**Министерство образования и науки РФ**

**Федеральное государственное автономное образовательное учреждение**

**Высшего образования**

**«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**ВЫСШАЯ ШКОЛА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ИНФОРМАЦИОННЫХ СИСТЕМ**

Специальность: 09.03.03. – Прикладная информатика

ОТЧЕТ ПО ТЕМЕ:

**РАСПОЗНАВАНИЕ АВТОМОБИЛЬНЫХ НОМЕРОВ**

Работу выполнили:

Студент 3 курса

Группа 11-406

Н.С.Ионов

Студент 3 курса

Группа 11-403

Б.В.Хасанов

Содержание

[**Набор данных**](#_ghxtncokcia0)[**2**](#_ghxtncokcia0)

[**Преобразование данных**](#_m0j8p34igljj)[**3**](#_m0j8p34igljj)

[**Методы машинного обучения**](#_cvewgp57dxi)[**4**](#_cvewgp57dxi)

[**Оценка качества алгоритма**](#_8y1hit5qr5cx)[**5**](#_8y1hit5qr5cx)

[**Выводы работы**](#_508jciusd7uz)[**6**](#_508jciusd7uz)

[**Листинг программы**](#_4irknz6jr1r9)[**7**](#_4irknz6jr1r9)

# Набор данных

Цель нашего проекта — реализация распознавания государственных номеров автомобилей. Мы взяли набор данных, который представлял собой подборку фотографии автомобилей с различимыми государственными номерами Российской Федерации. Данная база с номерами была нами найдена на просторах ресурса «Хабрахабр». Она была создана для проекта одной из компаний, которая занималась установкой систем контроля доступа.

Данные представляют собой более 3000 фотографий государственных номеров Российской Федерации и к этому более 15000 изображений с отдельными символами, которые используются в государственных номерах Российской Федерации. Они представлены в формате BMP. Данные изображения разделены на несколько категорий и рассортированы по разным директориям.

Т.к. в данной базе присутствовали не только фотографии номеров автомобилей, но и другие различные похожи изображения (номера телефонов на рамках номерных знаков, дорожные знаки, рекламные объявления и т.п.), то нами была произведена ручная сортировка и отсеивание изображений неподходящих под наш проект. Далее были выбраны наиболее удачные фотографии с наименьшим количеством цифрового шума, ряби и грязи. Также было произведено выравнивание номеров по горизонтали.

В конечном итоге у нас остались наиболее подходящие изображения, которые были использованы для обучения и тестирования работы нашей программы. В группу обучения вошли изображения отдельных символов с номерных знаков, а в группу тестирования - полные изображения гос. номеров, выдаваемых на территории РФ.

# Преобразование данных.

Для более удобной работы с базой символов гос. номеров, каждое изображение мы изменяем до формата 40 на 40 пикселей. Далее, полученную картинку мы преобразуем в одномерный массив. В нашем проекте, все изображения отдельных символов лежат по отдельным директориям. Имя каждой папке соответствует символу, изображения которого в данной папке располагаются. При формировании массивов из изображений, значение каждого изображения будет присвоен исходят из того, в какой директории данное изображение находится. На следующем этапе, используя метод главных компонент, мы уменьшаем каждый массив пикселей.

# Методы машинного обучения

После всех преобразований нашей базы данных, мы получили dataset, по которому будет происходить обучение. В обучении мы решили использовать метод опорных векторов. Данный метод основан на переводе исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве. Две параллельных гиперплоскости строятся по обеим сторонам гиперплоскости, разделяющей классы. Разделяющей гиперплоскостью будет гиперплоскость, максимизирующая расстояние до двух параллельных гиперплоскостей. Алгоритм работает в предположении, что чем больше разница или расстояние между этими параллельными гиперплоскостями, тем меньше будет средняя ошибка классификатора. Мы имеем возможность задавать количество изображений каждого символа, на которых будет происходит обучение. С помощью этого мы можем выбрать наиболее оптимальное количество изображений для обучения, при котором качество распознавания и время потраченное на обучение, будет наиболее оптимальным. После того, как мы обучили нашу машину распознавать символы, на начинаем её тестирование. Тестирование происходит на полноценных изображениях гос. номеров. Т.к. обучение происходило на отдельно взятых символах, то перед началом распознавания, изображение с гос. номером делится программой на отдельные элементы, каждый из которых содержит лишь один символ. Данное деление реализовано с помощью идентификации группы цветов на изображении и вычленении областей с наиболее большим количеством пикселей черного цвета. После того, как программа разделила изображение, она, на основе данных полученных в процессе обучения, начинает распознавание символов.

