На данный момент уже рассмотрено и применено множество подходов к задаче идентификации диктора. В основном алгоритм работы заключается в нескольких этапах:

1) Сбор базы данных голосовых дорожек.

2) Обработка и фильтрация аудиофайлов.

3) Выделение вектора признаков.

4) Обучение классификатора на полученных данных.

Для каждого из этапов есть различные пути реализации. Первый этап зависит от помещения, в котором проводится запись эталонных примеров, от приборов, которые используются для записи. Например, при использовании голоса, передаваемого по телефонному каналу, в общем случае нельзя гарантировать использование одного и того же микрофона и канала передачи, кроме того необходимо учитывать влияние посторонних помех. Использование более качественной записи возможно, например, в приложении для контроля доступа к помещению. В таком случае канал представляет собой микрофон, его кабель и аналого-цифровой преобразователь. Хранение аудиофайлов может так же храниться в множестве расширений, таких как wav, wma, mp3, flac, 3gp и другие, от этого так же зависят дальнейшие преобразования.

На втором этапе происходит очистка звука от шумов. Так как известно, что нужно очистить звуковой файл, основой которого является человеческий голос, в большинстве случаев применяют один из четырех фильтров:

1) Lowpass/Lowshelf

2) Highpass/Highshelf

3) Bandpass/Bandshelf

4) Notch/Peaking

Для каждого из этих фильтров используются входные данные: ограничивающие частоты (либо частоты, подлежащие удалению в случае последнего фильтра), частота дискретизации сигнала и порядок фильтра. На выходе получается отфильтрованный звуковой сигнал.

Основной упор делается на третий этап - выделение признаков из обработанного сигнала.

Теория.

Про нейронные сети.

Принцип свертки.

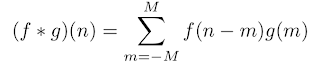
Сверточные нейронные сети были придуманы таким образом, чтобы для них была возможна работа с пространственными структурами. Идея СНС возникла благодаря зрительной системе мышей, и изначальной целью их создания была работа с изображениями. По сравнению с обычными полносвязными нейронными сетями, СНС имеют гораздо меньше параметров, что позволяет эффективно обучать очень глубокие архитектуры (обычно более 5 уровней, что практически невозможно для полносвязных сетей). Большинство слоев СНС используют операцию свертки. В непрерывном случае свертка двух функций f и g определяется следующим образом:



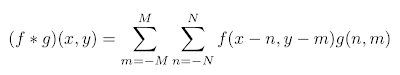
В дискретном случае интеграл заменяется суммой:



В случае, когда функция g определена от [-M;M]:

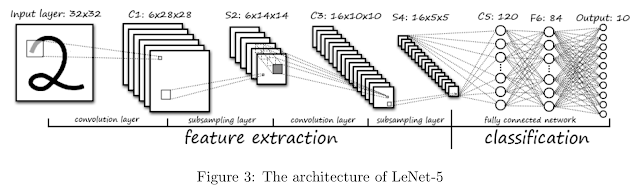


В этом случае g называется ядром. Все эти определения распространяются и на многомерный случай. СНС обычно выполняют двумерную свертку:



Архитектурные единицы СНС.

Одной из первых сверточных сетей является LeNet-5, которая использовалась для классификации рукописных цифр. Ее архитектура представлена на рисунке:



Эта сверточная сеть состоит из цепочки сверточных слоев, субдискретизирующих слоев и полносвязных слоев. Слой свёртки включает в себя для каждого канала свой фильтр, *ядро свёртки* которого обрабатывает предыдущий слой по фрагментам. Весовые коэффициенты ядра неизвестны и устанавливаются в процессе обучения. Скалярный результат каждой свёртки попадает на функцию активации, которая представляет собой некую нелинейную функцию. Слой активации обычно логически объединяют со слоем свёртки (считают, что функция активации встроена в слой свёртки). Функция нелинейности может быть любой по выбору исследователя, традиционно для этого использовали функции типа [гиперболического тангенса](https://ru.wikipedia.org/wiki/%D0%93%D0%B8%D0%BF%D0%B5%D1%80%D0%B1%D0%BE%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8) (f(x)=tanh(x), f(x)=|tanh(x)|) или [сигмоиды](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BC%D0%BE%D0%B8%D0%B4%D0%B0) (f(x)=(1+e−x)−1). Однако в 2000х годах была предложена новая функция активации — ReLU (*rectified linear unit*), которая позволила существенно ускорить процесс обучения и одновременно упростить вычисления (за счёт простоты самой функции), что означает блок линейной ректификации, вычисляющий функцию

f(x)=max(0,x). То есть по сути это операция отсечения отрицательной части скалярной величины. По состоянию на 2017 год эта функция и её модификации (Noisy ReLU, Leaky ReLU и другие) являются наиболее часто используемыми функциями активации в глубоких нейросетях, в частности, в свёрточных.

В процессе обработки матрицы ядром случается, что крайние пиксели не обрабатываются, потому что размерность ядра не позволяет включить края в центр. Если необходимо получить на выходе матрицу такого же размера, как на входе, помогает техника padding. Она добавляет к краям поддельные (fake) пиксели (обычно нулевого значения, вследствие этого к ним применяется термин “нулевое дополнение” — “zero padding”). Таким образом, ядро при проскальзывании позволяет неподдельным пикселям оказываться в своем центре, а затем распространяется на поддельные пиксели за пределами края, создавая выходную матрицу того же размера, что и входная.

Если же при работе со сверточным слоем необходимо получить выходные данные меньшего размера, чем входные, например в сверточных нейронных сетях, где размер пространственных размеров уменьшается при увеличении количества каналов, тогда используются субдискретизирующие слоя (pooling layer). Они представляют собой нелинейное уплотнение карты признаков, при этом группа пикселей (обычно размера 2×2) уплотняется до одного пикселя, проходя нелинейное преобразование. Наиболее употребительна при этом функция максимума. Преобразования затрагивают непересекающиеся прямоугольники или квадраты, каждый из которых ужимается в один пиксель, при этом выбирается пиксель, имеющий максимальное значение. Операция пулинга позволяет существенно уменьшить пространственный объём изображения. Пулинг интерпретируется так: если на предыдущей операции свёртки уже были выявлены некоторые признаки, то для дальнейшей обработки настолько подробное изображение уже не нужно, и оно уплотняется до менее подробного. К тому же фильтрация уже ненужных деталей помогает не переобучаться. Слой пулинга, как правило, вставляется после слоя свёртки перед слоем следующей свёртки. Еще один способ добиться этого — использовать stride (шаг).Идея stride заключается в том, чтобы пропустить некоторые области, над которыми скользит ядро. Шаг 1 означает, что берутся пролеты через пиксель, то есть по факту каждый пролет является стандартной сверткой. Шаг 2 означает, что пролеты совершаются через каждые два пикселя, пропуская все другие пролеты в процессе и уменьшая их количество примерно в 2 раза, шаг 3 означает пропуск 3-х пикселей, сокращая количество в 3 раза и т.д.