Омский Государственный
ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

Онлайн-курс

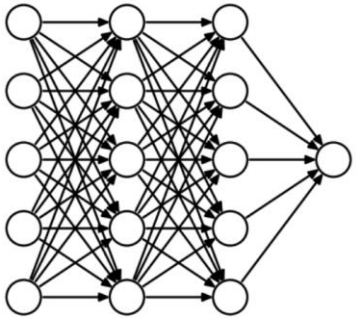
ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

8. Дропаут и нормализация по мини-батчам

Дропаут

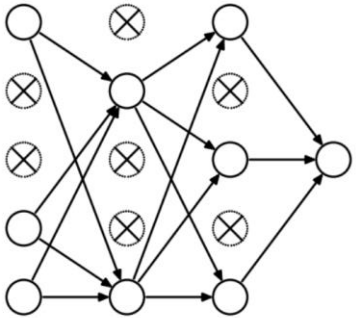
Неужели русского слова не нашлось?

Дропаут – это преднамеренная деактивация части нейронов на шаге обучения.



Зачем заниматься таким извращением?

Это реинкарнация идеи **ансамбля** – очень популярной вещи в машинном обучении.



Ансамбль (идея)

Если у вас есть один мозг (биологический или искусственный), то он может иногда ошибаться в принятии решений.

А вот если у вас будет целый **ансамбль мозгов,** то можно выслушать ответ каждого мозга и в качестве окончательного решения взять «усредненный ответ».

Точность ансамбля выше точности каждого из его членов.

Ансамбль в МО и ИИ

На идее ансамбля построены многие классические модели МО и ИИ. **Например, RandomForest**.

Там идея такая: у вас есть деревья решений, а вы из них составляете ансамбль (в данном случае: лес).

Лес принимает коллегиальное решение.

Каждое дерево в лесу строится по случайному подмножеству нецелевых признаков и случайному подмножеству объектов ТВ.

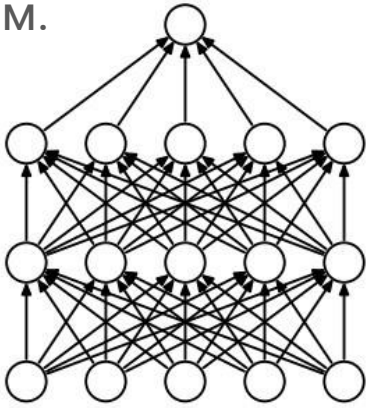
Вот НС. Где тут ансамбль?

Мы тренируем одиночные НС. Где вы видите тут ансамбль?

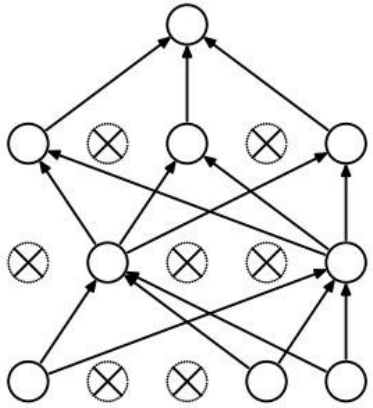
А разве НС не могут страдать раздвоением личности?))))

Мы можем условно разделить НС на части. Каждая часть будет формально считаться автономным мозгом.

Ответ всей НС будет фактически усредненным значением нескольких ее подсетей.



Standard Neural Net



After applying dropout

Неужели русского слова не нашлось?

Дропаут – это преднамеренная деактивация части нейронов на шаге обучения (а ещё это деталь велосипеда).

Перед тем, как произвести один шаг ГС, мы случайным образом **временно исключаем** из НС часть нейронов. Формально возникает новая сеть с новой функцией $F_{NN}(x)$ и новой функцией потерь $L(w)$. По этой функции потерь осуществляется один шаг ГС и пересчитываются веса сети.

Примечание: если после исключения части нейронов НС **перестает быть связной**, то есть в ней нет пути от входного слоя к выходу, то мы формируем новое множество нейронов для исключения (пока сеть без этих нейронов не станет связной).

Как реализовать убийство нейронов?

Фиксируется число *p* – вероятность «смерти» нейрона (обычно *p* берут из интервала [0.2, 0.5] – слишком большие и слишком малые значения *p* – плохо).

На каждой итерации ГС для каждого нейрона НС проводится **случайное испытание** – исключать его или нет.

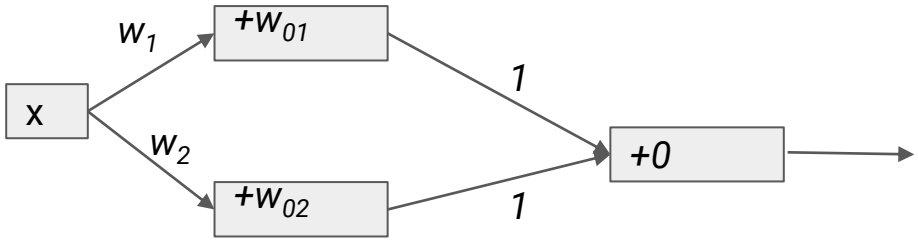
Формируется НС из «выживших» нейронов, для неё выписывается своя функция потерь. После этого осуществляется **одна** итерация ГС и дропаут повторяется.

Пример

Рассмотрим НС. Часть весов в ней уже зафиксирована (для простоты). Два нейрона из внутреннего слоя будем для краткости называть **верхним и нижним**.

