Регуляризация

Главная проблема машинного обучения состоит в том, что алгоритм должен хорошо работать на новых данных, которых он раньше не видел, а не только на тех, что использовались для обучения модели. Эта способность правильной работы на ранее не предъявлявшихся данных называется обобщением.

Обычно при обучении модели имеется доступ к обучающему набору. Возможно вычислить некоторую меру ошибки на обучающем наборе (ошибка обучения), и постараться минимизировать ее. Однако, требуется уменьшить не только ошибку обучения, но и ошибку обобщения (ошибку тестирования).

Ошибка обобщения – это математическое ожидание ошибки на новых входных данных. Здесь математическое ожидание вычисляется по возможным входным данным, выбираемым из распределения, которое, как мы предполагаем, может встретиться на практике.

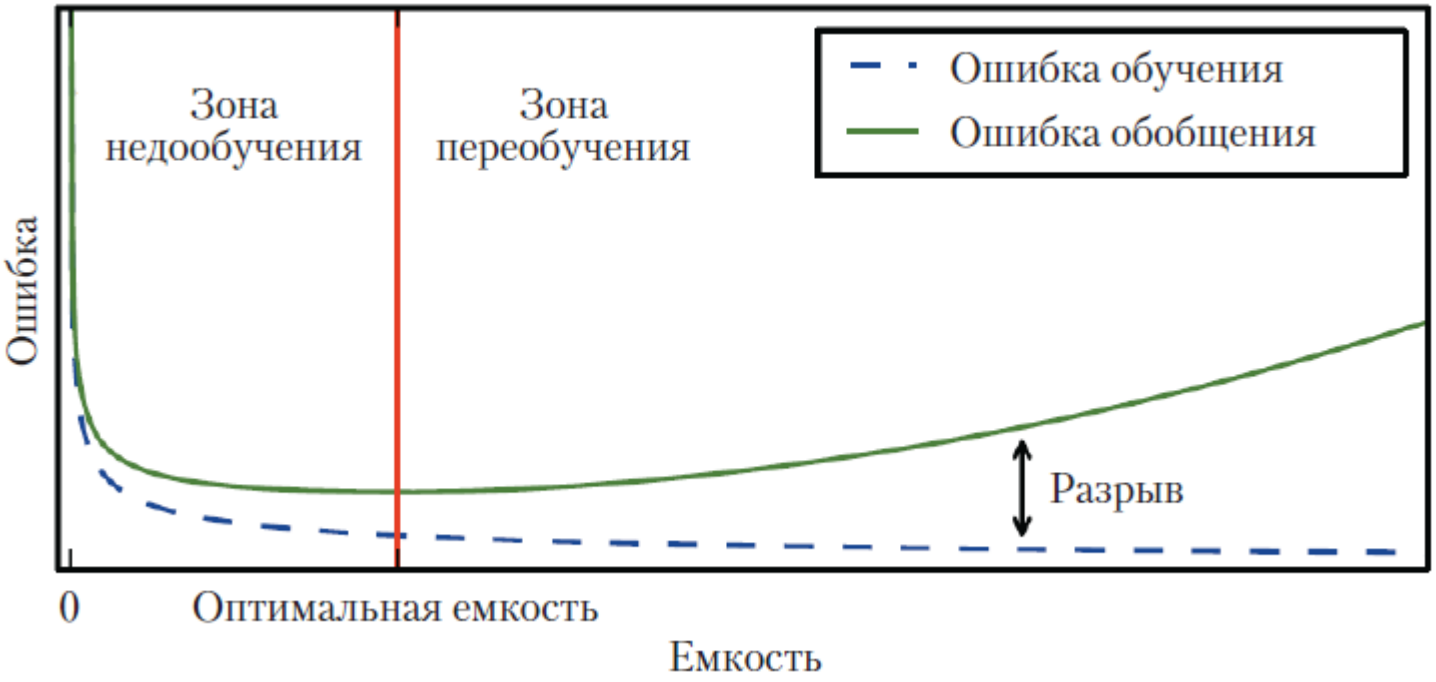
Регуляризация

Для повышения качества работы алгоритма обучения необходимо:
1) сделать ошибку обучения как можно меньше;
2) сократить разрыв между ошибками обучения и тестирования.

При решении этих задач возникают две центральные проблемы машинного обучения: недообучение и переобучение. Недообучение имеет место, когда модель не позволяет получить достаточно малую ошибку на обучающем наборе, а переобучение – когда разрыв между ошибками обучения и тестирования слишком велик.

Управлять склонностью модели к переобучению или недообучению позволяет ее емкость. Емкость модели описывает ее способность к аппроксимации широкого спектра функций. Модели малой емкости испытывают сложности в аппроксимации обучающего набора. Модели большой емкости склонны к переобучению, поскольку запоминают свойства обучающего набора, не присущие тестовому.

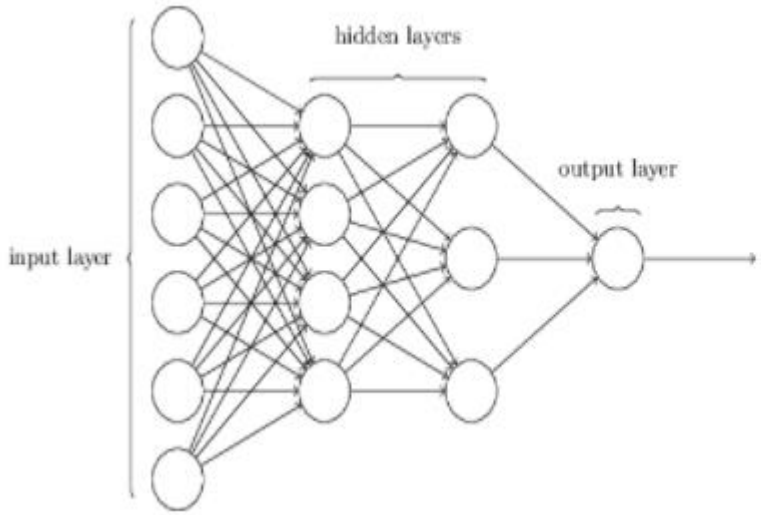
Регуляризация. Переобучение

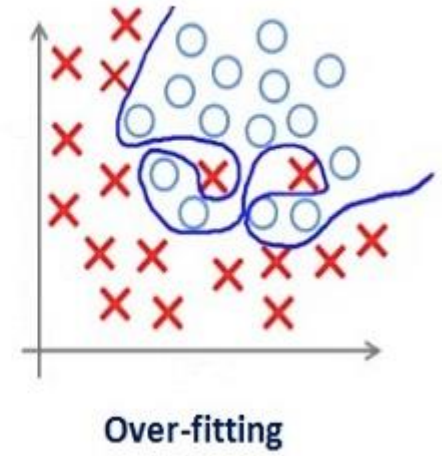


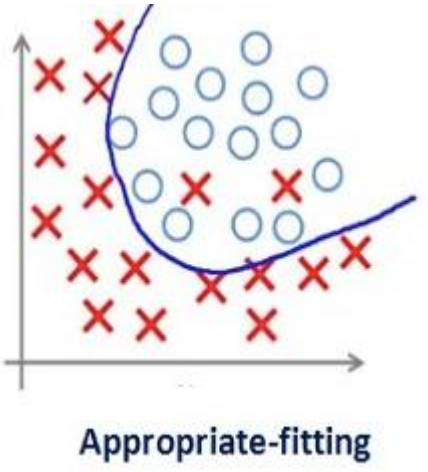
Как показано на рисунке с первых шагов обучения происходит уменьшение ошибки. На последующих шагах с целью уменьшения ошибки (целевой функции) параметры подстраиваются под особенности обучающего множества. Однако при этом происходит "подстройка" не под общие закономерности ряда, а под особенности его части - обучающего подмножества. При этом точность прогноза уменьшается.

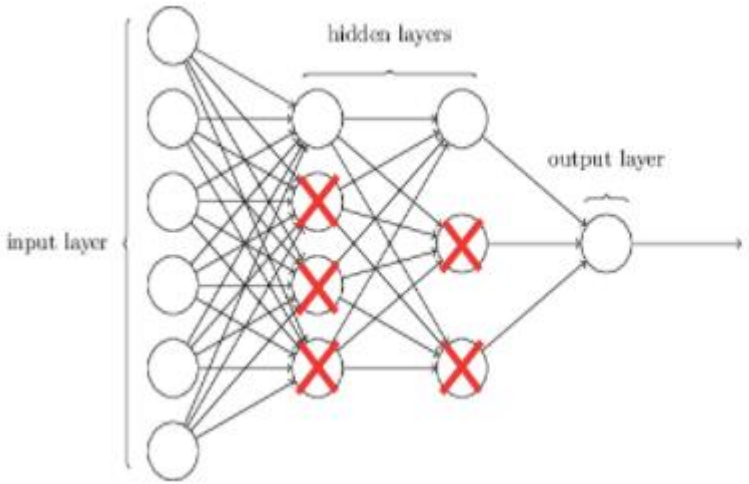
Регуляризация – это любая модификация алгоритма обучения, предпринятая с целью уменьшить его ошибку обобщения, не уменьшая ошибку обучения.

