Свёрточные автокодировщики для улучшения качества классификации изображений

Медведев Алексей

Введение

В этой работе мы рассмотрим многослойные нейронные сети и их применения для классификации и обработки изображений. Изображения возьмем из датасета stl-10, приведя их к размеру 32×32, а значения пикселей отобразим на отрезок $[-1, 1]$.

Эксперимент 1

Однослойная нейросеть, реализующая мультиномиальную логистическую регрессию(один полносвязный слой и функция потерь CrossEntropyLoss) показала точность 36%. При этом ансамбль из 1000 деревьев(RandomForest) показал результат 43%.

Эксперимент 2

Значит ли это что нейронные сети сильно уступают другим алгоритмам в решении этой задачи? Попробуем применить свертку в нашей нейросети, подберем ее параметры на валидационной выборке и посмотрим улучшиться ли результат.

Изменять будем следующие гиперпараметры:

- размер ядра свертки;
- количество фильтров на слое;
- количество сверточных блоков.

Визуализируем результаты нашего подбора параметров для первого сверточного блока.

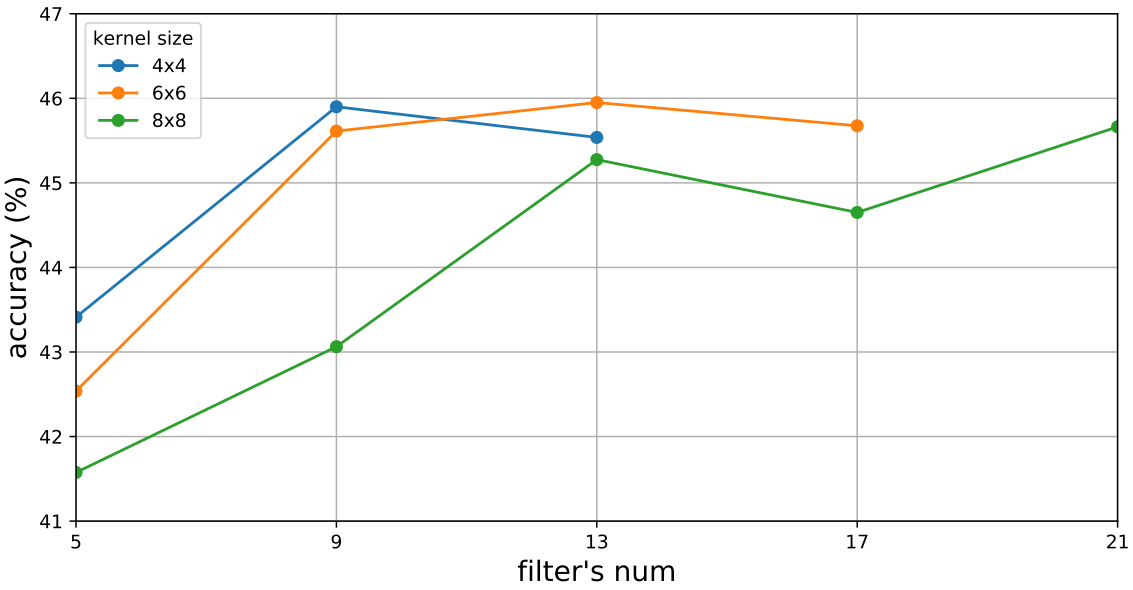

Рис. 1: Зависимость точности от размеров ядра свертки и количества фильтров на слое.

По графику построим таблицу, отобразив худший и лучший набор параметров для фиксированного размера ядра.

kernel size	filter num	conv out size	accuracy
4×4	5	980	43.41
4×4	9	1764	45.90
6×6	5	845	42.53
6×6	9	1521	45.95
8×8	5	720	41.57
8×8	21	3024	45.66

Таблица 1: Точность сети в зависимости от параметров

Из нашей визуализации следует, что необходимо искать баланс в размерах сжатия. Слишком большое сжатие приводит к потере важных признаков, а слишком слабое наоборот оставляет шумовые признаки. При поиске гиперпараметров нельзя ориентироваться только на размер выхода сверточного слоя, например для двух ядер разных размеров можно подобрать такое количество фильтров, что размер выхода у слоев получится почти одинаковым, но точность будет разной.

Подберем таким же образом параметры всех последующих слоев. Посмотрим как количество слоев влияет на время и качество работы сети.