Тренировочная выборка:

x	Y
1	3
2	2
3	1



Функция сети (обычная): $F_{NN}(x)=(w_1x+w_{01})+(w_2x+w_{02})$

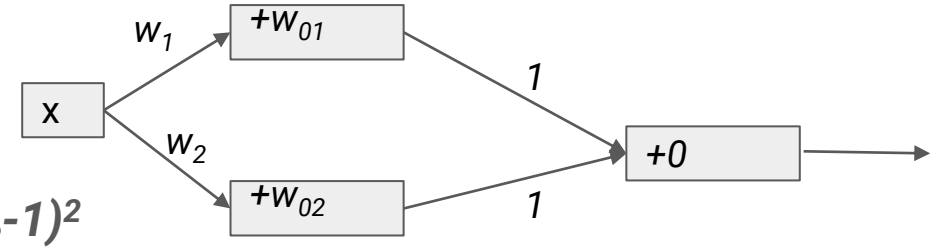
Функция потерь (обычная): $L(w)=(F_{NN}(1)-3)^2+(F_{NN}(2)-2)^2+(F_{NN}(3)-1)^2$

Шаг обучения $h=0.1$, начальное значение всех весов 0.
Вероятность исключения нейрона 0.5

Первая итерация

Допустим, при дропауте был **выброшен верхний нейрон**.

Тогда функция сети: $F_{NN}(x)=(w_2 x+w_{02})$.



Теперь функция потерь:

$$L_1(w)=(w_2+w_{02}-3)^2+(2w_2+w_{02}-2)^2+(3w_2+w_{02}-1)^2$$

Считаем ЧП:

$$\frac{\partial L_1}{\partial w_1} = \frac{\partial L_1}{\partial w_{01}} = 0,$$

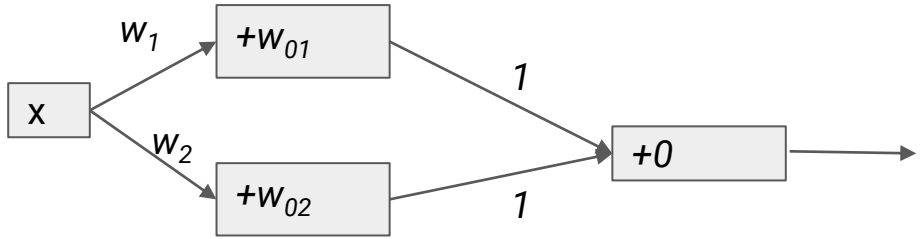
$$\frac{\partial L_1}{\partial w_2} = 2(w_2 + w_{02} - 3) + 4(2w_2 + w_{02} - 2) + 6(3w_2 + w_{02} - 1),$$

$$\frac{\partial L_1}{\partial w_{02}} = 2(w_2 + w_{02} - 3) + 2(2w_2 + w_{02} - 2) + 2(3w_2 + w_{02} - 1)$$

Первая итерация

Значения ЧП в точке спуска (напомним, что сейчас все веса=0) равны:

$$\frac{\partial L_1}{\partial w_1} = \frac{\partial L_1}{\partial w_{01}} = 0, \frac{\partial L_1}{\partial w_2} = -20, \frac{\partial L_1}{\partial w_{02}} = -12$$



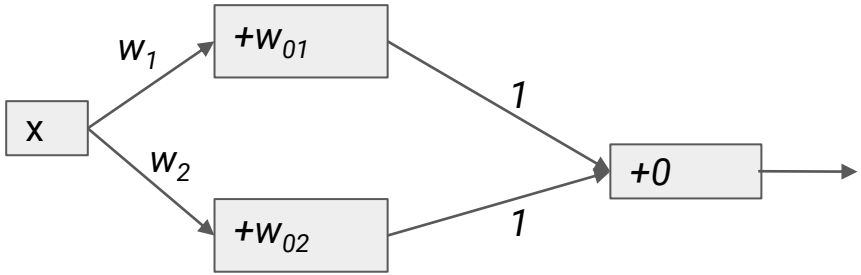
Новая точка спуска равна:

$w_1=0, w_{01}=0, w_2=2, w_{02}=1.2$

Щас будет вторая итерация, допустим, что на ней **выпилили нижний нейрон**.

Вторая итерация

Тогда функция сети: $F_{NN}(x)=(w_1x+w_{01})$



Теперь функция потерь:

$L_2(w)=(w_1+w_{01}-3)^2+(2w_1+w_{01}-2)^2+(3w_1+w_{01}-1)^2$

Считаем ЧП:

$$\frac{\partial L_2}{\partial w_2} = \frac{\partial L_2}{\partial w_{02}} = 0,$$

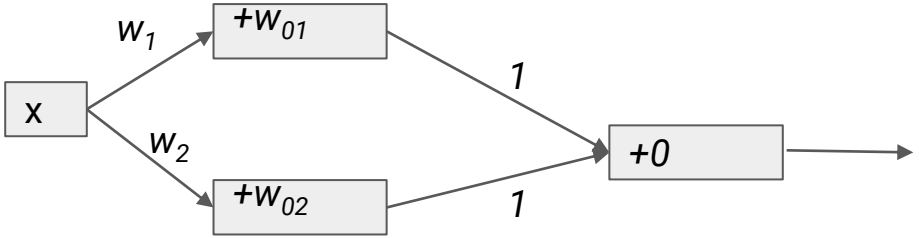
$$\frac{\partial L_2}{\partial w_1} = 2(w_1 + w_{01} - 3) + 4(2w_1 + w_{01} - 2) + 6(3w_1 + w_{01} - 1),$$

$$\frac{\partial L_2}{\partial w_{01}} = 2(w_1 + w_{01} - 3) + 2(2w_1 + w_{01} - 2) + 2(3w_1 + w_{01} - 1)$$

Вторая итерация

Значения ЧП в точке спуска равны:

$$\frac{\partial L_2}{\partial w_2} = \frac{\partial L_1}{\partial w_{02}} = 0, \frac{\partial L_2}{\partial w_1} = -20, \frac{\partial L_2}{\partial w_{01}} = -12$$



Новая точка спуска равна:

$w_1=2, w_{01}=1.2, w_2=2, w_{02}=1.2$

и так далее...