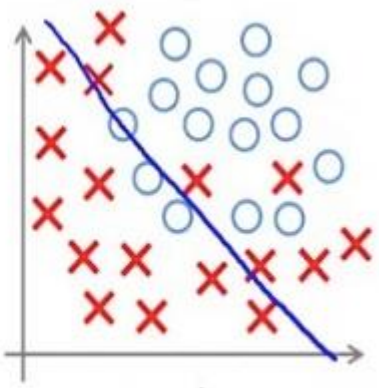
Регуляризация
Примеры переобучения и недообучения нейронной сети











Регуляризация
L2 & L1 регуляризация

Многие подходы к регуляризации основаны на прибавлении штрафа по норме параметра $\Omega(\theta)$ к целевой функции J. Мы будем обозначать регуляризованную целевую функцию как $\tilde{J}$:

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(\theta),$$

где $\alpha \in [0, \infty)$ – гиперпараметр, задающий вес члена Ω, штрафующего по норме относительно стандартной целевой функции J. Если α равен 0, то регуляризация отсутствует. Чем больше значение α, тем сильнее регуляризация.

Минимизируя регуляризованную целевую функцию $\tilde{J}$, алгоритм обучения одновременно уменьшает исходную целевую функцию J на обучающих данных и некоторую меру величины параметров θ (или какого-то подмножества параметров).

Обычно штраф выполняется по норме веса аффинного преобразования в каждом слое, оставляя смещения нерегуляризированными. Это определяется тем, что для точного подбора смещений, как правило, требуется меньше данных, чем для весов.

Регуляризация

L^2 регуляризация

Штраф параметров по норме L^2 часто называют снижением весов. Цель такой стратегии регуляризации – выбирать веса близкие к началу координат за счет прибавления к целевой функции члена регуляризации $\Omega(\theta) = 1/2 \|w\|^2$. Регуляризацию по норме L^2 называют также гребневой регрессией или регуляризацией Тихонова.

Рассмотрим градиент регуляризированной целевой функции. Для простоты опустим параметр смещения, т. е. будем считать, что θ совпадает с w. Тогда полная целевая функция имеет вид:

$$\tilde{J}(w; X, y) = (\alpha/2) w^\top w + J(w; X, y),$$

а градиент по параметрам:

$$\nabla_w \tilde{J}(w; X, y) = \alpha w + \nabla_w J(w; X, y).$$

Один шаг обновления весов с целью уменьшения градиента имеет вид:

$$w \leftarrow w - \varepsilon(\alpha w + \nabla_w J(w; X, y)).$$

Регуляризация
L^2 регуляризация

То же самое можно переписать в виде:

$$w \leftarrow (1 - \varepsilon\alpha)w - \varepsilon\nabla_w J(w; X, y))$$

Как видим, добавление члена сложения весов изменило правило обучения: теперь мы на каждом шаге умножаем вектор весов на постоянный коэффициент, меньший 1, перед тем как выполнить стандартное обновление градиента. Таким образом, мы рассмотрели один шаг. Рассмотрим процесс обучения в целом.

Упростим анализ, предположив квадратичную аппроксимацию целевой функции в окрестности того значения весов, при котором достигается минимальная стоимость обучения без регуляризации, $w^* = \arg\min_w J(w)$. Если целевая функция действительно квадратичная, то такая аппроксимация идеальна. Аппроксимация J описывается формулой:

$$\hat{J}(\theta) = J(w^*) + {}^1/_2(w - w^*)^\top H(w - w^*)$$

где H – матрица Гессе J относительно w, вычисленная в точке w^*.

Регуляризация

L^2 регуляризация

В этой квадратичной аппроксимации нет члена первого порядка, потому что w*, по определению, точка минимума, в которой градиент обращается в нуль. Из того, что w* – точка минимума J, следует также, что матрица H положительно полуопределенная.

Минимум J достигается там, где градиент $\nabla_{\boldsymbol{w}} \hat{J}(\boldsymbol{w}) = \boldsymbol{H}(\boldsymbol{w} - \boldsymbol{w}^*)$ равен 0.

Чтобы изучить эффект снижения весов, модифицируем уравнение, прибавив градиент снижения весов. Теперь возможно найти из него минимум регуляризированного варианта $\hat{J}$. Обозначим $\tilde{\boldsymbol{w}}$ положение точки минимума.

Когда α стремится к 0, регуляризированное решение $\tilde{\boldsymbol{w}}$ стремится к w*. Рассмотрим случай возрастания α. Поскольку матрица H вещественная и симметричная, мы можем разложить ее в произведение диагональной матрицы Λ и ортогональной матрицы собственных векторов Q: $H = Q\Lambda Q^\top$.

Регуляризация

L^2 регуляризация

Подставляя это разложение, получаем

$$\begin{aligned}\tilde{\boldsymbol{w}} &= (\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top + \alpha\boldsymbol{I})^{-1}\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^*, \\ &= [\boldsymbol{Q}(\boldsymbol{\Lambda} + \alpha\boldsymbol{I})\boldsymbol{Q}^\top]^{-1}\boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^*, \\ &= \boldsymbol{Q}(\boldsymbol{\Lambda} + \alpha\boldsymbol{I})^{-1}\boldsymbol{\Lambda}\boldsymbol{Q}^\top\boldsymbol{w}^*.\end{aligned} \tag{1}$$

Мы видим, что результатом снижения весов является масштабирование w* вдоль направлений собственных векторов H. Точнее, компонента w*, параллельная i-му собственному вектору H, умножается на коэффициент $\lambda_i /(\lambda_i + \alpha)$.

Вдоль направлений, для которых собственные значения H относительно велики, например когда $\lambda_i \gg \alpha$, эффект регуляризации сравнительно мал. Те же компоненты, для которых $\lambda_i \ll \alpha$, сжимаются почти до нуля.

Относительно неизменными остаются только те направления, в которых параметры дают сильный вклад в уменьшение целевой функции. Если это не так, то собственное значение гессиана мало, т. е. движение в этом направлении не приводит к заметному возрастанию градиента.

Регуляризация
L^2 регуляризация

Компоненты вектора весов, соответствующие таким малозначимым направлениям, снижаются почти до нуля благодаря использованию регуляризации в ходе обучения.

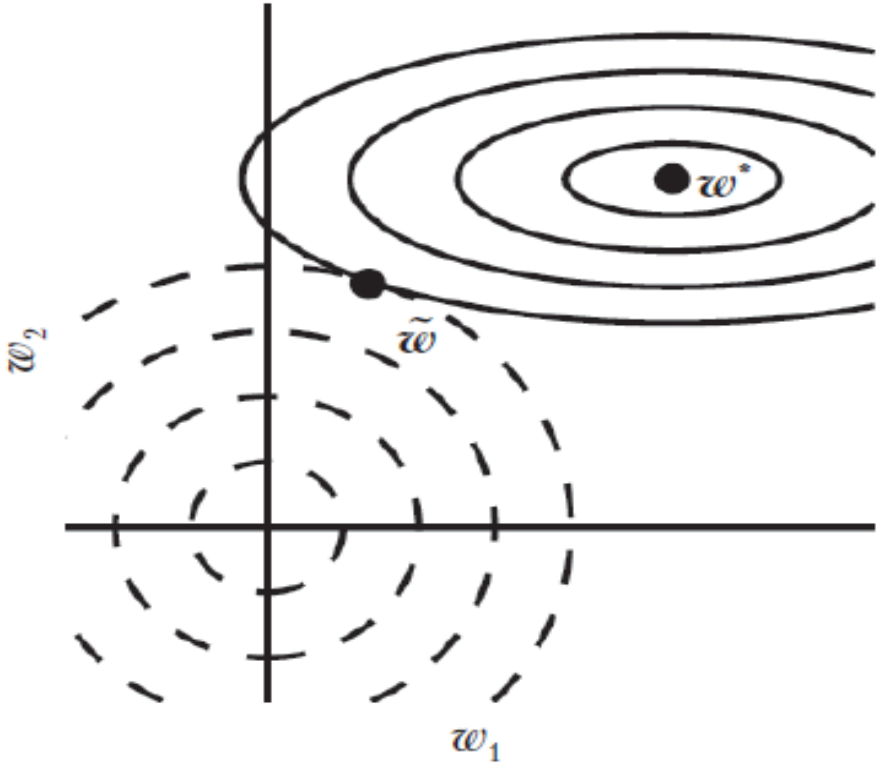


Иллюстрация влияния регуляризации по норме L^2 (снижения весов) на значение оптимального вектора w.