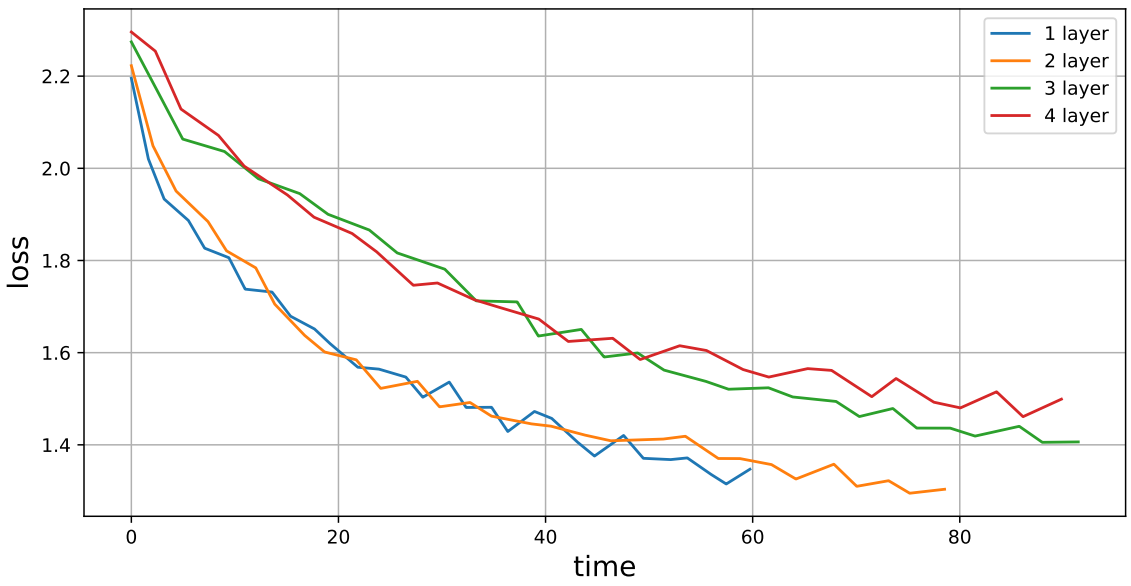

Рис. 2: Зависимость точности от размеров ядра свертки и количества фильтров на слое.

При этом при двух слоях качество вырастает до 48.5%, а при дальнейшем увеличении количества слоев начинает уменьшаться. Вероятно это происходит из-за отсеивания или зашумления важных признаков, при плавном уменьшении размеров входа полносвязного слоя. Также это может быть связано с недообучением сети(хотя количество эпох и было увеличено в разы), т.к на графике видно, что функционал ошибки у сети с увеличением слоев убывает все медленнее.

Рассмотрим теперь зависимость времени и качества обучения сети от такого параметра нашего оптимизационного алгоритма, как momentum. Этот параметр определяет влияние на текущий градиент градиентов, вычисленных на предыдущих итерациях(точнее количество предыдущих итераций, которые будут иметь вес на текущем шаге).

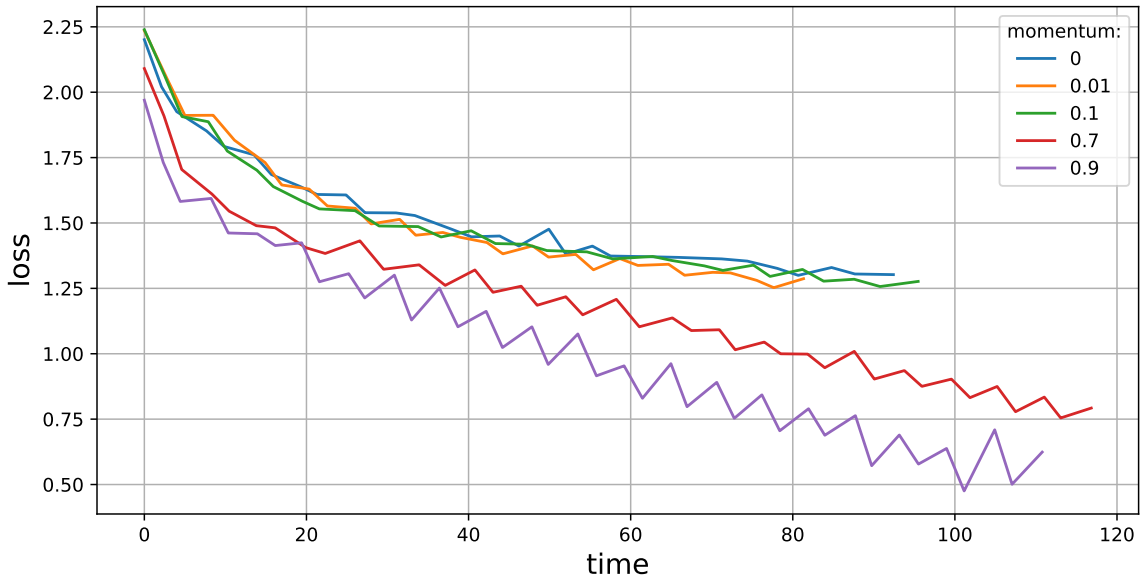

Рис. 3: Зависимость точности от размеров ядра свертки и количества фильтров на слое.