Выход сети с дропаутом

Когда НС будет натренирована с помощью дропаута, и мы захотим проверить предсказание сети на новом объекте *A*, то надо не забыть следующее:

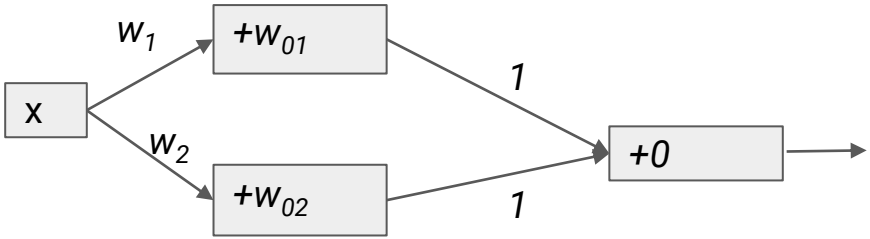
выход каждого нейрона домножается на число *p* (вероятность исключения этого нейрона).

Продемонстрируем это правило на примере.

Пример

Если нейроны внутреннего слоя будут попеременно исключаться из-за дропаута, то веса сойдутся к значениям:

$$F_{NN}(x)=(-x+4)+(-x+4)$$



(из-за дропаута каждый нейрон пытается в одиночку восстановить зависимость).

x	Y
1	3
2	2
3	1

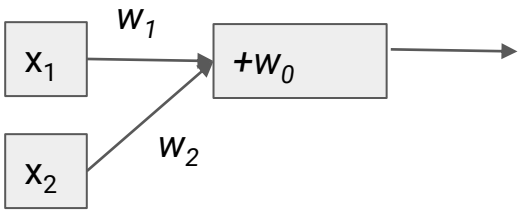
Поскольку нейроны исключались с вероятностью 0.5, то окончательная функция НС равна:

$$F_{NN}(x)=0.5(-x+4)+0.5(-x+4)=-x+4$$ – **и это правильный ответ**!

Дропаут на входном слое

На входном слое тоже можно (и нужно) применять дропаут.
Фактически – это **выбор случайного подмножества признаков для обучения** на каждой итерации ГС.

Для примера рассмотрим НС с дропаутами на входном слое.



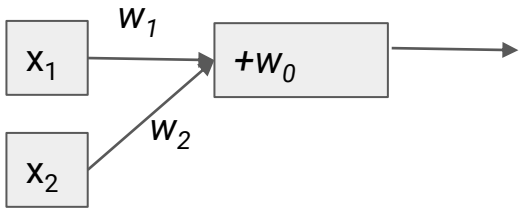
x_1	x_2	Y
0	0	0
0	1	1
1	0	1
1	1	2

Шаг обучения *h=0.1*, начальное значение всех весов *0*.
Вероятность исключения нейрона входного слоя 0.5

Дропаут на входном слое

Функция сети (обычная):

$F_{NN}(x_1,x_2)=w_1 x_1+w_2 x_2+w_0$

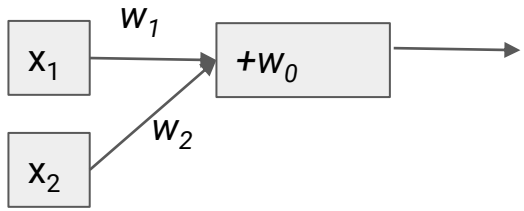


Функция потерь (обычная): $L(w)=(F_{NN}(0,0)-0)^2+(F_{NN}(0,1)-1)^2+(F_{NN}(1,0)-1)^2+(F_{NN}(1,1)-2)^2$

x_1	x_2	Y
0	0	0
0	1	1
1	0	1
1	1	2

Дропаут на входном слое

Если дропаут **выкидывает вход** x_1, то функция сети и функция потерь равны:



$$F_{NN1}(x_1,x_2)=w_2x_2+w_0$$

$$L_1(w)=(w_0-0)^2+(w_2+w_0-1)^2+(w_0-1)^2+(w_2+w_0-2)^2$$

Если дропаут **выкидывает вход** x_2, то функция сети и функция потерь равны:

$$F_{NN2}(x_1,x_2)=w_1x_1+w_0$$

$$L_2(w)=(w_0-0)^2+(w_1+w_0-1)^2+(w_0-1)^2+(w_1+w_0-2)^2$$

x_1	x_2	Y
0	0	0
0	1	1
1	0	1
1	1	2

Дропаут на входном слое

А дальше ГС на каждой итерации минимизирует либо функцию потерь $L_1(w)$ либо $L_2(w)$ (в зависимости от того, какой вход удаляется).

Идея тренировать модель машинного обучения по **случайному подмножеству признаков** не нова. Это применяется в классической модели Random Forest.