Регуляризация

L^2 регуляризация

На рисунке сплошными эллипсами представлены линии равных значений нерегуляризированной целевой функции, а пунктирными – линии равных значений L^2-регуляризатора. В точке $\tilde{w}$ эти конкурирующие цели достигают равновесия. По первому измерению собственное значение гессиана J мало. Целевая функция слабо растет при удалении от w* по горизонтали. Поскольку целевая функция не показывает предпочтения к этому направлению, регуляризатор дает для него значительный эффект. Регуляризатор подтягивает w_1 ближе к нулю. По второму направлению целевая функция очень чувствительна к удалению от w*. Соответствующее собственное значение велико, наблюдается сильная кривизна линий. Поэтому снижение весов влияет на положение w_2 сравнительно слабо.

Регуляризация

L^1 регуляризация

Регуляризация по норме L^2 – самая распространенная форма снижения весов, но есть и другие способы штрафовать за величину параметров модели. Одним из них является L^1-регуляризация.

Формально L^1-регуляризация параметров модели w определяется по формуле $$\Omega(\boldsymbol{\theta}) = \|\boldsymbol{w}\|_1 = \sum_i |w_i|,$$

т. е. как сумма абсолютных величин отдельных параметров. Снова рассмотрим простую квадратичную аппроксимацию целевой функции, опустим параметр смещения. Определим различия между двумя видами регуляризации. Как и в предыдущем случае, сила регуляризации контролируется путем умножения штрафа на положительный гиперпараметр α. Таким образом, регуляризированная целевая функция $\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y})$ описывается формулой $$\tilde{J}(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}) = \alpha\|\boldsymbol{w}\|_1 + J(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{y}),$$

Регуляризация

L^1 регуляризация

а ее градиент (точнее, частичный градиент) равен

$$\nabla_w \tilde{J}(w; X, y) = \alpha \operatorname{sign}(w) + \nabla_w J(w; X, y),$$

где sign(w) означает, что функция sign применяется к каждому элементу w.

Из последнего уравнения видно, что эффект L^1-регуляризации не такой, как L^2-регуляризации. Теперь вклад регуляризации в градиент уже не масштабируется линейно с ростом каждого w_i, а описывается постоянным слагаемым, знак которого совпадает с $\operatorname{sign}(w_i)$. Одним из следствий является тот факт, что мы уже не получим алгебраических выражений квадратичной аппроксимации J(X; y, w), как в случае L^2-регуляризации.

Квадратичную функцию стоимости можно представить ее рядом Тейлора. Будем считать, что это первые члены ряда Тейлора, аппроксимирующие функцию стоимости более сложной модели.

Регуляризация

L^1 регуляризация

Градиент в этой конфигурации равен

$$\nabla_w \hat{J}(w) = H(w - w^*)$$

где H –матрица Гессе J относительно w, вычисленная в точке w*.

Поскольку штраф по норме L^1 не допускает простого алгебраического выражения в случае полного гессиана общего вида, то мы сделаем еще одно упрощающее предположение: будем считать матрицу Гессе диагональной, $H = \mathrm{diag}([H_{1,1}, \ldots, H_{n,n}])$, где все $H_{i,i} > 0$. Это предположение справедливо, если данные были предварительно обработаны для устранения корреляции между входными признаками.

Рассматриваемую квадратичную аппроксимацию L^1 - регуляризированной целевой функции можно представить в виде суммы по параметрам:

$$\hat{J}(w; X, y) = j(w^*; X, y) + \sum_i \left[\frac{1}{2} H_{i,i} (w_i - w_i^*)^2 + \alpha |w_i| \right].$$

Регуляризация

L^1 регуляризация

У задачи минимизации этой приближенной функции стоимости имеется аналитическое решение (для каждого измерения i) вида:

$$w_i = \text{sign}(w_i^*)\max\left\{|w_i^*| - \frac{\alpha}{H_{i,i}}, 0\right\}$$

Предположим, что $w_i^* > 0$ для всех i. Тогда есть два случая:

1) $w_i^* \le \alpha / H_{i,i}$. Тогда оптимальное значение w_i для регуляризированной целевой функции будет просто $w_i = 0$. Причина в том, что вклад J(w; X, y) в регуляризированную целевую функцию перевешивается – в направлении i – L^1-регуляризацией, которая сдвигает значение w_i в нуль;
2) $w_i^* > \alpha / H_{i,i}$. Тогда регуляризация не сдвигает оптимальное значение w_i в нуль, а просто смещает его в этом направлении на расстояние $\alpha/H_{i,i}$.

Аналогичное рассуждение подходит, когда $w_i^* < 0$, только L^1-штраф увеличивает w_i на $\alpha/H_{i,i}$ или обращает в 0.

Регуляризация

L^1 регуляризация

По сравнению с L^2-регуляризацией, L^1-регуляризация дает более разреженное решение. В этом контексте под разреженностью понимается тот факт, что у некоторых параметров оптимальное значение равно 0. Разреженность L^1-регуляризации является качественным отличием от поведения L^2-регуляризации. Уравнение (1) дает решение $\tilde{w}$ для L^2-регуляризации.

Применив к нему предположение о диагональности и положительной определенности гессиана H, которое мы приняли для анализа L^1-регуляризации, найдем, что

$$\tilde{w}_i = \frac{H_{i,i}}{H_{i,i} + \alpha} w_i^*$$

Если w_i^* было не равно 0, то и $\tilde{w}_i$ окажется не равным нулю. Это значит, что L^2-регуляризация не приводит к разреженности параметров, тогда как в случае L^1-регуляризации это возможно, если α достаточно велико.

Регуляризация

Штраф по норме как оптимизация с ограничениями

Рассмотрим функцию стоимости, регуляризированную путем добавления штрафа по норме параметров:

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(\theta).$$

Для минимизации функции при наличии ограничений мы можем построить обобщенную функцию Лагранжа, состоящую из исходной целевой функции и набора штрафов. Каждый штраф имеет вид произведения коэффициента, называемого множителем Каруша–Куна–Таккера, на функцию, показывающую, удовлетворяется ли ограничение. Если мы хотим, чтобы ограничение $\Omega(\theta)$ было меньше некоторой константы k, то можем определить следующую обобщенную функцию Лагранжа:

$$\mathcal{L}(\theta, \alpha; X, y) = J(\theta; X, y) + \alpha(\Omega(\theta) - k)$$

Решение задачи с ограничениями имеет вид:

$$\theta^* = \arg\min_{\theta} \max_{\alpha, \alpha \geq 0} \mathcal{L}(\theta, \alpha)$$

Регуляризация

Штраф по норме как оптимизация с ограничениями

Для решения этой задачи могут применяться различные процедуры – метод градиентного спуска, аналитические методы, позволяющие искать точки, в которых градиент равен нулю, – но в любом случае α должно увеличиваться, когда $\Omega(\theta) > k$, и уменьшаться, когда $\Omega(\theta) < k$.

Любое положительное α приводит к уменьшению $\Omega(\theta)$. Оптимальное значение α^* поощряет уменьшение $\Omega(\theta)$, но не настолько сильно, чтобы $\Omega(\theta)$ оказалось меньше k.

Чтобы составить представление о влиянии ограничения, зафиксируем α^* и будем рассматривать задачу как функцию только от θ:

$$\theta^* = \arg\min_{\theta} \mathcal{L}(\theta, \alpha^*) = \arg\min_{\theta} J(\theta; X, y) + \alpha^* \Omega(\theta)$$

Это в точности то же самое, что регуляризированная задача обучения путем минимизации $\tilde{J}$. Таким образом, можно считать, что штраф по норме параметра налагает ограничение на веса.

Регуляризация

Штраф по норме как оптимизация с ограничениями

Если в качестве нормы Ω берется L^2, то веса должны лежать в единичном L^2-шаре. Если же Ω – это норма L^1, то веса должны лежать в области с ограниченной нормой L^1. Обычно нам неизвестен размер области ограничений, соответствующей снижению весов с коэффициентом α^*, поскольку значение α^* не говорит прямо о значении k. В принципе, можно решать уравнение относительно k, но связь между k и α^* зависит от вида J. Хотя мы не знаем точный размер области ограничений, мы можем грубо управлять им, уменьшая или увеличивая α с целью увеличить или уменьшить область ограничений. Чем больше α, тем меньше область ограничений, и наоборот.

Возможно использовать явные ограничения, а не штрафы.

Регуляризация

Штраф по норме как оптимизация с ограничениями

Возможно модифицировать алгоритм, например, стохастического градиентного спуска, так чтобы он производил шаг спуска по $J(\theta)$, а затем выполнял обратное проецирование θ в ближайшую точку, для которой $\Omega(\theta) < k$. Это полезно, если мы заранее знаем, какое значение k приемлемо, и хотим избежать временных затрат на поиск значения α, соответствующего этому k.

