УДК 004.896

СВЁРТОЧНЫЕ НЕЙРОННЫЕ СЕТИ В ЗАДАЧАХ ГЛУБОКОГО ОБУЧЕНИЯ

Каунг Мьят Сан, Абрамов Ю.А., Гинзгеймер С.А.
Калужский филиал федерального государственного бюджетного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», г. Калуга (248000, г. Калуга, ул. Баженова, д. 2.), e-mail: kominhtet94@gmail.com

Глубокие модели оказались ключом, который подходит ко всем замкам сразу: новые архитектуры и алгоритмы обучения, а также увеличившиеся мощности и появившиеся огромные наборы данных, привели к революционным прорывам в компьютерном зрении, распознавании речи, обработке естественного языка и многих других типично "человеческих" задачах машинного обучения. Глубокое обучение – часть машинного обучения. Доказано, что это эффективный метод поиска паттернов в сырых данных, таких как звук или изображение. Обучение называется "глубоким", поскольку в течение времени нейронная сеть охватывает всё большее число уровней, и чем "глубже" проникает сеть, тем выше её производительность. Несмотря на то, что в настоящее время большая часть глубокого обучения осуществляется под контролем человека, цель учёных – создать нейронные сети, способные формироваться и "обучаться" самостоятельно и независимо. В данной статье мы представляем обзор визуального представления функций глубокого обучения. В отличие от стандартных подходов к проблемам компьютерного зрения, основанных на особенностях ручной работы, методы глубокого обучения направлены на изучение особенностей, необходимых для выполнения поставленной задачи. Эта статья начинается с базового описания наиболее распространенной модели искусственной нейронной сети (ИНС) - многослойного персептрона (МП). Затем мы вводим модели свёрточной нейронной сети (СНС), специализированной архитектуры ИНС, особенно полезной для решения проблем компьютерного зрения. Затем мы кратко опишем, как достигается обучение в таких моделях. В заключении мы расскажем о методах регуляризации: ранней остановке, распаде веса, отсеве, пакетной нормализации.

Ключевые слова: глубокое обучение, нейронные сети, регуляризация

CONVOLUTIONAL NEURAL NETWORKS IN DEEP LEARNING TASKS

Kaung Myat San, Abramov Y.A., Ginzgamer S.A.
Bauman Moscow State Technical University (Kaluga Branch), Kaluga (248000, Kaluga, Bazhenova st, 2.), e-mail: kominhtet94@gmail.com

Deep models turned out to be the key which suits to all locks at once: new architectures and learning algorithms, and also increased power and appeared huge data sets, led to revolutional breachs in computer vision, speech recognition, natural language processing and many other typically "human" machine learning tasks. Deep learning is a part of machine learning. It's proved that it's an effective method for pattern searching in raw data, such as sound or image. Learning is called "deep" since during time neural network covers more and more level number, and the "deeper" network gets, the higher its performance. Despite the fact that today the most part of deep learning is under human control, the target of scientists is to create neural networks which are able to be formed and "learnt" independent. We present a review of deep learning functions visual presentation in this article. Unlike standard approaches to computer vision problems based on manual work features, deep learning methods are directed to study features needed for completing objective. This article begins with basic description of most popular model of artificial neural network – multilayer perceptron. Then we introduce convolutional neural network model, special architecture ANN, which is specially useful for computer vision problems decision. Then we briefly describe how to reach learning in these models. In conclusion we will tell about regularization methods: early stop, weight decay, elimination and package normalization.

Key words: deep learning, neural networks, regularization

Многослойный Персептрон

Первой моделью глубокого обучения был многослойный персептрон. Данная модель была первоначально вдохновлена нейронными системами. МП - это тип

моделей машинного обучения, которые применяют последовательность нелинейных преобразований к входным данным.

Математически МП с L слоями можно описать следующими уравнениями:

$$\begin{aligned} y = f(x,\theta) &= h_L \\ h_l &= \sigma_l(W_l h_{l-1} + b_l), \forall l \in \{1,2,.....,\mathrm{L}\} \\ \mathrm{h}_0 &= x, \end{aligned} \tag{1}$$

где $\theta = \{W_1, b_l\}, \forall l \in \{1,.....,\mathrm{L}\}$ - множество обучаемых параметров для каждого слоя $l, x \in R^{d_{in}}$ - это входной вектор, $y \in R^{d_{out}}$ является выходом в сеть и σ_l - нелинейные функции активации в слое l.

