**Краткий отчет о проделанной работе по проекту «Поиск по картинкам на примере распознавания афиш»**

Идея работы состоит в выполнении практической задачи поиска по базе изображения (в данном случае афиши), наиболее похожего на входное. Потенциально можно создать приложение, которое по данной фотографии сможешь найти в интернете соответствующее ей изображение и дать ссылку на ресурс, на котором она находится (и который будет вести к фильму), к примеру kinopoisk.

Работа над данным проектом ведется мной с середины февраля. Ниже я опишу то, что было мной сделано за это время.

1) В самом начале работы мною был написан файл First\_steps.cpp. В нем показываются самые базовые операции над изображениями и их представлениями в языке программирования. С помощью библиотеки OpenCV для C++ на данном этапе я научился таким операциям как чтение и представление изображения в виде матрицы (класс Mat), вывод изображения на экран, инвертирование цветов с помощью прямого прохода по пикселям, а также поворот картинки на произвольный угол.

Примерно сюда же можно отнести и файл grads\_and\_etc.cpp, на примере которого я разбирал понятие градиента.

Градиент — векторная величина показывающая направление наискорейшего возрастания функции интенсивности изображения.

Примерный срок : середина февраля — начало марта.

2) На втором этапе я познакомился с такими понятиями как детектор и дескриптор изображения.

В процессе описания изображения возникает задача выделения на нем особых точек, т. е. точек, обладающих свойствами отличимости, уникальности, стабильности к преобразованиям, инвариантности, а также обладающие характерными только им окрестностями. Если говорить проще, на изображении нужно выделить точки, которые останутся на своих местах и не потеряют своей информативности при аффинных и иных преобразованиях изображения. И чтобы это сделать надо пользоваться один из алгоритмов обнаружения таких точек по заданному изображению, иными словами, использовать детекторы.

Детекторы — алгоритмы нахождения на изображениях особых точек. Как пример, один из первых детекторов — детектор Харриса (улучшенная версия алгоритма Моравца), основанный на распознавании контуров — углов монохромных изображений, основываясь на измерениях векторов градиентов в точках.

После вычисления точек их нужно как-то описать, чтобы в дальнейшем иметь к ним доступ. Для этого используются дескрипторы изображений — алгоритмы, описывающие особые точки. Например, SIFT описывает для каждой точки направления градиентов в ее окрестностях. В частности, в своей работе я использовал проприетарные SIFT и SURF.

В процессе написания файла detectors.cpp я познакомился с синтаксисом операций определения и отождествления особых точек различных изображений. Мной были испробованы 3 вида детекторов — SIFT, SURF и Blobe, и два дескриптора — SIFT и SURF. Целью этого этапа было исследовать оптимальные входные параметры данных функций для достижения компромисса между количеством найденных точек и точностью отождествления (скажем, изображения с его повернутой копией).

Примерный срок : начало — середина марта.

3) На третьем этапе работы я познакомился с понятием кластеризации.

Кластеризация — это алгоритм машинного обучения, который разбивает данные на вход элементы на группы по определенным признакам. В случае данного проекта используется kmeans кластеризация.

(Пара слов о ней. Kmeans получает на вход описание точек изображения(ний), располагает их на условной n-мерной плоскости (по количеству параметров, задающих каждую точку) и случайно (или почти случайно, если использовать PP модернизацию) выбирает K точек — центров кластеров. Далее для каждой точки определяется центр, находящийся от нее на наименьшем расстоянии по заданной метрике.)

Вообще, у кластеризации есть 2 основных критерия — пожелания. Во-первых, нужно пытаться увеличить межклассовые расстояния, а во-вторых, стараться уменьшим расстояния внутри самих кластеров.