Онлайн-курс

ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

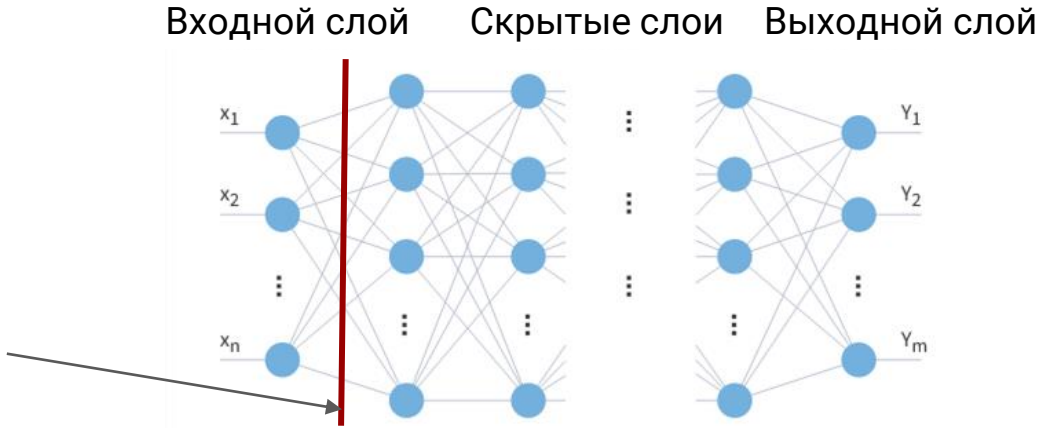
8. Дропаут и нормализация по мини-батчам

Нормализация данных во внутренних слоях сети

Нормализация входов НС

В предыдущих лекциях говорилось, что данные для входа НС нужно **нормализовать** (привести к одному масштабу).

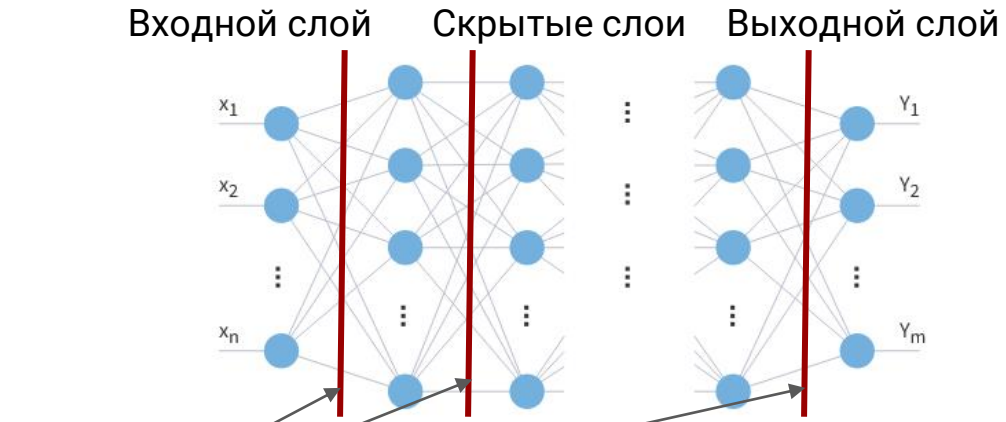
Иными словами, данные, передаваемые между входным слоем и первым внутренним слоем **нормализованы**.



Нормализация входов НС

А почему бы не нормализовать данные, передаваемые между **каждой парой соседних слоев?**

А зачем это нужно?



Через эти связи передаются нормализованные данные

Доводы в пользу тотальной нормализации

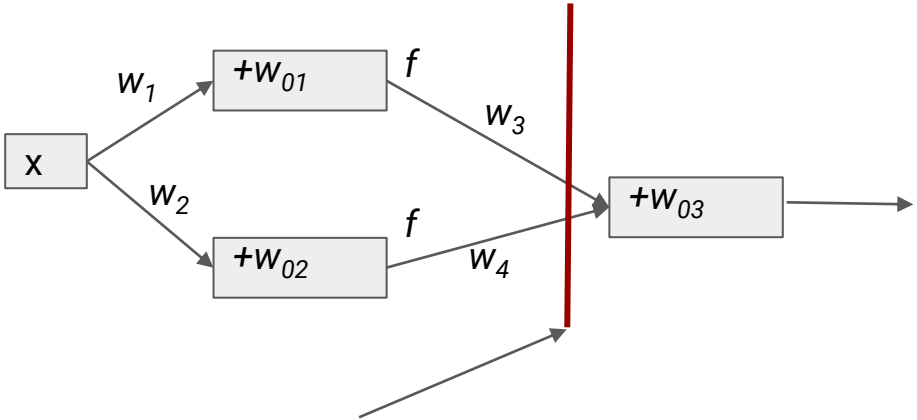
1. Ну а какая принципиальная разница между входным слоем и внутренними слоями сети? Первый внутренний слой можно считать входным слоем при преобразованных данных.

1. Это следует из инициализации Ксавье. Перед ГС веса генерируются из **единого интервала** *[-a,a]*. Следовательно, сигнал, передаваемый по всем связям должен быть одного масштаба. Иначе НС будет долго обучаться, ей потребуется много времени, чтобы самой подогнать веса к масштабу данных.

И как это делать?

Общее правило:

нужно нормализовать данные, входящие в каждый слой НС.



нормализировать данные нужно здесь
(мы предполагаем, что данные входного слоя нормализованы)

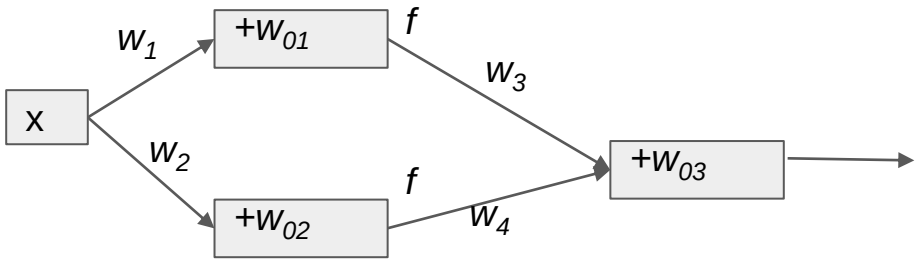
Рассмотрим на **конкретном** примере, какие формулы здесь возникают.