Общие нелинейные функции активации для скрытых единиц измерения $(\sigma_l, l \in \{1,.....,L-1\})$.

гиперболический тангенс

$$Tanh(\mathrm{x}) = \frac{e^{2x-1}}{e^{2x+1}} \tag{2}$$

и выпрямленный линейный блок (ВЛб)

$$ВЛб(\mathrm{x}) = \max(0, \mathrm{x}) \tag{3}$$

Обратите внимание на важность функции активации: без неё вся система была бы стеком линейных операций и могла бы быть эквивалентно записана как одна матрица. Выбор функции активации является целиком эмпирическим вопросом. Если сеть очень глубокая, активации ReLU вероятны, поскольку они уменьшают вероятность исчезновения градиента.

Сверточные Нейронные Сети

Сверточные нейронные сети (СНС) являются естественным продолжением МП для обработки данных, которое имеет известную сетчатую топологию. [3] СНС используют пространственную корреляцию сигнала, чтобы ограничить архитектуру более разумным способом. Их архитектура, несколько вдохновленная биологической визуальной системой, обладает двумя ключевыми свойствами, которые делают их чрезвычайно полезными для приложений изображений: пространственно разделяемые веса и пространственное объединение. Такого рода сети изучают особенности, инвариантные к сдвигу, т. е. фильтры, которые полезны для всего изображения. Слои объединения отвечают за снижение чувствительности выходных данных к незначительному сдвигу входных данных и искажениям.

Типичная сверточная сеть состоит из нескольких этапов. Выходные данные каждого этапа состоят из набора двумерных массивов, называемых картами объектов. Каждая карта объектов является результатом одного сверточного фильтра, примененного к полному изображению. Точечная нелинейная функция активации всегда следует за сверточным слоем.

В более общем виде сверточная сеть может быть записана как:

$$\begin{aligned} Y = f(\mathrm{X},\theta) &= H_L \\ H_l &= \sigma_l(W_l H_{l-1} + b_l), \forall l \in \{1,2,.....,\mathrm{L}\} \\ H_0 &= X, \end{aligned} \tag{4}$$

где $\theta = \{W_l, b_l\}, \forall l \in \{1, ..., L\}$ - множество параметров, поддающихся обучению, как и в $MLP, X \in R^{c*h*w}$ - входное изображение (с цветовыми каналами, высотой H пикселей и шириной W пикселей), $Y \in R^{n*h'*w'}$ (с размером $h'*w'$) выходных векторов размерности n (каждый вектор представляет собой нелинейное кодирование окна ввода), σ_l является нелинейностью в слое l и $pool_l$ - это функция (опциональная) объединения в слое l .

Основное различие между МП и СНС заключается в матрицах параметров W_l : в МП матрицы могут принимать любую общую форму, в то время как в СНС эти матрицы являются ограничениями для матриц [6]. Каждый узел массива H_l может быть выражен в качестве дискретной свертки времени между ядрами из W_l и предыдущей скрытой единицей H_{l-1} (преобразованной через точечную нелинейность и, возможно, объединенной). Точнее,

$$H_{lp} = pool_l(\sigma_l(b_{lp} + \sum_{q \in parents(p)} w_{lq} * H_{l-1,q})), \quad (5)$$

где H_{lp} это p^{th} компонент l^{th} карты объектов H_l.

Из приведенного выше математического описания базовая сверточная нейронная сеть может рассматриваться как стек из трех различных компонентов (см. рисунок 1): [4]