Еще одна причина использовать явные ограничения и обратное проецирование, а не косвенные ограничения в виде штрафов, состоит в том, что из-за штрафа процедура невыпуклой оптимизации может застрять в локальном минимуме, соответствующем малому θ. При обучении нейронных сетей это обычно проявляется в виде сети, обучившейся с несколькими «мертвыми блоками». Так называются блоки, которые мало что вносят в поведение обученной сетью функции, потому что веса всех входных или выходных сигналов очень малы.

Регуляризация

Штраф по норме как оптимизация с ограничениями

При обучении со штрафом на норму весов такие конфигурации могут оказаться локально оптимальными, даже если возможно значительно уменьшить J, сделав веса больше. Явные ограничения, реализованные с помощью обратного проецирования, в таких случаях могут работать гораздо лучше, т.к. не поощряют приближения весов к началу координат. Такие ограничения вступают в силу, только когда веса становятся большими и грозят выйти за пределы области ограничений.

Наконец, явные ограничения на основе обратного проецирования привносят устойчивость в процедуру оптимизации. Если скорость обучения высока, то есть риск попасть в петлю с положительной обратной связью, когда большие веса приводят к большим градиентам, а это, в свою очередь, приводит к большому обновлению весов. Явные ограничения с обратным проецированием не дают таким петлям вызывать неограниченный рост весов.

Регуляризация.
Метод увеличения объема данных

Эффективный способ повысить обобщаемость модели машинного обучения – обучить ее на большем объеме данных. Но, конечно же, на практике объем данных ограничен. Один из путей решения проблемы – добавить в обучающий набор искусственно сгенерированные данные. Для некоторых задач создать такие данные довольно легко.

Проще всего применить этот подход к классификации. Классификатор принимает сложный многомерный вход x и сопоставляет ему идентификатор категории y. Это означает, что для классификатора главное – инвариантность относительно широкого спектра преобразований. Мы легко можем сгенерировать новые пары (x, y) путем различных преобразований данных x, уже имеющихся в обучающем наборе.

Пополнение набора данных доказало высокую эффективность в задаче распознавания объектов.

Регуляризация.
Метод увеличения объема данных

Изображения характеризуются высокой размерностью и огромным количеством факторов изменчивости, многие из которых легко смоделировать. Такие операции, как сдвиг обучающих изображений на несколько пикселей в каждом направлении, могут значительно улучшить обобщаемость. Многие другие операции, например поворот или масштабирование изображения, также оказались весьма эффективными.

Пополнение набора данных эффективно также в задачах распознавания речи.

Пример применения метода при работе с изображениями (поворот изображений, масштабирование, сдвиг и т.д.) показан ниже.

Регуляризация

Робастность относительно шума

Наложение шума на входные данные может рассматриваться как стратегия пополнения набора данных. Для некоторых моделей добавление шума с очень малой дисперсией к входу модели эквивалентно штрафованию по норме весов.

Еще один способ использования шума ради регуляризации моделей – прибавление его к весам. Эта техника применялась в основном в контексте рекуррентных нейронных сетей. Прибавление шума к весам – практически удобный стохастический способ отразить неопределенность. Применение шума к весам можно также интерпретировать как эквивалент (при определенных допущениях) более традиционной формы регуляризации, поощряющей устойчивость обучаемой функции.

Регуляризация

Робастность относительно шума

Привнесение шума в выходные метки. В большинстве наборов данных выходные метки y содержат сколько-то ошибок. Максимизация log p(y | x), когда y ошибочно, чревата неприятными последствиями. Предотвратить это можно, в частности, путем явного моделирования шума в метках. Так, мы можем предположить, что для некоторой малой константы ε метка обучающего набора y правильна с вероятностью 1 – ε, а в противном случае правильной может быть любая другая возможная метка. Это предположение можно учесть в функции стоимости аналитически, а не путем явной выборки зашумленных примеров. Например, сглаживание меток регуляризирует модель, основанную на функции softmax с k выходными значениями, путем замены жестко заданных идентификаторов классов 0 и 1 метками ε/(k–1) и 1 – ε соответственно.

Регуляризация
Робастность относительно шума

Затем к этим мягким меткам можно применить стандартную функцию потерь на базе перекрестной энтропии. Обучение методом максимального правдоподобия с softmax-классификатором и жесткими метками может никогда не сойтись – softmax не умеет предсказывать вероятность, в точности равную 0 или 1, поэтому будет бесконечно продолжать обучать все большие и большие веса, делая все более экстремальные предсказания. Такого развития событий можно избежать, применив другие стратегии регуляризации, например снижение весов. У сглаживания меток есть то преимущество, что оно предотвращает стремление к жестким вероятностям, не ставя под угрозу правильность классификации. Эта стратегия используется начиная с 1980-х годов и до сих пор занимает видное место в современных нейронных сетях.

Регуляризация

Многозадачное обучение

Многозадачное обучение – способ улучшить обобщаемость путем пулинга примеров (которые можно рассматривать как мягкие ограничения на параметры), возникающих в нескольких задачах. Дополнительные обучающие примеры оказывают большее влияние на параметры модели, отдавая предпочтение значениям, которые хорошо обобщаются, и точно так же, если часть модели совместно используется несколькими задачами, то эту часть накладывается больше число ограничений, направляющих ее в сторону хороших значений (в предположении, что разделение обосновано), что зачастую улучшает обобщаемость.

Широко распространенная форма многозадачного обучения предусматривает, что разные задачи обучения с учителем (предсказание y(i) при известном x) разделяют один и тот же вход x, а также некоторое промежуточное представление h (shared), запоминающее общий пул факторов.

Регуляризация

Многозадачное обучение

В общем случае модель можно разделить на две части и ассоциированные с ними параметры:

1) параметры, специфичные для конкретной задачи (для достижения хорошей обобщаемости им нужны только примеры, относящиеся к «своей» задаче);
2) общие параметры, разделяемые всеми задачами (они извлекают выгоду из организации пула данных всех задач);

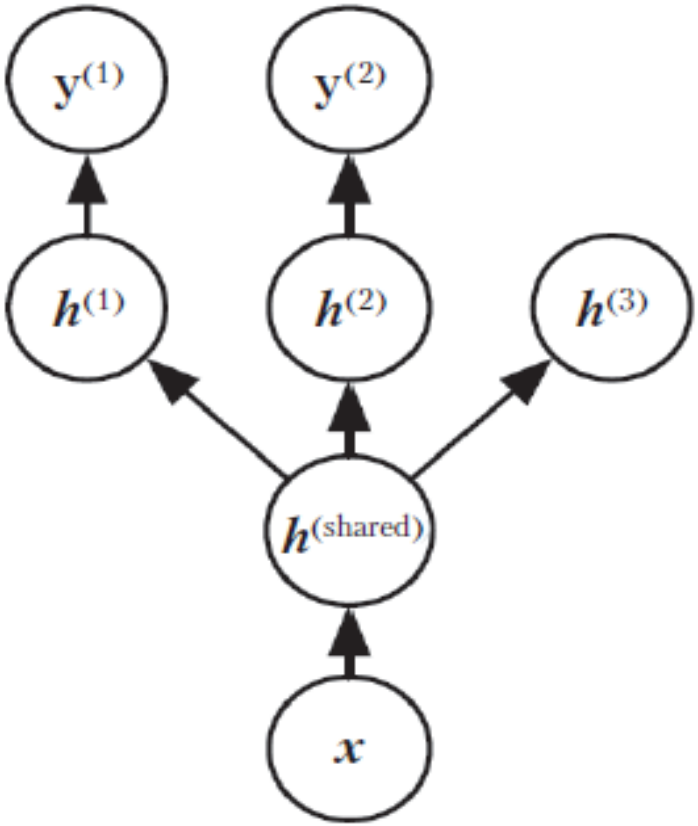
Улучшенной обобщаемости и ограничения ошибки обобщения можно достичь благодаря разделяемым параметрам, статистическую мощность которых можно значительно повысить (соразмерно с увеличенным числом примеров для разделяемых параметров, по сравнению со случаем однозадачных моделей). Разумеется, для этого нужно, чтобы выполнялись предположения относительно статистической связи между различными задачами, выражающие тот факт, что у некоторых задач действительно имеется что-то общее.

Регуляризация

Многозадачное обучение

С точки зрения глубокого обучения, априорная гипотеза состоит в следующем: среди факторов, объясняющих вариативность, наблюдаемую в данных, ассоциированных с разными задачами, некоторые являются общими для двух или более задач.