Пример

Имеем НС с функцией $F_{NN}(x)=f(w_1x+w_{01})w_3+f(w_2x+w_{02})w_4+w_{03}$

Функция потерь: $L(w)=(F_{NN}(-1)+1)^2+(F_{NN}(1)-1)^2$

ТВ (данные уже нормализованы, у них среднее 0, отклонение 1):



x	Y
-1	-1
1	1

Пример

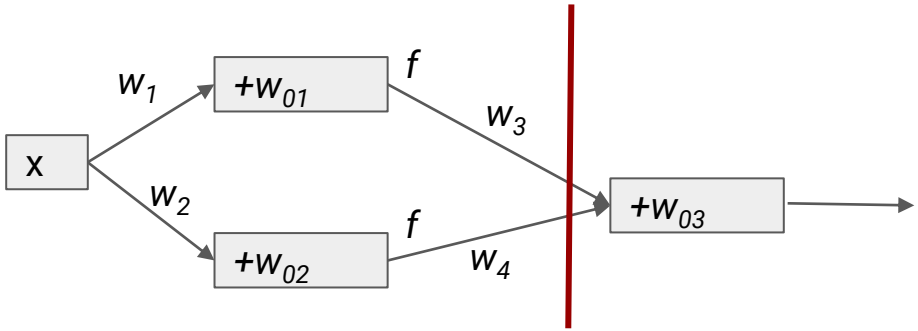
Считаем (**в общем виде**) вход в последний слой для объектов из ТВ.

Первый объект дает вход: $A=f(-w_1+w_{01})w_3+f(-w_2+w_{02})w_4$

Второй объект даёт вход: $B=f(w_1+w_{01})w_3+f(w_2+w_{02})w_4$

Обозначим $C=(A+B)/2$ – среднее значение,

$S = \sqrt{((A-C)^2+(B-C)^2)/2}$ отклонение



x	Y
-1	-1
1	1

Формула для отклонения

Ранее мы считали отклонение по формуле

$$S = \sqrt{1/(n-1)[()^2 + \ldots + ()^2]}$$

но в этой лекции мы будем использовать формулу

$$S = \sqrt{1/n[()^2 + \ldots + ()^2]}$$

Многие формулы упростятся. Замена одной формулы на другую по сути не сильно меняет ответы.

Пример

Теперь сигнал, входящий в последний нейрон нужно нормализовать с использованием среднего и отклонения по ТВ.

В нашей НС в этот нейрон входит сигнал $f(w_1 x+w_{01})w_3+f(w_2 x+w_{02})w_4$,

а теперь будет входить $(f(w_1x+w_{01})w_3+f(w_2x+w_{02})w_4-C)/S$.

x	Y
-1	-1
1	1

Это приводит к изменению функции сети:

$$F_{NNN}(x)=(f(w_1x+w_{01})w_3+f(w_2x+w_{02})w_4-C)/S+w_{03}$$

и к получению новой функции потерь

$$L_N(w)=(F_{NNN}(-1)+1)^2+(F_{NNN}(1)-1)^2$$

вот **её и надо минимизировать**.

Если слой состоит из многих нейронов,

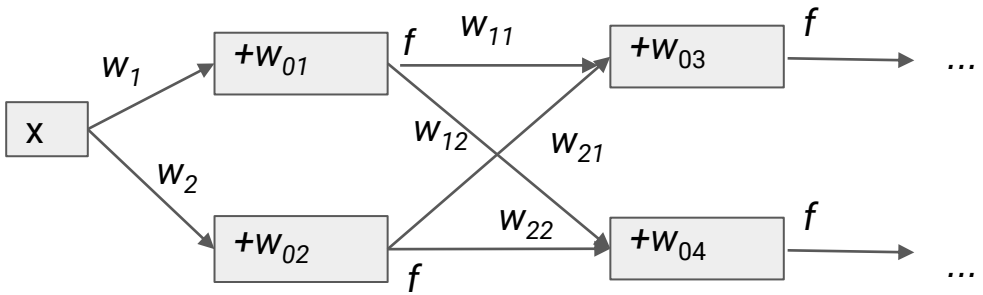
… то вход каждого нейрона нормируется по ТВ.

Берем верхний нейрон правого слоя. Если объекты ТВ прогнать через НС, то придут два значения:

$$A=f(-w_1+w_{01})w_{11}+f(-w_2+w_{02})w_{21}$$

$$B=f(w_1+w_{01})w_{11}+f(w_2+w_{02})w_{21}$$

x	Y
-1	-1
1	1



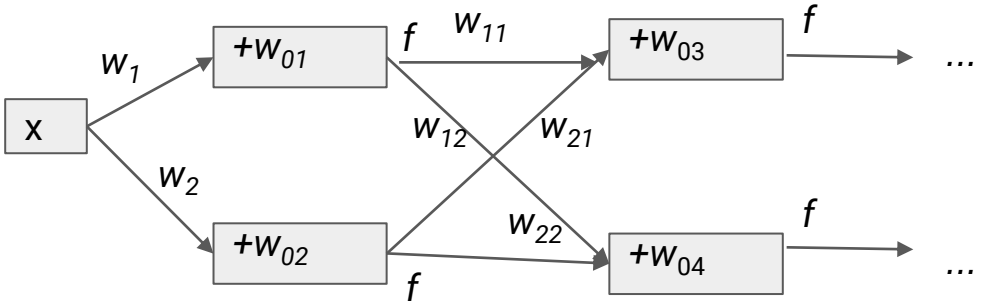
Если слой состоит из многих нейронов

Далее для выражений A, B считаем среднее *C* и отклонение *S*.
Тогда выход из верхнего нейрона правого слоя будет равен:

$$f((f(xw_1+w_{01})w_{11}+f(xw_2+w_{02})w_{21}-C)/S+w_{03})$$

x	Y
-1	-1
1	1

Аналогично по ТВ нормируется вход во второй нейрон из правого слоя (там будут свои выражения для среднего и отклонения).
Всё это приводит к модификации функции сети.