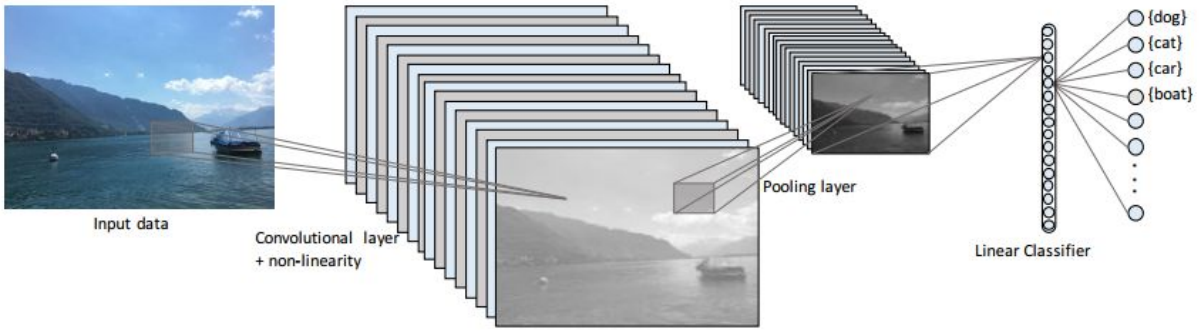

Рисунок 1 – Архитектура типичной сверточной сети для распознавания объектов. Мы представляем один простой блок (состоящий из сверточного, нелинейного и объединяющего слоя) и конечный линейный многоклассовый классификатор.

Сверточный Слой. Входной сигнал каждого слоя – трёхмерный массив с картами особенностей c_i двумерного размера $h_i * w_i$. Каждый компонент обозначается x_{ijk}, а карта объектов - x_i. Выход также представляет собой 3D-массив y, соответствующий картам объектов размера $h_0 * w_0$. Обучаемый фильтр w_{ij} (и параметр смещения b j) является обучаемым ядром размера $k_1 * k_2$ и соединяет входную карту объектов xi с выходной картой объектов y_j. Сверточный модуль вычисляется следующим образом:

$$y_j = b_j + \sum_i w_{ij} * x_i \quad (6)$$

где * является двумерным дискретным оператором свертки. Каждый фильтр w_{ij} изучает определенный объект в каждом местоположении на входе (следовательно,

имеется пространственная инвариантность). Эта свертка может иметь шаг больше 1. В недавней реализации сверточного слоя стало популярным заполнять входной слой так, чтобы выходные данные сверточного слоя имели такое же измерение, как и входные данные.

Уровень Активации. Подобно случаю МП, точечная нелинейность применяется к каждому местоположению (ijk) векторных карт.

Слой Объединения. Этот слой отвечает за уменьшение пространственного размера каждой карты объектов своего входного сигнала. Это свойство важно в моделях изображений по трем причинам: (i) модель устойчива к небольшим изменениям в расположении объектов в предыдущих слоях, (ii) увеличивается восприимчивое поле сети и (iii) оно контролирует емкость модели. Операция объединения может быть применена (опционально) после каждого уровня активации. Мы всегда рассматриваем слои Max pooling, которые состоят из отчетов о максимальном выходе в пределах прямоугольной окрестности. Другие популярные функции объединения включают в себя прямоугольную окрестность, норму L^2 прямоугольной окрестности или средневзвешенное значение на основе расстояния от центрального пикселя. Нужно выбирать шаг таким, чтобы слой объединения равнялся размеру ядра (например, $2*2$ слой объединения принимает максимальное значение в каждом $2 * 2$ окне и смещён на 2, следовательно, происходит уменьшение пространственной размерности функции карты в 2 раза).

Обучение

В этой статье мы кратко объясним, как оценка параметров (обучение) осуществляется в контексте нейронных сетей (тот же принцип применяется к МП, СНС или любому другому типу архитектуры). Различные проблемы сегментации определяются как дискриминационные задачи. Поэтому мы будем рассматривать только обучения для задач, в которых сеть предназначена для моделирования условных вероятностей.

Функция потерь

Как только модель определена, ее структура может быть абстрагирована и сеть может рассматриваться как аппроксиматор функций. В задачах классификации нейронные сети моделируются таким образом, что их выход можно рассматривать как условное распределение цели y, учитывая входные x и параметры θ. Эта вероятность может быть смоделирована путем преобразования выходных данных сети $f_c(x,\theta)$ (для каждого класса $c \in \{1,.....,C\}$) функцией softmax:

$$p(y^i \mid x^i,\theta) = \prod_{c=1}^{C} \left(\frac{e^{f_c(x^i,\theta)}}{\sum_j f_j(x^i,\theta)} \right)^{1_c(y^i)} \tag{7}$$

где $1_a(x)$ - функция индикатора

$$1_a(x) \begin{cases} 1, if \;\; x = a \\ 0, otherwise \end{cases}$$

В особом случае, когда $C = 2$, функция softmax сводится к логистической регрессии. Поэтому softmax можно рассматривать как обобщение многоклассовой логистической регрессии.

