# Методы построения общего семантического векторного пространства на основе нейросетевых признаков

Николай Карпачёв, группа M05-894a

24 декабря 2018 г.

## 1 Введение. Постановка задачи.

Во многих задачах компьютерного зрения возникает необходимость в получении векторных представлений - семантических отображений из множества объектов в векторное пространство небольшой фиксированной размерности. Основное свойство таких отображений - близость векторов похожих объектов (свойство похожести определяется типом задачи).
Это свойство позволяет решать различные задачи ранжирования и поиска (в том числе кросс-модального) естественным образом, используя в качестве метрики похожести метрики близости Евклидова пространства.
В качестве представлений можно использовать признаки от моделей, обученных для других задач анализа изображений (например, часто используются классификационные признаки, дообученные для более специфичной задачи). Однако для многих задач такой подход не работает. К примеру, для задачи распознавания лиц такая схема работает не очень хорошо, так как количество классов очень велико. Для некоторых других задач, таких как кросс-модальный поиск и ранжирование, гораздо более естественной и реализуемой является идея оптимизации метрического пространства напрямую, используя понятие соответствия объектов.

## 2 Обучение общего семантического пространства. Общая схема.

Для каждой конкретной задачи построения метрического пространства признаков для объектов используется понятие общего контекста. Общий контекст образуют объекты, близкие по значению, таким образом желаемое свойство эмбеддингов - близость векторов объектов из общего контекста.
Например, для задачи поиска изображения по описанию общий контекст можно задать естественным образом - его образуют пары (изображение,

описание), соответствующие друг другу. Для задачи распознавания лиц контекст - это множество снимков одного и того же лица. Для задачи кластеризации текстов - текстовые фрагменты одной тематики.
Идея метода в том, чтобы используя в качестве основного feature extractor'а для объекта нейросетевую модель, определять целевую функцию таким образом, чтобы оптимизировать близость представлений объектов внутри одного контекста, сближая их вектора.
Такой подход был впервые предложен для задачи информационного поиска в статье *Learning Deep Structured Semantic Models for Web Search using Clickthrough Data* (DSSM). Схема работы сети представлена на схеме ниже. Имея сигнал о связи запроса и документа в виде кликов (это в данном случае является контекстом), оптимизируются метрики расстрояния между представлением для запроса и представлением для документов, соответствующих ему.

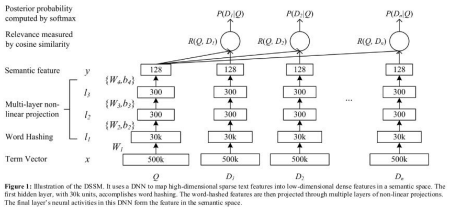


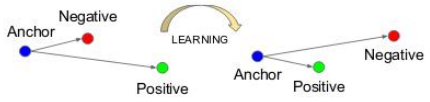
**Figure 1:** Illustration of the DSSM. It uses a DNN to map high-dimensional sparse text features into low-dimensional dense features in a semantic space. The first hidden layer, with 30k units, accomplishes word hashing. The word-hashed features are then projected through multiple layers of non-linear projections. The final layer's neural activities in this DNN form the feature in the semantic space.

# 3 Построение общего признакового пространства для распознавания лиц. FaceNet. Triplet loss

В статье *FaceNet: A Unified Embedding for Face Recognition and Clustering* была предложена архитектура построения векторных представлений для лиц, оптимизирующая напрямую результирующее Евклидово пространство, в котором расстояние между векторами представлений является метрикой похожести. Схема архитектуры аналогична описанной выше.


Входные изображения лиц подаются на вход глубокой сверточной архитектуре, на выходе из которой они преобразованы в вектора фиксированной

размерности (в данном случае она равна 128). В процессе обучения сэмплируется тройка объектов (триплет), состоящая из объекта-якоря (*anchor*), положительного примера (*positive*) и отрицательного примера (*negative*). Объекты выбраны следующим образом: пара (anchor, positive) - положительная пара - сэмплирована из общего контекста, в то время как (anchor, negative) - отрицательная пара - не принадлежат общему контексту.




По этой тройке вычисляется positive distance - норма разности anchor и positive и negative distance - норма разности anchor и negative.
Функция потерь в triplet loss - максимум между разностью между positive distance и negative distance с некоторым аддитивным параметров margin и нулем. Это соответствует получению штрафа за неправильно отранжированные объекты (когда представление одного из положительных объектов ближе к представлению отрицательного примера, чем к соответствующему ему второму положительному), либо недостаточно хорошо отранжированные (когда расстояние между положительными слабо превосходит расстояние между положительым и отрицательным)
В качестве нормы используется косинусное расстояние между l2 - отнормированными векторными представлениями объектов из триплета. Задача таким образом превращается в сближение похожих точек на поверхности сферы радиуса 1.

$$L = \sum_{i=1}^{N} max(0, margin + ||f_i^a - f_i^p||_2^2 - ||f_i^a - f_i^n||_2^2)$$

где:
$f_i^a$ - признаки anchor объекта
$f_i^p$ - признаки positive объекта
$f_i^n$ - признаки negative объекта

Есть и другая, более интерпретируемая метрика качества - доля правильно отранжированных триплетов.
Таким образом, для минимизации триплетной функции потерь сеть должна 'стягивать' элементы из одного контекста в одну область пространства.

# 4 Сэмплирование отрицательных примеров

Важным элементом алгоритма является стратегия сэмплирования отрицательных примеров (*negatives sampling*). Наиболее очевидная мысль - выбирать отрицательный пример случайным образом - тогда с высокой вероятностью он не будет принадлежать контексту *anchor*. У такого сэмплирования есть существенный недостаток - выбранный таким образом отрицательный пример как правило очень просто отличить от якоря. Сеть быстро научится различать случайно выбранные непохожие друг на друга примеры, однако на практике часто требуется ранжировать достаточно похожие объекты.

Для этого используют сэмплирование сложных отрицательных примеров. На некотором промежуточном шаге признаки от сети уже довольно осмысленны, и близкие объекты имеют близкие эмбеддинги. При таком сэмплировании (**hard negative mining**) в ранжирующий триплет попадает отрицательный пример с наиболее близким представлением к якорю. Это соответствует тройке, в ранжировании которой сеть "уверена" меньше всего. На практике это позволяет добиться более оптимального решения, так как сеть учится на более сложных примерах.

В статье FaceNet используется некоторая модификация hard negatives sampling: **semi-hard negatives sampling**. А именно, наиболее сложный отрицательный пример сэмплируется среди только тех примеров, которые не нарушают ранжирование внутри триплета (то есть, для которых расстояние внутри отрицательной пары не меньше расстояния внутри положительной пары). Утверждается, что такие примеры не настолько трудные для обучения сети, как *hard negatives*, но достаточно содержательные, чтобы сеть распознавала нетривиальные закономерности, что позволяет сети сходиться лучше на ранних эпохах.