# Алгоритм распознавание жестов

## *Описание задачи*

Для распознавания жестов мы применить подход, основанный на использовании рекуррентной нейронной сети с долгой краткосрочной памятью (Long short-term memory, LSTM). Данный вид сетей успешно приминается при решении схожей задачи распознавания движений руки по показаниям акселерометра «умных» часов.

## *Исходные данные*

Обучающая выборка представляет собой совокупность потоков данных, каждый их которых содержит серию жестов, совершенных одним пользователем.

Каждый элемент потока содержит в себе:

1. прошедшее время с момента начала записи;
2. вектор скорость движение;
3. поток кадров (пока не знаю размер кадра)

## *Структура нейронной сети*

Предлагаемая архитектура сети, размеры слоев и функции активации изображены на рис. Ниже. Размер последнего слоя совпадает с числом распознаваемых жестов.

Результатом работы нейронной сети является вероятный жест, где — число жестов.

В качестве алгоритма оптимизации весов сети можно использовать разновидность стохастического градиентного спуска — алгоритм Adagrad (есть и другие, но пока понял этот лучше всего).

## *Adagrad*

Когда мы обучаем нейронную сеть с миллионами параметров, это не очень хорошо. Какие-то параметры будут меняться чаще, какие-то реже. Интуитивно кажется, что параметры, которые меняются редко, надо менять с большей скоростью, чем те, которые меняются часто. В работе предложен алгоритм, который подстраивает скорость обучения каждого из параметров в зависимости от того как сильно данные параметры менялись ранее.

Вместо единого скалярав качестве скорости обучения, на каждой итерации будем определять вектор . В данном случае η, которую мы использовали ранее будет просто начальной скоростью обучения. Для первой итерации мы положим

Итак, для каждой итерации у нас есть текущий вектор параметров:

,

и вектор градиента:

Определим вспомогательный вектор:

у которого каждый элемент есть сумма квадратов частных производных функции ошибки по соответствующему параметру, т.е.

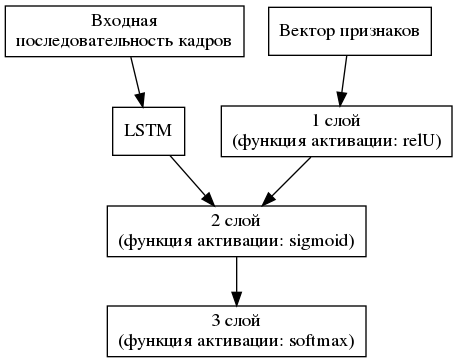
и, наконец, компоненты вектора скорости обучения определим как:

, - параметр страхующий от деления на ноль (~).

Изменение параметров осуществляем практически по той же формуле, что и для стохастического градиентного спуска:

Функция потерь:

Структура нейронной сети



Функции активации подберу таким образом, чтобы функция потерь была минимальна.

## *Постановка эксперимента*

Для построения обучающей выборки будет использовать следующий сценарий. Участник обучающей группы стоит рядом с платой, в которой уже прошита наша программа. Далее я включаю запись и прошу участника совершить жест. Просим его повторить жест аналогичным образом участник, но только под разными углами (или просто повторить - посмотрим как будет лучше). Аналогично со всеми участник проводим те же процедуры и сохраняем результаты в какой-нибудь базе данных.

Сначала попробуем простые жесты:

1. Влево-вправо
2. Вверх-вниз
3. Круговые движения по часовой и против
4. По диагонали

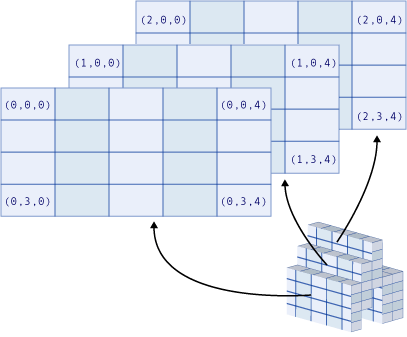
Далее попробуем увеличить тестовый набор жеста вращением, растяжением и сжатием. И если алгоритм в качестве результата выдаст не тот жест, который совершил участник эксперимента, назовем такой жест ошибочным.

И еще, пока доступа к плате у меня нет. Думаю записывать на веб-камеру жест, делать кадры монохромными и на них распознавать жест. Потом несложно будет переделать.

# *Мои размышления*

Вот как я примерно представляю себе работу диплома. Пока не очень понимаю как работают рекуррентные сети, какую структуру лучше использовать и куча всего. Ну думаю, что разберусь =)

На работе от меня требуют простой алгоритм, который сохраняет жест в виде трехмерного массива. Потом простить повторить пока он в точности не сделает тот же жест.

время ширина и высота кадра

Но так совсем не интересно и для диплома не годится. Думаю лучше сетей ничего не придумать.

Самое обидное — это, если я все напишу на питоне, но не смогу потом переписать на си = (.