На рисунке показана типовая форма многозадачного обучения



Регуляризация

Многозадачное обучение

На рисунке показана ситуация, когда задачи разделяют общий вход, но у них разные выходные случайные величины. Нижние слои глубокой сети могут разделяться задачами, тогда как специфичные для задач параметры (ассоциированные соответственно с весами входных и выходных сигналов $\boldsymbol{h}^{(1)}$ и $\boldsymbol{h}^{(2)}$) могут быть обучены в дополнение к тем, что входят в разделяемое представление $\boldsymbol{h}^{(\text{shared})}$.

Предполагается, что существует общий пул факторов, объясняющих вариативность входа $\mathbf{x}$, а с каждой задачей ассоциировано подмножество этих факторов. В примере дополнительно предполагается, что блоки верхнего скрытого слоя $\boldsymbol{h}^{(1)}$ и $\boldsymbol{h}^{(2)}$ специализированы для каждой задачи (предсказывают соответственно $\mathbf{y}^{(1)}$ и $\mathbf{y}^{(2)}$), тогда как некое промежуточное представление $\boldsymbol{h}^{(\text{shared})}$ разделяется всеми задачами. ($\boldsymbol{h}^{(3)}$) - это те факторы, которые объясняют часть вариативности входа, но не имеют отношения к предсказаниям $\mathbf{y}^{(1)}$ или $\mathbf{y}^{(2)}$.

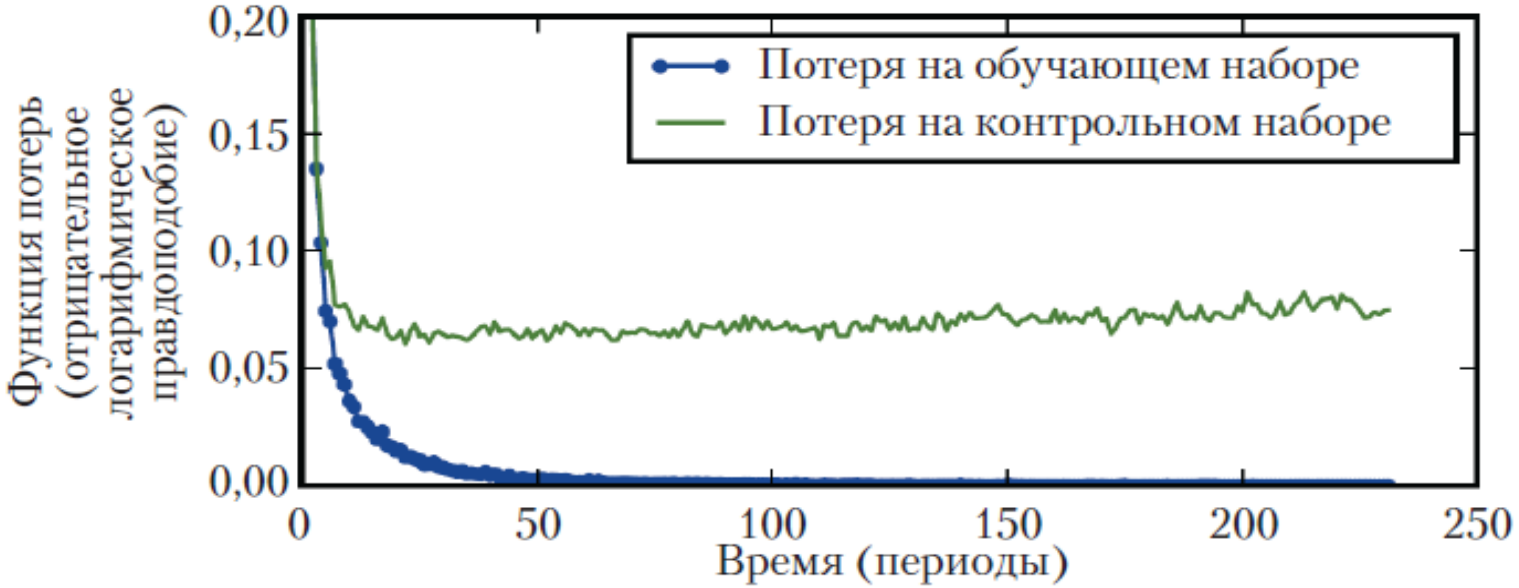
Регуляризация

Метод ранней остановки

При обучении больших моделей, репрезентативная емкость которых достаточна для переобучения, мы часто наблюдаем, что ошибка обучения монотонно убывает со временем, но ошибка на контрольном наборе снова начинает расти.

Это означает, что мы могли бы получить модель с более низкой ошибкой на контрольном наборе (и, хочется надеяться, на тестовом тоже), вернувшись к тем значениям параметров, которые существовали на момент наименьшей ошибки. Всякий раз как ошибка на контрольном наборе улучшается, мы сохраняем копию параметров модели. Когда алгоритм обучения завершается, мы возвращаем именно эти, а не самые последние параметры. А завершается алгоритм, когда на протяжении заранее заданного числа итераций не удается улучшить параметры, по сравнению с наилучшими запомненными ранее. Эта стратегия называется ранней остановкой.

Регуляризация
Многозадачное обучение



На рисунке показаны кривые обучения, отражающие изменение отрицательного логарифмического правдоподобия со временем (представленного в виде количества итераций обучения на наборе данных, или периодов). Отметим, что целевая функция монотонно убывает, но средняя потеря на контрольном наборе в какой-то момент снова начинает расти, так что образуется U-образная кривая.

Регуляризация

Метод ранней остановки

Считается, что стратегия ранней остановки является самой распространенной формой регуляризации в машинном обучении. Своей популярностью она обязана простоте и эффективности.

Одна из возможных интерпретаций ранней остановки – очень эффективный алгоритм выбора гиперпараметров. С этой точки зрения, число шагов обучения – просто еще один гиперпараметр. Большинство гиперпараметров, управляющих емкостью модели, имеет кривую качества именно U-образной формы (как кривая, представленная на рис.). В случае ранней остановки мы управляем эффективной емкостью модели, определяя, сколько шагов ей может потребоваться для аппроксимации обучающего набора.

Ранняя остановка – ненавязчивая форма регуляризации в том смысле, что не требуется вносить почти никаких изменений в базовую процедуру обучения, целевую функцию или множество допустимых значений параметров. Следовательно, раннюю остановку можно легко использовать, не изменяя динамику обучения.

Регуляризация

Метод ранней остановки

Совершенно не так обстоит дело со снижением весов, когда нужно внимательно следить за тем, чтобы не снизить веса слишком сильно и не завести сеть в локальный минимум, соответствующий патологически малым весам.

Раннюю остановку можно использовать автономно или в сочетании с другими стратегиями регуляризации.

Для ранней остановки необходим контрольный набор, а значит, часть обучающих данных не следует подавать на вход модели. Чтобы использовать эти отложенные данные более эффективно, можно провести дополнительное обучение, после того как начальная фаза с ранней остановкой завершилась, и тогда уже включить все обучающие данные. На этом втором раунде обучения применяются две основные стратегии.

Первая стратегия – снова инициализировать модель и заново обучить ее на всех данных. Но при этом мы ограничиваемся числом шагов, которое было признано оптимальным в первом раунде. Тут есть некоторые тонкости.

Регуляризация

Метод ранней остановки

Например, не существует хорошего способа решить, следует ли при повторном обучении производить столько же обновлений параметров или столько же проходов по набору данных, как в первом раунде. Во втором раунде каждый проход по набору данных приводит к большему числу обновлений параметров, потому что сам обучающий набор больше.

Другая стратегия использования всех данных – оставить параметры, полученные на первом раунде, и продолжить обучение, но уже на всех данных. Теперь у нас больше нет подсказок, после скольких шагов остановиться. Вместо этого мы следим за функцией потерь на контрольном наборе и продолжаем обучение, пока ее среднее значение не окажется меньше величины целевой функции, при которой сработала ранняя остановка. Эта стратегия позволяет избежать дорогостоящего повторного обучения модели с нуля, но ее поведение оставляет желать лучшего. Например, целевая функция может никогда не достичь нужного значения на контрольном наборе, и тогда обучение не завершится.

Регуляризация

Метод ранней остановки

Ранняя остановка полезна также тем, что уменьшает вычислительную стоимость процедуры обучения. Помимо очевидного сокращения стоимости вследствие ограничения числа итераций, она еще и обеспечивает преимущества регуляризации, не требуя включения дополнительных штрафов в функцию стоимости и вычисления градиентов этих добавочных членов.

Ранняя остановка выступает в роли регуляризатора. Благодаря ранней остановке процедура оптимизации ограничивается просмотром сравнительно небольшой области пространства параметров в окрестности начального значения параметров θ_o. Допустим, что выбрано τ шагов оптимизации (что соответствует τ итерациям обучения) и скорость обучения ε. Произведение $\varepsilon\tau$ можно рассматривать как меру эффективной емкости. В предположении, что градиент ограничен, наложение ограничений на число итераций и скорость обучения лимитируют область пространства параметров, достижимую из θ_o. В этом смысле $\varepsilon\tau$ ведет себя как величина, обратная коэффициенту снижения весов.