# Оценка качества алгоритма

Метод опорных векторов показал себя как удобный и действенный метод для распознавания символов с изображений. Даже при использовании малого количества изображений для обучения, он довольно точно отрабатывает на простых изображениях с четкими границами символами и без помех. При увеличении количества изображений для обучения, начинает возрастать точность распознавания. Одним из главных недостатков, с которым мы столкнулись в ходе использования этого алгоритма - долгая обработка обучающих изображений. С помощью изменения количества изображений для обучения, мы можем найти такое количество, при котором нас будет устраивать и время обучения, и результат распознавания.

# Выводы работы

Выполнение данного проекта позволило углубиться в понимание машинного обучения. Данный проект расширил знания не только в сфере машинного обучения и его алгоритмов, но и в сфере работы с изображениями, их распознаванием и обработкой. В дальнейшем, можно усложнять задачу используя не только печатные символы русского языка, но рукописные, которые будут включать в себя буквы из других языков. Также усложнением задачи будет использование для распознавания не только чистых изображений, но изображений с различными шумами, контрастами и прочими помехами.

# Листинг программы

# -\*- coding: utf-8 -\*-

**from** PIL **import** Image

**import** os

**from** sklearn.model\_selection **import** GridSearchCV

**from** sklearn.decomposition **import** PCA

**from** sklearn.svm **import** SVC

**import** numpy **as** np

###############################################################################

# используемые символы

iconset = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'e', 'h', 'k', 'p',

'm', 't', 'x', 'y']

###############################################################################

# листы для хранения изображений с символами

imageset = []

targetset = []

###############################################################################

# количиство изображений для изучения символа

symbols\_count = 100

###############################################################################

# цикл проходит по изображениям с символами, меняет размер изображений на 40x40,

# преобразует изображение в массив пикселей, добавляет полученный массив в imageset,

# добавляет значение изображения в targetset

**for** letter **in** iconset:

**for** img **in** os.listdir('./symbols/%s/' % (letter))[:symbols\_count]:

im = Image.open("./symbols/%s/%s" % (letter, img))

im = im.resize((40, 40))

imageset.append(np.ravel(np.array(im)))

targetset.append(letter)

**print** ("Symbols loaded")

**print** ("..............")

X = np.array(imageset)

y = np.asarray(targetset)

###############################################################################

# использование метода главных компонент

n\_components = 160

pca = PCA(n\_components=n\_components, svd\_solver='randomized',

whiten=True).fit(X)

X\_pca = pca.transform(X)

**print** ("PCA completed")

**print** ("..............")

###############################################################################

# использование метода опорных векторов

param\_grid = {'C': [1e3, 5e3, 1e4, 5e4, 1e5],

'gamma': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1], }

clf = GridSearchCV(SVC(kernel='rbf', class\_weight='balanced'), param\_grid)

clf.fit(X\_pca, y)

**print** ("Training completed")

**print** ("..............")

###############################################################################

# здесь должна происходить загрузка гос. номеров, нахождение в них символов,

# преобразование данных символов в изображения и массивы пикселей, но мы не

# смогли найти (или придумать) подходящего решения

###############################################################################

# преобразование изображения

im = Image.open("3.bmp")

im = im.convert("P")

im = im.resize((40, 40))

###############################################################################

# преобразование изображения в массив пикселей

predict\_array = np.ravel(np.array(im))

predict\_array = pca.transform(predict\_array).reshape(1, -1)

**print**(clf.predict(predict\_array))