Из графика следует, что momentum сильно влияет на скорость сходимости к оптимуму, а так же на гладкость функционала. Но как видно из таблицы, лучше оставить momentum равным нулю.

momentum	accuracy
0	48.67
0.01	48.21
0.1	47.18
0.7	47.09
0.9	41.33

Таблица 2: Зависимость точности от размеров ядра свертки и количества фильтров на слое.

Эксперимент 3

По результатам предыдущего эксперимента видно, что правильный подбор параметров существенно увеличил точность алгоритма, реализуемого нашей сетью. Но есть и другой способ увеличить качество классификации. Воспользуемся автокодировщиком, специальным вычислительным графом, реализующим тождественную функцию. Идея состоит в том, что автокодировщик постепенно сворачивает изображение, а затем пытается восстановить. Таким образом мы сможем выделить наиболее важные признаки, перейти в признаковое пространство меньшей размерности.

Из таблицы можно увидеть какие параметры сильнее уменьшают ошибку, прочерками отмечены параметры, оставленные без изменений. Итоговая ошибка составила всего 0.0214. Попробуем визуализировать приближенные автокодировщиком изображения и их прообразы для наглядности и понимания того, что стоит за этой цифрой.

num	conv-out-channels	conv-kernel-size	conv-padding	momentum	MSE
1	6	2	0	0	0.1494
2	-	3	-	-	0.1137
3!	-	4	-	-	0.0922
4	-	5	-	-	0.1126
5	7	4	-	-	0.0917
6	-	-	1	-	0.0705
7	-	-	2	-	0.0628
8	-	-	3	-	0.0570
9	-	8	6	-	0.0451
10	-	-	-	0.7	0.0310
11	-	-	-	0.95	0.0214

Таблица 3: Ошибка атокодировщика в зависимости от параметров

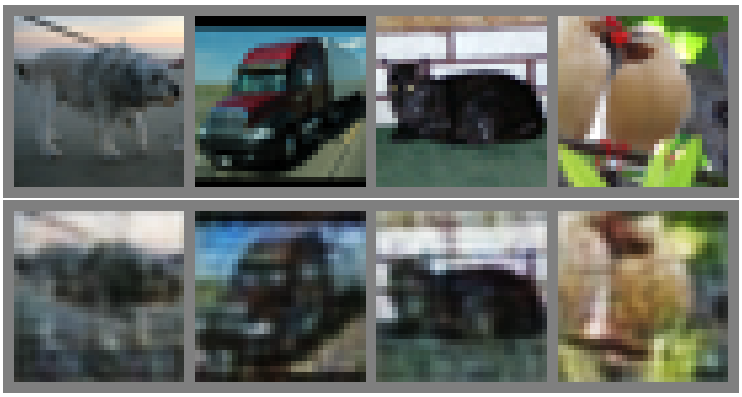

Рис. 4: Восстановленные и оригинальные изображения.

Эксперименты 4 и 5

Посмотрим как изменится результат первого эксперимента при переходе в новое признаковое пространство. Качество логистической регрессии выросло до 45%, в то время как для деревьев кчество повысилось лишь до 48%. Скорее всего это связано с тем, что деревья сами умеют выявлять нелинейные зависимости, в отличие от логистической регрессии.

Подобрав параметры в третьем эксперименте, мы теперь можем использовать автокодировщик для нашей первоначальной цели. Попробуем два способа применения получившейся сети: просто подавать выход промежуточного блока автокодировщика на вход нашей сети(параметры которой мы подберем), сделать сверточные блоки нашего автокодировщика частью сети(в этом случае сам автокодировщик будет тоже обучаться вместе со всей сетью).

Опустив подробности подбора параметров для новых сетей сразу проанализируем результаты. В первом случае точность составила 49.95%, а во втором целых 51.5%. Получается, что неразмеченные данные могут тоже быть полезными и значительно улучшить качество классификации.