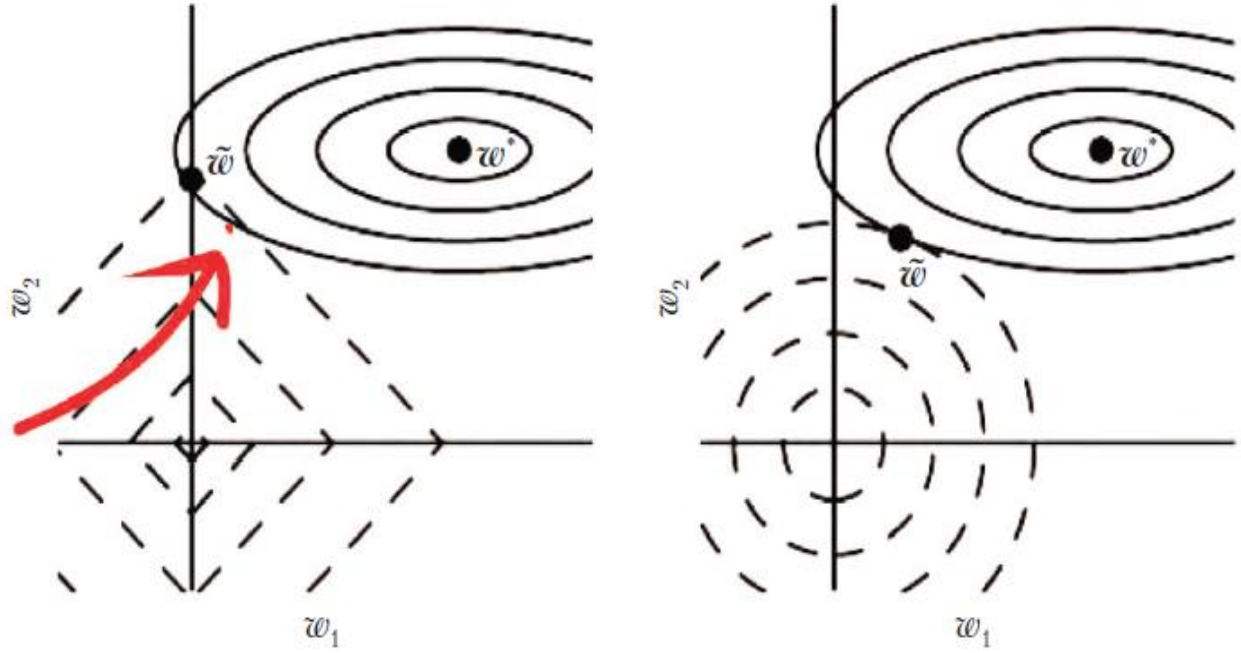
Регуляризация

Метод ранней остановки

В случае простой линейной модели с квадратичной функцией ошибки и обычного градиентного спуска ранняя остановка эквивалентна L^2-регуляризации. Траектория длины τ обрывается в точке, соответствующей минимуму L^2-регуляризированной целевой функции. Конечно, ранняя остановка – больше, чем простое ограничение на длину траектории; ранняя остановка обычно подразумевает наблюдение за ошибкой на контрольном наборе, чтобы оборвать траекторию в удачной точке пространства. Поэтому, по сравнению со снижением весов, у ранней остановки есть то преимущество, что она автоматически определяет правильную степень регуляризации, тогда как при использовании снижения весов требуется много экспериментов с разными значениями гиперпараметра.

Регуляризация

Метод ранней остановки



На рис. (Слева) Сплошными линиями показаны изолинии отрицательного логарифмического правдоподобия, штриховыми – траектория стохастического градиентного спуска, начатого из начала координат. Благодаря ранней остановке траектория заканчивается не в точке w*, минимизирующей стоимость, а в более ранней точке $\tilde{w}$. (Справа) Результат L^2-регуляризации для сравнения. Штриховыми кругами показаны изолинии штрафа по норме L^2, благодаря которому минимум полной функции стоимости оказывается ближе к началу координат, чем минимум нерегуляризированной функции стоимости.

Регуляризация

Связывание и разделение параметров

На практике возможны ситуации, когда мы не знаем точно, какие значения должны принимать параметры, но имеющиеся знания о предметной области и архитектуре модели позволяют заключить, что между параметрами должны существовать некие зависимости.

Распространенный тип зависимости – близость некоторых параметров друг к другу. Рассмотрим такую ситуацию: есть две модели, решающие одну и ту же задачу классификации (с одинаковым набором классов), но с различающимися распределениями входных данных. Формально имеется модель A с параметрами w(A) и модель B с параметрами w(B). Обе модели отображают входы в разные, но взаимосвязанные выходы: y (A) = f(w(A), x) и y (B) = g(w(B), x).

Допустим, что задачи похожи настолько (быть может, имеют похожие распределения входов и выходов), что есть основания полагать, что их параметры должны быть близки: $\forall i$ $w_i(A)$ должно быть близко к $w_i(B)$. Эту информацию можно применить для регуляризации: использовать штраф по норме параметров вида $\Omega(w(A), w(B)) == ||w(A) - w(B)||^2_2$. В данном случае для вычисления штрафа мы применили норму L^2, но возможны и другие варианты.

Регуляризация

Связывание и разделение параметров

При таком подходе параметры одной модели, обученной для классификации с учителем, регуляризируются с целью сделать их близкими параметрам другой модели, обученной без учителя (для нахождения распределения наблюдаемых входных данных). Хотя штраф по норме параметров можно использовать для регуляризации параметров с целью обеспечить их близость, более популярен другой способ: наложить ограничения, требующие, чтобы множества параметров совпадали. Этот метод регуляризации часто называют разделением параметров, поскольку мы считаем, что разные модели или компоненты моделей разделяют некоторое общее множество параметров. Важным преимуществом разделения параметров над регуляризацией посредством штрафа по норме с целью обеспечения близости является тот факт, что в памяти нужно хранить только подмножество всех параметров (общее множество).

Регуляризация

Связывание и разделение параметров

Сверточные нейронные сети. Самое популярное и важное применение идея разделения параметров находит в сверточных нейронных сетях (СНС) в компьютерном зрении. У естественных изображений есть много статистических свойств, инвариантных к параллельному переносу. Например, фотография кошки остается таковой, если сдвинуть ее на один пиксель вправо. В СНС это свойство учитывается с помощью разделения параметров по нескольким областям изображения. Один и тот же признак (скрытый блок с одинаковыми весами) вычисляется по нескольким участкам входных данных. Это означает, что один и тот же детектор кошек найдет кошку вне зависимости от того, находится она в столбце изображения с номером i или i + 1.

Разделение параметров дает СНС возможность значительно уменьшить число уникальных параметров модели и увеличить размер сети, не требуя соответственного увеличения объема обучающих данных.

Регуляризация

Разреженные представления

Снижение весов работает благодаря штрафованию непосредственно параметров модели. Другая стратегия – штрафовать за активацию блоков нейронной сети, стремясь к разреженности активаций. Это налагает косвенный штраф на параметры модели.

Штраф по норме L^1 ведет к разреженной параметризации, т. е. многие параметры обращаются в 0 (или оказываются близки к 0). С другой стороны, под разреженным понимается такое представление, многие элементы которого равны 0 (или близки к 0).

Для регуляризации представлений применяются те же механизмы, что для регуяризации параметров. Регуляризация представления путем штрафа по норме производится путем прибавления к функции потерь J штрафа по норме представления. Этот штраф обозначается $\Omega(h)$.

Как и раньше, будем обозначать регуляризированную функцию потерь :

$$\tilde{J}(\theta; X, y) = J(\theta; X, y) + \alpha\Omega(h),$$

где $\alpha \in [0, \infty)$ – относительный вес штрафа; чем больше α, тем сильнее регуляризация.

Регуляризация

Разреженные представления

Точно так же, как штраф по норме L^1 параметров индуцирует разреженность параметров, штраф по норме L^1 элементов представления индуцирует разреженность представления: $\Omega(h) = \|h\|_1 = \Sigma_i |h_i|$. Разумеется, L^1-штраф – лишь один из возможных штрафов, приводящих к разреженному представлению. Из других назовем штрафы на основе априорного t-распределения Стьюдента и расхождения Кульбака–Лейблера, особенно полезные для представлений, элементы которых принадлежат единичному интервалу. Имеются стратегии, основанные на регуляризации, требующей, чтобы активация, усредненная по нескольким примерам $(1/m)\Sigma_i h_i$, была близка к некоторому целевому значению, скажем, вектору, все элементы которого равны 0,01. В других подходах разреженность представления достигается за счет жесткого ограничения на значения активации.