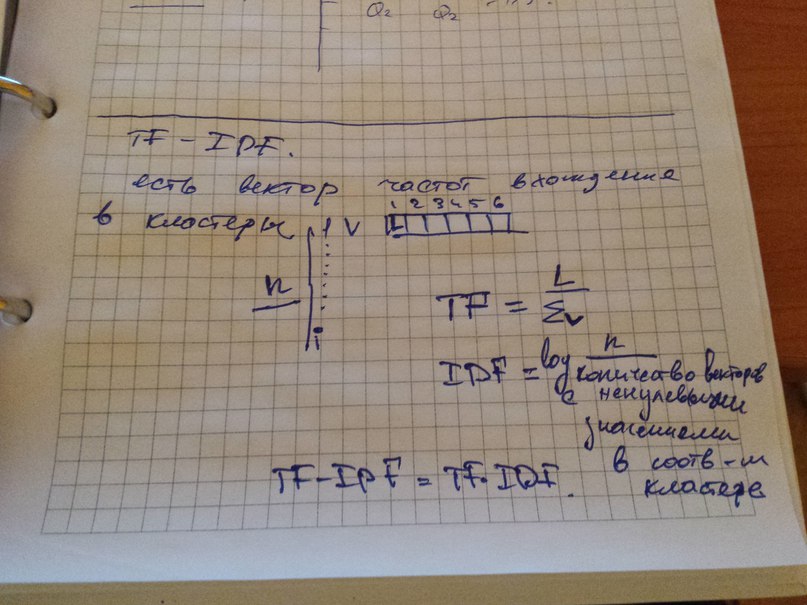
В файле clustering.cpp я смотрю, как происходит разбиение точек некоторого заданного изображения на кластеры.

Примерный срок : середина марта.

4) Представление изображений в виде визуальных слов.

После того, как мы смогли разбить базу изображений на кластеры, можно каждой особой точке каждого изображения присвоить номер ее кластера, и далее, как продолжение этого действия, создать для каждой картинки свой уникальный вектор размера K – количество кластеров – с количеством точек, содержащихся в данном изображении и принадлежащих одному из K кластеров. Получается своего рода гистограмма, описывающая изображения.

Такой метод описания по аналогии с задачей по обработке текстов называют Bag of Words, или раз уж речь идет о компьютерном зрении, Bag of Visual Words.



Далее возникла необходимость как-то сузить область поиска, т. к. подсчет расстояния до каждой из картинок базы — все же дело весьма затратное, тем более, что есть такие точки (например, зеленая трава), которые не несут отличительной информации, поскольку присутствуют на многих изображениях. Чтобы снизить их влияние на результат сравнения, используется TF-IDF метрика. Она позволяет сделать так, чтобы наиболее частые слова имели меньший вес и наоборот. Ее описание приведу на фотографии.

Примерный срок : конец марта.

5)На этом этапе происходила компиляция полученных мной за полтора месяца знаний. Реализован базовый поиск по картинкам на основании нормированных векторов визуальных слов для каждого изображения. Файл base\_creation.cpp умеет создавать базу изображений, записывая данный в yaml файлы, дополнять эту базу одиночными изображениями, а также собственно и искать картинки по их индексу. В качестве базы используются изображения, скачанные с Кинопоиска.

(Не уверен, что нужно приводить здесь подробное описание кода.)

Примерный срок : начало — середина апреля.

Дальнейшие планы:

* реализовать аффинные преобразования используя алгоритмы ransac. (Середина апреля — начало мая)
* Если остается время, попробовать сделать приложение или просто нечто наглядное, на чем можно запустить этот проект так, чтобы он имел практическую пользу.

В перспективе хотелось бы продолжить изучение данной темы, особенно учитывая ее стремительное развитие в последнее время во многих сферах как слабого ИИ.

Все актуальные версии программ и примеры кода находятся в моем репозитории на bitBucket по адресу https://lokodenis@bitbucket.org/lokodenis/cv\_project.git

Проект выполняет студент 1 курса бакалавриата ФКН ВШЭ Беляков Денис

под руководством Горбачева Вадима.