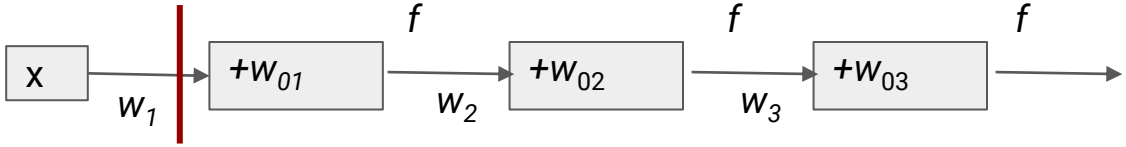
Всё очень сложно

Нормализацию по ТВ нужно проводить для всех слоев НС. При этом выражение для $F_{NN}(x)$ многократно усложняется.

Сейчас я **сделаю вам больно** на примере следующей простой НС:

x	Y
-1	-1
1	1

Нужно нормировать вход из *x* в первый нейрон:



Всё очень сложно

Входы в нейрон для объектов ТВ равны: $-w_1$, w_1. Среднее значение =0, отклонение равно w_1. Следовательно, нормированный вход в нейрон равен:

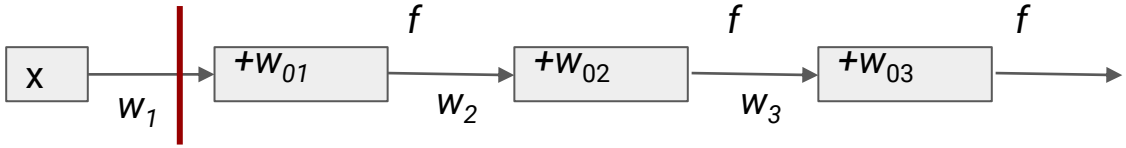
$$(w_1x-0)/w_1=x$$

x	Y
-1	-1
1	1

Тогда вход в следующий нейрон вычисляется по формуле:

$$w_2f(x+w_{01})=w_2f(x+w_{01})$$

Теперь будем нормировать вход в следующий нейрон



Всё очень сложно

x	Y
-1	-1
1	1

Вход в нейрон вычисляется по формуле:

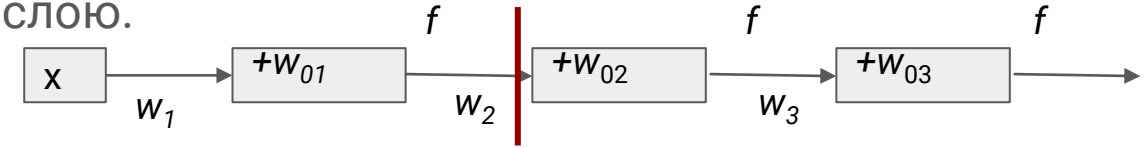
$w_2 f(x+w_{01})=w_2 f(x+w_{01})$

Тогда для объектов ТВ мы получим значения: $w_2 f(-1+w_{01})$, $w_2 f(1+w_{01})$.

Вычисляем их среднее и отклонение: $C_1=(w_2 f(-1+w_{01})+w_2 f(1+w_{01}))/2$,

$$S_1 = \sqrt{((w_2 f(-1+w_{01}) - C_1)^2 + (w_2 f(1+w_{01}) - C_1)^2)/2}$$

нормируем вход в нейрон: $(w_2 f(x+w_{01})-C_1)/S_1$ и переходим к следующему слою.



Всё очень сложно

Вход в нейрон вычисляется по формуле: $w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})$

Тогда для объектов ТВ мы получим значения:

x	Y
-1	-1
1	1

$w_3 f([(w_2 f(-1+w_{01})-C_1)/S_1]+w_{02})$,

$w_3 f([(w_2 f(1+w_{01})-C_1)/S_1]+w_{02})$

Вычисляем их среднее и отклонение:

$C_2=(w_3 f([(w_2 f(-1+w_{01})-C_1)/S_1]+w_{02})+w_3 f([(w_2 f(1+w_{01})-C_1)/S_1]+w_{02}))/2$

S_2=[**18+: выражение содержит шокирующий контент**].

Всё очень сложно

x	Y
-1	-1
1	1

Вход в нейрон вычисляется по формуле: $w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})$

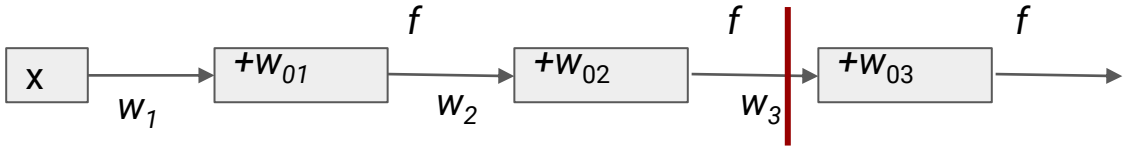
Тогда для объектов ТВ мы получим значения:

$w_3 f([(w_2 f(-1+w_{01})-C_1)/S_1]+w_{02})$, $w_3 f([(w_2 f(1+w_{01})-C_1)/S_1]+w_{02})$

Нормируем вход в нейрон:

$(w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})-C_2)/S_2$

и только теперь можно выписать выражение для выхода НС...