Регуляризация

Разреженные представления

Например, в методе ортогонального согласованного преследования (orthogonal matching pursuit – OMP) вход x кодируется с помощью представления h, решающего следующую задачу оптимизации с ограничениями:

$$\arg\min_{\boldsymbol{h},\|\boldsymbol{h}\|_0<k} \|\boldsymbol{x}-\boldsymbol{W}\boldsymbol{h}\|^2$$

где $\|h\|_0$ – число ненулевых элементов h. Эту задачу можно эффективно решить, когда на матрицу W наложено ограничение ортогональности. Этот метод часто называют OMP-k, где k обозначает допустимое число ненулевых признаков. OMP-k может быть чрезвычайно эффективным экстрактором признаков для глубоких архитектур.

Отметим, что любую модель со скрытыми блоками можно сделать разреженной.

Регуляризация
Прореживание

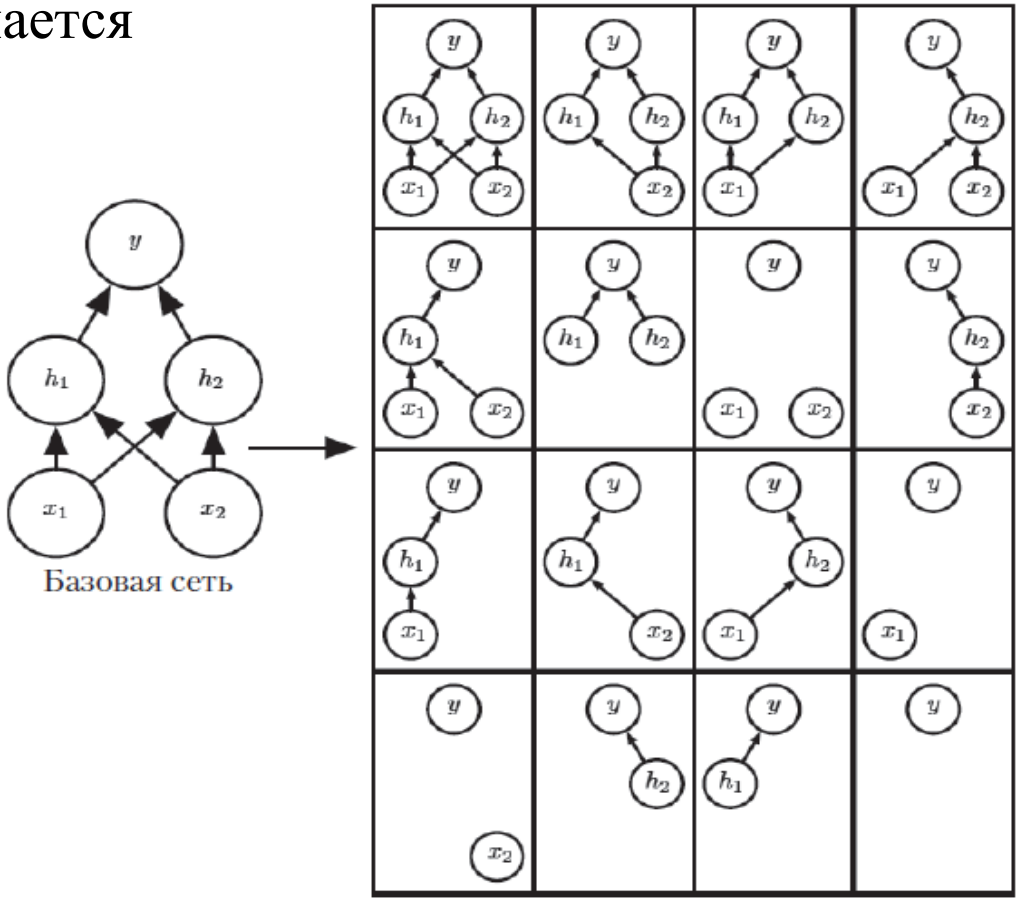
Прореживание (dropout) – это вычислительно недорогой, но мощный метод регуляризации широкого семейства моделей. Прореживание предлагает дешевую аппроксимацию обучения и вычисления ансамбля экспоненциально большого числа нейронных сетей.

В процессе прореживания обучается ансамбль, состоящий из подсетей, получаемых удалением невыходных блоков из базовой сети. В большинстве современных нейронных сетей, основанных на последовательности аффинных преобразований и нелинейностей, можно эффективно удалить блок, умножив его выход на 0.

Из рисунка видно, что прореживание обучает ансамбль, состоящий из всех подсетей, которые можно построить путем удаления невыходных блоков из базовой сети. В данном случае мы начинаем с сети, имеющей два видимых и два скрытых блока. Из этих четырех блоков можно составить 16 подмножеств. На рисунке показаны все 16 подсетей, которые можно построить выбрасыванием разных подмножеств блоков из исходной сети.

Регуляризация
Прореживание

В этом примере есть много сетей, в которых вообще нет входных блоков или не существует пути, соединяющего вход с выходом. Эта проблема теряет остроту для сетей с более широкими слоями, когда вероятность выбросить все возможные пути от входов к выходам уменьшается



Базовая сеть

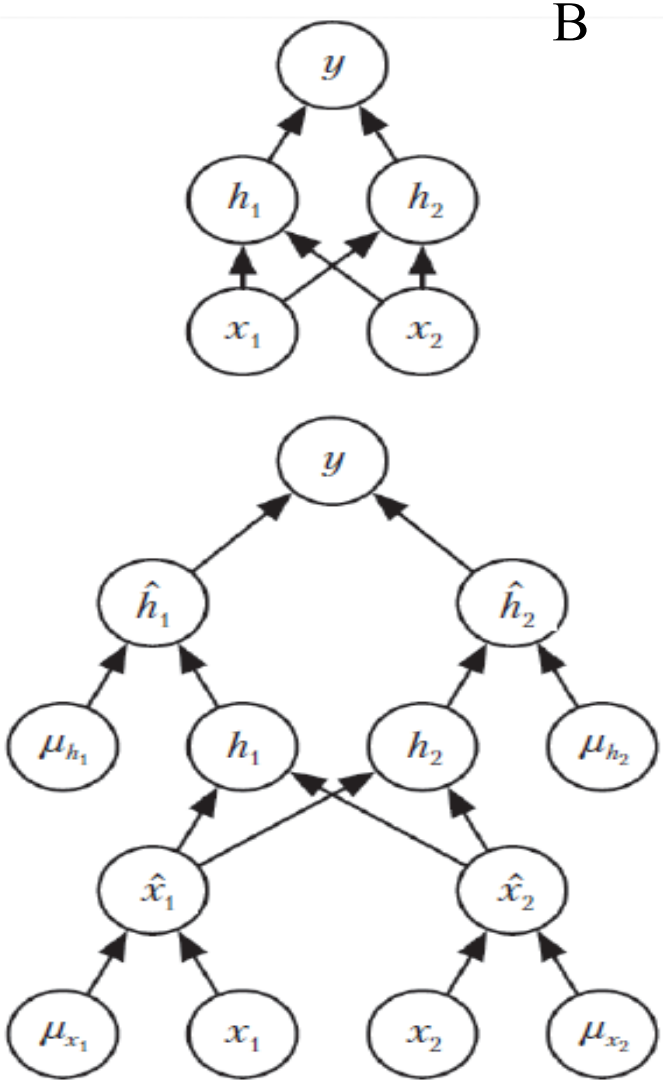
Ансамбль подсетей

Регуляризация
Прореживание

Для обучения методом прореживания используется алгоритм, основанный на мини-пакетах, который делает небольшие шаги, например алгоритм стохастического градиентного спуска. При загрузке каждого примера в мини-пакет случайным образом генерируется битовая маска, применяемая ко всем входным и скрытым блокам сети. Элемент маски для каждого блока выбирается независимо от всех остальных. Вероятность включить в маску значение 1 (означающее, что соответствующий блок включается) – гиперпараметр, фиксируемый до начала обучения, а не функция текущего значения параметров модели или входного примера. Обычно входной блок включается с вероятностью 0.8, а скрытый – с вероятностью 0.5. Затем, как обычно, производятся прямое распространение, обратное распространение и обновление. На рис. показано, как работает прямое распространение с прореживанием.

Регуляризация

Прореживание



В приведенном примере сеть прямого распространения имеет два входных блока, один скрытый слой с двумя блоками и один выходной блок. (Внизу) Для прямого распространения с прореживанием мы случайно выбираем вектор μ, имеющий по одному элементу для каждого входного и скрытого блока. Элементами μ являются нули или единицы, и каждый элемент выбирается независимо от других. Вероятность единицы – гиперпараметр, который обычно равен 0.5 для скрытых блоков и 0.8 для входных. Каждый блок сети умножается на соответствующий элемент маски, а затем прямое распространение продолжается по оставшейся части сети, как обычно. Это эквивалентно случайному выбору одной из подсетей и прямому распространению в ней

Регуляризация
Прореживание

Формально, пусть вектор-маска μ определяет, какие блоки включать, а $J(\theta, \mu)$ – стоимость модели с параметрами θ и маской μ. Тогда обучение с прореживанием заключается в минимизации $\mathbb{E}_\mu J(\theta, \mu)$. Математическое ожидание содержит экспоненциально много членов, но мы можем получить несмещенную оценку его градиента путем выборки значений μ.