Ниже мы рассмотрим набор данных $D = \{X, Y\} = \{x^i, \mathrm{y}^i\}, i \in \{1, ..., N\}$, где x^i-входное изображение, а y^i-связанная с ним цель (метка). Мы вычисляем мультикатегориальные проблемы, в которых каждая надпись y^i принадлежит к одной из категорий C, $y^i \in \{1, ..., \mathrm{C}\}$. Параметры модели найдены путем максимизации вероятности по данным обучения D относительно параметров θ :

$$\begin{aligned} \theta^* &= \arg\max\ \mathrm{p}(Y \mid X, \theta) \\ &= \arg\max\ \mathrm{p}(\mathrm{y}^1, \mathrm{y}^2, ..., y^N \mid x^1, x^2, ..., \mathrm{x}^N, \theta) \\ &= \arg\max \prod_{i=2}^{N} p(y^i, x^i, \theta) \end{aligned} \tag{8}$$

где последняя строка предполагает, что обучающий набор выбирается из неизвестного независимого, одинаково распределенного (HOP) распространения. Эквивалентно, эту задачу оптимизации можно рассматривать как минимизацию отрицательной логарифмической вероятности обучающих данных (поскольку функция логарифма монотонна):

$$\theta^* = \arg\min_{\theta} - \sum_i \sum_c 1_c(y^i)[\mathrm{f}_c(x^i, \theta) - \log(\sum_j e^{f_j(x^i, \theta)})]. \tag{9}$$

Обратите внимание, что эта проблема минимизации эквивалентна кросс-энтропии реального распределения цели и распределения, генерируемого нейронной сетью. В задачах классификации принято рассматривать однострунное кодирование в целевом распределении. [2] В этом случае проблему минимизации можно переписать следующим образом:

$$\theta^* = \arg\min_{\theta} - \sum_i [\mathrm{f}_{c_i^*}(x^i, \theta) - \log(\sum_j e^{f_j(x^i, \theta)})], \tag{10}$$

где c_i^* - кодирование одной горячей метки примера обучения i. Эта проблема не имеет замкнутого решения, и мы применяем итеративный подход для нахождения минимума. То есть параметры θ сети оцениваются путем минимизации следующей функции потерь (также известной как функция стоимости, целевая функция или критерий):

$$L(\theta, X, Y) = -\sum_i \left[f_{c_i^*}(x^i, \theta) - \log(\sum_j e^{f_j(x^i, \theta)}) \right], \tag{11}$$

Наиболее распространенным методом оптимизации, используемым при машинной обработке, является метод градиентного спуска, который со случайным образом инициализированным набором параметров θ_1 определяется как:

$$\theta_{t+1} \longleftarrow\!\!\!- \theta_t - \eta \Delta_\theta L(\theta, X, Y) \tag{12}$$

Этот метод также известен как метод спуска градиента. В крупных проблемах машинного обучения (например, проблемы сегментации), этот метод может стать практически неэффективным, так как он требует наличия всех данных на каждой

итерации. Более того, спуск градиента может быть избыточным, если набор данных содержит аналогичные примеры.

Общим способом решения этой проблемы является рассмотрение стохастического приближения градиента, обычно называемого стохастическим градиентным спуском (СГС). В этом случае для оценки градиента используется случайная обучающая выборка $\{x^i, y^i\}$ и параметры обновляются следующим образом:

$$\theta_{t+1} \longleftarrow\!\!\!- \theta_t - \eta \Delta_\theta L(\theta, \mathrm{x}^i, \mathrm{y}^i) \tag{13}$$

Чаще всего мы оптимизируем функцию потерь, используя промежуточный между стохастическим и пакетным метод, называемый мини-пакетным градиентным спуском. Этот подход, который использует подмножество $n < |D|$ обучающих выборок для выполнения каждого обновления параметров, определяется как:

$$\theta_{t+1} \longleftarrow\!\!\!- \theta_t - \eta \Delta_\theta L(\theta, \mathrm{x}^{(i,\ldots,\mathrm{i+n})}, \mathrm{y}^{(i,\ldots,i+n)}). \tag{14}$$

Существует много различных вариантов этого правила обучения, используемого для уменьшения шума в стохастических направлениях, например, метод импульса, среднего стохастического градиентного спуска, Adagrad [5], Rmsprop и Адама [6]. Другим важным классом алгоритмов являются методы второго порядка, которые используют информацию второй производной функции потерь для улучшения оптимизации. Однако эти методы невозможно вычислить в крупномасштабных задачах, рассматриваемых в статье.

