Алимов Нурислам

Задача

Провести сравнительных анализ одного из стандартных методов обработки изображений, применяемых к задачам классификации изображений с методов, где применялись бы свёрточные нейронные сети.

Данные

CIFAR-10

https://www.cs.toronto.edu/~kriz/cifar.html

В TensorFlow использовался бинарный формат.

С применением opencv использовался https://www.kaggle.com/c/cifar-10/, с форматом png.

Данные представляют собой 50 тысяч картинок в тренировочном наборе и 300 тысяч картинок в тестовом наборе.

10 классов:
- лошади
- автомобили
- олени
- собаки
- лягушки
- коты
- грузовики
- корабли
- птицы
- самолёты

Машинное обучение и дескрипторы

Существует несколько стандартных способов извлекать фичи из изображений, в основном применяются дескрипторы SIFT, SURF, HOG, GLOH.

В этой работе применялся SIFT.

Применялась стандартная функция из библиотеки opencv. В итоге каждая картинка характеризуется матрицей, где в столбцах хранятся особые точки, а каждый столбец представляет собой вектор с 128 компонентами дескриптора.

Здесь возник вопрос, для дальнейшего применения методов машинного обучения нам нужны вектора, а не матрицы. Соотвественно необходимо как-то превратить матрицы в вектора.

В этой работе было принято решение, получить вектора, путём поиска среднего значения по каждой компоненте дескриптора SIFT среди всех особых точек, найденных алгоритмом. В итоге каждому изображению соответствует вектор размерности 1x128.

В качестве алгоритма машинного обучения был выбрал RandomForest из библиотеки scikit-learn.

Также была идея применить: метод Bags of words (прочитал несколько статей, где такое применялось) - стоит ли применять? В тестировании не хватило оперативной памяти, получаются вектора размерности 1x4736.

Сверточные нейронные сети

Используем Tensor Flow.

Видеокарта Geforce 750 Ti.

CUDA 8.0, cuDNN 5.1

Схема используемой сети:

conv1 => pool1 => norm1 => conv2 => norm2 => pool2 => local3 => local4 => softmax_linear

conv - свёрточный слой
norm - нормализация
pool - уменьшение размерности
local - полная сборка выходов предыдущего слоя (нейрон с линейной функцией активации)

Соотвественно, в этой сети 2 свёрточных слоя, на выходе решение о принадлежности какому-то классу принимается на основе пороговой функции.

Обучение производилось методом обратного распространения ошибки, оптимизация весов градиентным спуском.

Результаты сравнительного анализа

Время обучения:

tensor flow: 50 тысяч итераций - 3 часа
opencv: 15 минут

Точность:

tensor: precision = 0.81
opencv: precision = 0.2, f1 = 0.2
random choose: 0.1 соответственно при 10 классовой классификации

Что можно сделать еще

Попробовать использовать более сложные нейронные сети: VGG16, VGG19.
Попробовать использовать caffe, оценить точность, скорость обучения.

Подумать как улучшить качество классификации с применением дескрипторов.

Подумать как применить метод Виолы - Джонса в задаче классификации.

Рассмотреть другие методы извлечения признаков из изображений.

Исходные коды: https://github.com/nsmalimov/nuralnet_opencv_classif