В случае прореживания каждая подмодель, задаваемая вектором-маской μ, определяет распределение вероятности $p(y \mid x, \mu)$. Среднее арифметическое по всем маскам равно

$$\sum_{\mu} p(\mu) p(y \mid x, \mu),$$

где $p(\mu)$ – распределение вероятности, которое использовалось для выборки μ на этапе обучения.

Поскольку количество членов в этой сумме экспоненциально велико, вычислить ее можно только в случае, когда структура модели допускает какое-то упрощение. Пока что неизвестны нейронные сети, допускающие такое упрощение.

Регуляризация
Прореживание

Но можно аппроксимировать вывод с помощью выборки, усреднив выходы для нескольких масок. Даже 10–20 масок часто достаточно для получения хорошего качества.

Есть подход еще лучше – он позволяет получить хорошую аппроксимацию предсказаний всего ансамбля, произведя всего одно прямое распространение. Для этого мы вместо среднего арифметического распределений, предсказанных членами ансамбля, будем вычислять среднее геометрическое. Показано, что в этом контексте среднее геометрическое дает качество, сравнимое со средним арифметическим.

Вообще говоря, не гарантируется, что среднее геометрическое нескольких распределений вероятности само является распределением вероятности. Чтобы получить такую гарантию, потребуем, чтобы ни одна подмодель не назначала никаким событиям вероятность 0, а затем произведем нормировку получившегося распределения.

Регуляризация
Прореживание

Ненормированное распределение вероятности, определяемое средним геометрическим, имеет вид:

$$\tilde{p}_{\text{ensemble}}(y \mid \boldsymbol{x}) = \sqrt[2^d]{\prod_{\mu} p(y \mid \boldsymbol{x}, \boldsymbol{\mu})},$$

где d – число блоков, которые можно выбросить. Здесь для простоты взято равномерное распределение μ, но неравномерные распределения тоже допустимы. Чтобы можно было делать предсказания, необходимо перенормировать ансамбль:

$$p_{\text{ensemble}}(y \mid \boldsymbol{x}) = \frac{\tilde{p}_{\text{ensemble}}(y \mid \boldsymbol{x})}{\sum_{y'} \tilde{p}_{\text{ensemble}}(y' \mid \boldsymbol{x})}$$

Основная идея прореживания состоит в том, что p_{ensemble} можно аппроксимировать, вычислив p(y | x) для одной модели: она включает все блоки, но веса связей, исходящих из i-го блока, умножаются на вероятность включения этого блока. Обоснование такой модификации – стремление получить правильное ожидаемое значение выхода блока. Этот подход называется правилом вывода с масштабированием весов.

Регуляризация
Прореживание

Поскольку обычно принимается вероятность включения 1/2, то правило масштабирования весов сводится к делению весов пополам в конце обучения, после чего модель используется как обычно. Другой способ получить тот же результат – умножать состояния блоков на 2 на этапе обучения. В любом случае цель состоит в том, чтобы ожидаемый суммарный вход в блок на этапе тестирования был приблизительно равен ожидаемому суммарному входу в тот же блок на этапе обучения, несмотря на то что в среднем половина блоков во время обучения отсутствует.

Для многих классов моделей, не имеющих нелинейных скрытых блоков, правило вывода с масштабированием весов является точным. Оно также является точным и в других конфигурациях, в т. ч. в регрессионных сетях с условно нормальными распределениями на выходе, а также в глубоких сетях, в скрытых слоях которых нет нелинейностей. Однако для глубоких моделей с нелинейностями это правило всего лишь аппроксимация. Хотя теоретической оценки этой аппроксимации не существует, на практике она часто дает хорошие результаты.

Регуляризация
Прореживание

Дальнейшего улучшения можно добиться, комбинируя прореживание с другими видами регуляризации.

Одно из преимуществ прореживания – вычислительная простота. Применение прореживания на этапе обучения требует всего O(n) вычислений на каждый пример на каждое обновление – для генерирования n случайных двоичных чисел и умножения их на состояние. Стоимость вывода с помощью обученной модели в расчете на один пример такая же, как если бы прореживание не использовалось, хотя к накладным расходам следует отнести стоимость однократного деления весов на 2 до применения вывода к примерам.

У прореживания есть еще одно важное преимущество: оно не налагает существенных ограничений на тип модели или процедуру обучения. Оно одинаково хорошо работает практически с любой моделью, если в ней используется распределенное представление и ее можно обучить методом стохастического градиентного спуска.

Регуляризация
Прореживание

Сюда входят нейронные сети прямого распространения, и рекуррентные нейронные сети. Многие другие стратегии регуляризации сравнимой мощности налагают куда более строгие ограничения на архитектуру модели.

Хотя стоимость одного шага применения прореживания к конкретной модели пренебрежимо мала, его общая стоимость для модели в целом может оказаться значительной. Будучи методом регуляризации, прореживание уменьшает эффективную емкость модели. Чтобы компенсировать этот эффект, мы должны увеличить размер модели. Как правило, оптимальная ошибка на контрольном наборе при использовании прореживания намного ниже, но расплачиваться за это приходится гораздо большим размером модели и числом итераций алгоритма обучения. Для очень больших наборов данных регуляризация не сильно снижает ошибку обобщения. В таких случаях вычислительная стоимость прореживания и увеличение модели могут перевесить выигрыш от регуляризации.

Регуляризация
Прореживание

Если в нашем распоряжении очень мало помеченных обучающих примеров, а также имеются непомеченные данные, то прореживание менее эффективно, чем другие методы.

Использование стохастичности в обучении с прореживанием не является необходимым условием успеха. Это просто средство аппроксимации суммы по всем подмоделям. Рассмтривалась возможность использования аналитической аппроксимации. Такой метод получил название «быстрое прореживание». Он сходится быстрее благодаря уменьшению стохастичности при вычислении градиента. Этот метод можно применять и на стадии тестирования, как теоретически более обоснованную (хотя вычислительно более дорогую) аппроксимацию среднего во всем подсетям, по сравнению с масштабированием весов. Быстрое прореживание по качеству почти не уступает стандартному на небольших нейронных сетях, но пока не сумело достичь существенного улучшения и не применялось к большим задачам.

Регуляризация
Прореживание

Следует отметить, что стохастичность не является достаточным условием для достижения регуляризирующего эффекта прореживания. Чтобы продемонстрировать это был разработан метода усиленного прореживания (dropout boosting), в котором используется точно такой же масочный шум, что в традиционном прореживании, однако эффект регуляризации не достигается. Алгоритм усиленного прореживания обучает ансамбль совместно максимизировать логарифмическое правдоподобие на обучающем наборе. Как и предполагалось, эксперименты с усиленным прореживанием не показали почти никакого эффекта регуляризации, по сравнению с обучением всей сети как одной модели. Эффект регуляризации ансамбля достигается, только когда стохастически выбранные члены ансамбля обучаются хорошо работать независимо друг от друга.

Идея прореживания дала начало другим стохастическим подходам к обучению экспоненциально больших ансамблей моделей, разделяющих веса.

Регуляризация
Прореживание

Метод DropConnect – частный случай прореживания, в котором каждое произведение одного скаляр ного веса и состояния одного скрытого блока рассматривается как блок, подлежащий выбрасыванию). В отличие от Dropout, случайному обнулению (обычно так же с вероятностью 50%) подвергаются не выходы нейронов, а их веса:

$$v_j^k = \sum_i M_{ij} w_{ij}^k y_i^{k-1},$$

где Mij – матричная маска, состоящая из независимых бинарных величин, подчиняющихся распределению Бернулли.

Существует более общий взгляд на прореживание. В соответствии с ним метод прореживания обучает не просто ансамбль моделей, а ансамбль моделей с общими скрытыми блоками. Это означает, что каждый скрытый блок должен демонстрировать хорошее поведение вне зависимости от того, какие еще скрытые блоки есть в модели. Скрытые блоки должны быть готовы к тому, что окажутся в другой модели.

Использованные материалы

В презентации использованы материалы, предоставленные:

Бутырским Евгением Юрьевичем, доктором физико-математических наук, профессором кафедры теории управления СПбГУ;

Гришкиным Валерием Михайловичем, кандидат технических наук, доцентом кафедры компьютерного моделирования и многопроцессорных систем.

А также следующие источники:

Ян Гудфеллоу, Аарон Курвилль, Иошуа Бенджио Глубокое обучение