenes and natural language with recursive neural networks //ICML. – 2011.

## Механизм внимания

<https://sysblok.ru/knowhow/vnimanie-vse-chto-vam-nuzhno-kak-rabotaet-attention-v-nejrosetjah/>