Всё очень сложно

Нормализация данных в каждом слое НС приводит к изменению функции всей сети. Теперь $F_{NN}(x)$ равна:

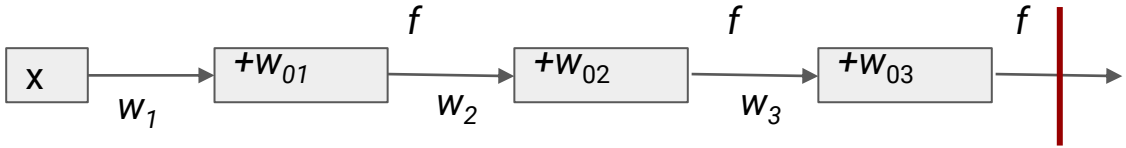
x	Y
-1	-1
1	1

$$F_{NNN}(x)=f([(w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})-C_2)/S_2]+w_{03})$$

Соответственно нужно минимизировать более сложную функцию потерь:

$$L_N(w)=(F_{NNN}(-1)+1)^2+(F_{NNN}(1)-1)^2 .$$

Также не нужно забывать, что **выражения C_1,C_2,S_1,S_2 зависят от весов НС**, поэтому вычислить градиент функции потерь становится сложнее.



Онлайн-курс

ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

8. Дропаут и нормализация по мини-батчам

Тонкая настройка нормализации

Что за магические константы!

Почему мы нормализуем данные, делая их среднее значение равным 0 и отклонение равным 1? **Почему числа 0 и 1 такие особенные**?

Это обычные числа, поэтому можно нормализовать данные к другим значениям – если сильно хочется. Но к каким?

А вот пусть НС сама разберётся! Она же умная.

Продвинутая нормализация

Введем два **настраиваемых** параметра НС: *c,s*, и входы каждого нейрона в сети будут нормироваться «средним» *c* и «отклонением» *s*.

Параметры *c,s* попадут в выражение для $F_{NN}(x)$ и в выражение для функции потерь L(w).

Когда будет найдена точка минимума функции потерь, **мы получим оптимальные значения** *c,s*. И мы поймём, какое среднее и отклонение в данных оптимально для нашей задачи.

Это неформальное объяснение. Более детально см. след. слайды.

Коэффициенты корректировки

Пусть x' - отнормированное входное значение нейрона. Выражение x' участвует в $F_{NNN}(x)$ и функции потерь.

А теперь давайте на вход слою НС подавать не x', а выражение $ax'+b$, где a,b - новые параметры сети, оптимальное значение которых будет искаться во время обучения.

Справка: если среднее значение и отклонение величины x' были равны 0 и 1, то среднее значение и отклонение величины $ax'+b$ будут равны b и a соответственно.

Модифицируем изученный ранее пример с использованием параметров a,b.

Старый пример

x	Y
-1	-1
1	1

Для нашей НС ранее была найдена функция сети:

$$F_{NN}(x)=f([(w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})-C_2)/S_2]+w_{03})$$

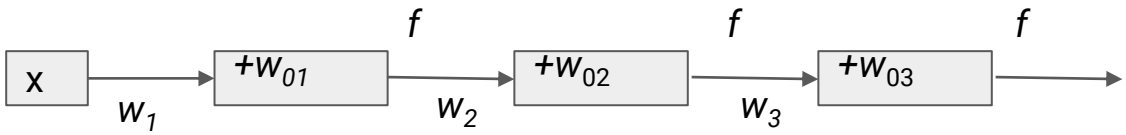
причем входные значения в каждый слой равны:

x

$$(w_2 f(x+w_{01})-C_1)/S_1$$

$$(w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})-C_2)/S_2$$

Навесим на них параметры *a,b* и получим….



Старый пример

x	Y
-1	-1
1	1

$ax+b$

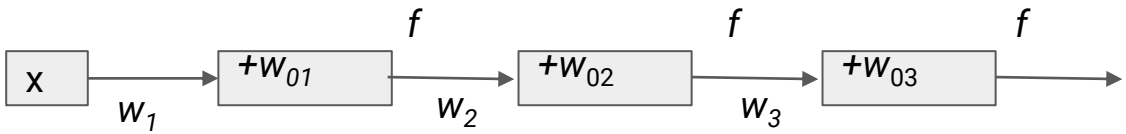
$a(w_2 f(ax+b+w_{01})-C_1)/S_1+b$

$a(w_3 f([a(w_2 f(ax+b+w_{01})-C_1)/S_1+b]+w_{02})-C_2)/S_2+b$

Тогда функция сети равна $F_{NNN}(x)=f([a(w_3 f([a(w_2 f(ax+b+w_{01})-C_1)/S_1+b]+w_{02})-C_2)/S_2+b]+w_{03})$

эта функция будет участвовать в выражении для функции потерь $L(w)$.

Мы минимизируем $L_N(w)$, и значение a,b в точке минимума даст нам оптимальные значения для коэффициентов нормализаци



Старый пример

$$F_{NNN}(x)=f([a(w_3 f([a(w_2 f(ax+b+w_{01})-C_1)/S_1+b]+w_{02})-C_2)/S_2+b]+w_{03})$$

эта функция будет участвовать в выражении для функции потерь $L_N(w)$.

Мы минимизируем $L(w)$ и значения **a**,**b** в точке минимума дают нам оптимальные значения для коэффициентов нормализации.