Регуляризация

Нейронные сети содержат большое количество свободных параметров, которые необходимо изучить. Эти модели могут описывать огромный диапазон явлений, но требуют много данных, чтобы избежать переобучения, то есть, когда модель способна достичь хорошей производительности на тренировочных данных, но плохо работает на тестовых данных.

Простейший метод регуляризации называется ранней остановкой. Он состоит, как следует из названия, в том, чтобы остановить обучение, как только ошибка проверки достигает минимальной ошибки.

Распад веса является еще одним очень распространенным методом регуляризации. Он используется для ограничения значений больших весов. Эти методы обычно реализуются путем добавления дополнительных условий для функций сети. В регуляризации L^2 к функции стоимости добавляется дополнительный член, который штрафует квадратную величину всех параметров. То есть, для каждого веса w_i в сети, мы добавляем термин $\frac{1}{2}\lambda w_i^2$ к функции потери, где присутствует сила регуляризации. Регуляризация L^2 имеет интуитивную интерпретацию сильно штрафующих векторов острого веса и предпочитающих диффузные векторы веса. Другой общей регуляризацией распада веса является регуляризация L^1, которая создает разреженные ограничения на вес. Подобно L^2 , регуляризация L^1 включает в себя дополнительный термин к стоимости, функцию $\lambda |w_i|$. Нейроны с L^1 регуляризацией в конечном итоге используют только разреженное подмножество их наиболее важных входов и становятся почти инвариантными к “шумным” входам.

Чрезвычайно эффективным методом регуляризации нейронных сетей является отсев [7]. Во время тренировки отсев осуществляется только путем поддержания активности нейрона с определенной вероятностью p или его обнулением в противном случае. Его можно интерпретировать как выборку в пределах полной нейронной сети и обновление параметров выбранной сети на основе входных данных. На время теста, мы в идеале хотели бы найти пример среднего значения всех возможных 2^n выпадающих сетей. К сожалению, это невозможно для больших значений n. Тем не менее, мы можем найти аппроксимацию, используя полную сеть с выходом каждого узла, взвешенным на коэффициент p, поэтому ожидаемое значение выхода любого узла такое же, как и на этапах обучения.

Пакетная нормализация [8] - еще один популярный метод регуляризации глубоких сверточных нейронных сетей. [1] Этот метод частично снимает проблему внутреннего ковариантного сдвига в глубоких сетях, то есть обучение глубоких сетей затруднено из-за изменения распределения входных данных каждого слоя во время обучения, как и изменения параметров предыдущего слоя. В СГД обучении каждая активация мини-пакета центрируется в нулевую медиану и единичную дисперсию. Среднее значение и дисперсия измеряются по всей мини-партии, независимо для каждой активации. Пакетная нормализация позволяет нам использовать гораздо более высокие скорости обучения и быть менее осторожными при инициализации.

Вывод

В этой статье мы кратко представили различные аспекты проблемы сегментации изображений. Затем мы представили базовое введение в методы глубокого обучения, наиболее часто используемые для работы с проблемами компьютерного зрения.

Список литературы

1. Маркина Ю.Ю., Белов Ю.С. Кепстральные коэффициенты как необходимая характеристика процесса создания системы имитации голоса человека с помощью методов глубокого обучения // Международный студенческий научный вестник. 2018. № 1. С. 78.
2. Логинов Б.М., Коржавый А.П., Белов Ю.С., Либеров Р.В. Методика распознавания и классификации образов структур многокомпонентных материалов для электронных систем // Электромагнитные волны и электронные системы. 2017. Т. 22. № 2. С. 4-12.
3. Гришанов К.М., Белов Ю.С. Модель свёрточной нейронной сети в задачах машинного зрения // Электронный журнал: наука, техника и образование. 2017. № СВ1 (11). С. 100-106.
4. Нестеров А.Ю., Белов Ю.С. Распознавание образов по уникальным точкам на примере дорожных знаков // Электронный журнал: наука, техника и образование. 2016. № 4 (9). С. 113-119.
5. John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization // Journal of Machine Learning Research (JMLR). 2011.
6. Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization // International Conference on Learning Representations (ICLR). 2014.

7. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting // Journal of Machine Learning Research (JMLR). 2014.
8. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift // International Conference on Machine Learning (ICML) Deep Learning Workshop. 2015.