Кстати, если модель решит, что значения a=1, b=0 для нее оптимальны, **то мы вернемся к стандартной нормализации** со средним 0 и отклонением 1.

Более сложная модель

Сейчас числа *a,b* одни на всю сеть. А можно свои коэффициенты *a,b* завести для каждого слоя НС (или даже для каждого нейрона).

Было:	Стало:
$ax+b$	a_1x+b_1
$a(w_2f(ax+b+w_{01})-C_1)/S_1+b$	$a_2(w_2f(a_1x+b_1+w_{01})-C_1)/S_1+b_2$
$a(w_3f([a(w_2f(ax+b+w_{01})-C_1)/S_1+b]+w_{02})-C_2)/S_2+b$	$a_3(w_3f([a_2(w_2f(a_1x+b_1+w_{01})-C_1)/S_1+b_2]+w_{02})-C_2)/S_2+b_3$
$F_{NNN}(x)=f([a(w_3f([a(w_2f(ax+b+w_{01})-C_1)/S_1+b]+w_{02})-C_2)/S_2+b]+w_{03})$	$F_{NNN}(x)=f(a_3(w_3f([a_2(w_2f(a_1x+b_1+w_{01})-C_1)/S_1+b_2]+w_{02})-C_2)/S_2+b_3+w_{03})$

Онлайн-курс

ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

8. Дропаут и нормализация по мини-батчам

Нормализация по мини-батчам

Проблемка

В предыдущих примерах было показано, что нормализация данных во всех слоях НС приводит к **сильному усложнению** вида функции потерь.

Более того, $L_N(w)$ содержит выражения для среднего и отклонения по объектам ТВ. Эти выражения являются суммами с числом слагаемых равным объёму ТВ.

И эти выражения **встречаются в $L_N(w)$ много раз**.

Если ТВ огромная, то вычислить градиент функции потерь **чрезвычайно трудоемко**.

Это напоминает одну проблему из статистики...

Выборочный метод в статистике

Допустим, вам нужно вычислить **средний рост людей на планете.**
Измерять абсолютно всех – трудоемко и нереализуемо.

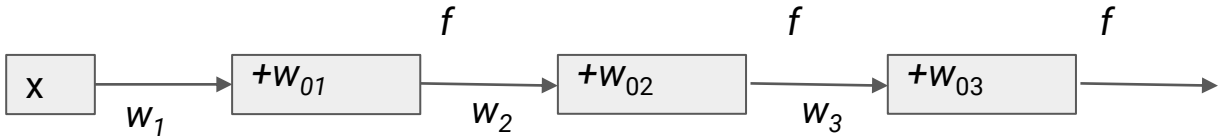
Альтернатива: выборочный метод.
Нужно измерить небольшую группу **случайно выбранных** людей,
и их средний рост **не будет сильно отличаться** от среднего роста всего человечества.

Идея
Если трудно считать среднее и отклонение по всей ТВ, то можно взять небольшой набор случайно выбранных объектов из ТВ и подсчитать среднее и отклонение по ним.

Такой случайно выбранный наборчик объектов называется мини-батчем.

Как это работает?

Давайте возьмём старый пример и допустим, что у нас уже большая ТВ



x	Y
-1	-1
1	1
2	2
...	...

Как провести одну итерацию ГС?

Случайно выбираем мини-батч (размер мини-батча – это гиперпараметр). Допустим, что **совершенно случайно** (рояль в кустах!) **были выбраны первые два объекта ТВ.**

Это означает, что при нормализации мы забываем про остальную выборку.

Как это работает

У нас получатся **абсолютно такие же формулы**, что и в старом примере. В частности, получим:

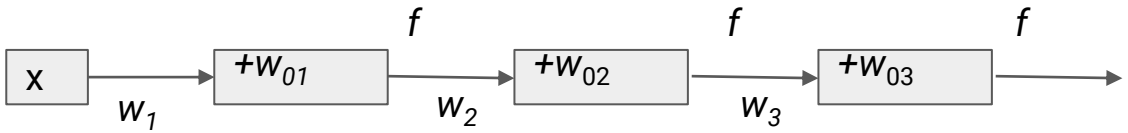
$$F_{NNN}(x)=f([(w_3 f([(w_2 f(x+w_{01})-C_1)/S_1]+w_{02})-C_2)/S_2]+w_{03})$$

А вот функция потерь использует уже все объекты из ТВ:

$$L_N(w)=(F_{NNN}(-1)+1)^2+(F_{NNN}(1)-1)^2+(F_{NNN}(2)-2)^2+\ldots$$

А на следующей итерации ГС будет выбран уже другой мини-батч, функция $F_{NNN}(x)$ на следующей итерации ГС изменится.

x	Y
-1	-1
1	1
2	2
...	...



К чему в итоге пришли?

Нормализация по мини-батчам оказалось супер-успешной технологией для тренировки больших НС.

В большинстве ситуаций если используется **батч-нормализация**, то можно не применять ни **дропаут** ни **регуляризацию**.

К чему в итоге пришли?

В общем, современные нейронные сети – это:

L – **loss function** (функция потерь);

G – **gradient descent** (градиентный спуск);

B – **batch-normalization** (батч-нормализация);

T – **training set** (тренировочная выборка).

Онлайн-курс

ВВЕДЕНИЕ В ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

8. Дропаут и нормализация по мини-батчам

Выводы

Выводы:

- Мы познакомились с процедурой дропаута. Она позволяет представить НС в виде ансамбля нескольких НС.
- Мы поняли, что передачу данных между слоями НС нужно нормировать. Для сокращения вычислений нормировать нужно не по всей ТВ, а по мини-батчам.