«Национальный исследовательский университет

«Высшая школа экономики»

Групповой проект

**«Классификация изображений,**

**датасет на основе скриншотов/моделей видеоигры»**

Выполнили:

*Баранова Дарья*

*Павлов Сергей*

*Петренко Денис*

*Озеров Максим*

Содержание

**Введение3**

Цели и задачи работы3

**Сбор данных4**

**Построение baseline модели6**

Предпосылки6

Описание метода6

Метрики и полученное качество7

**Улучшение модели8**

Calibrated SVM8

**Заключение9**

**Литература10**

**Приложение11**

1. **Введение**
   1. **Цели и задачи работы**

Целью работы ставится:

* Изучение подходов к обработке изображений и выделению из них ключевых признаков;
* Изучение способов классификации изображений с помощью классических подходов машинного обучения;
* Решение собственной задачи классификации изображений методами классического машинного обучения (без использования нейронных сетей).

В качестве собственной задачи была выбрана задача классификации изображений со скриншотами моделей оборудования из игры «[Космические рейнджеры 2: Доминаторы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D1%81%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%80%D0%B5%D0%B9%D0%BD%D0%B4%D0%B6%D0%B5%D1%80%D1%8B_2:_%D0%94%D0%BE%D0%BC%D0%B8%D0%BD%D0%B0%D1%82%D0%BE%D1%80%D1%8B)».

Для достижения цели были выделены следующие задачи работы:

1. Собрать и обработать (при необходимости) датасет необходимого размера;
2. Выбрать метрики качества, с помощью которых будет оцениваться качество будущей модели;
3. Разработать baseline метод для классификации изображений, а также спроектировать и обучить модель;
4. Сформулировать и обосновать возможные способы улучшить полученное качество классификации и реализовать их. Оценить, насколько улучшилось качество классификации;
5. Подвести итоги и описать наилучшую полученную модель для решения поставленной задачи.

1. **Сбор данных**

В качестве изображений для классификации были выбраны статические рейнджеры оборудования в космическом корабле игрока из игры «Космические рейнджеры 2: Доминаторы».

Сбор данных был произведён вручную с помощью созданной фанатами утилиты [1], предназначенной для извлечения графических элементов из дистрибутива игры. Такая возможность позволила не пользоваться внутриигровыми скриншотами, которые содержали бы в себе лишние элементы фона и, возможно, были бы хуже исходников по качеству.

В исходниках игры каждое изображение оборудования представляет собой циклический анимированный ряд из ~100 кадров. Поскольку для задачи необходимы статические изображения, для каждой модели оборудования был вручную выбран наиболее репрезентативный (на глаз) кадр.

В итоге каждое исходное изображение обладает следующими характеристиками:

* формат: PNG
* размер: 75 \* 75 pixel
* фон: прозрачный

Исходная выборка содержит 102 изображения, которые относятся к различным 9 классам оборудования со следующим распределением:

|  |  |
| --- | --- |
| Наименование класса | Количество уникальных изображений |
| Оружие (WEAPON) | 15 |
| Генератор защитного поля (DEFGENERATOR) | 11 |
| Двигатель (ENGINE) | 11 |
| Топливный бак (FUELTANKS) | 11 |
| Радар (RADAR) | 11 |
| Ремонтный дроид (REPAIRROBOT) | 11 |
| Сканер (SCANER) | 11 |
| Корпус (HULL) | 11 |
| Захват (CARGOHOOK) | 10 |

Каждый класс представляет собой множество элементов, внешне сильно различающихся между собой. Например, класс «Оружие» состоит из 15 различных видов орудийный инструментов различной формы, размера и цвета. Однако все эти орудийные инструменты принадлежат категории «Оружие», а значит должны быть отнесены к одному классу несмотря на их внешнее сильное различие. Изображения элементов классов представлены в приложении.

Чтобы увеличить объем обучающей выборки исходные изображения были аугментированы разными искажениями до получения 3000 изображений со следующим распределением:

|  |  |
| --- | --- |
| Наименование класса | Количество уникальных изображений |
| Оружие (WEAPON) | 480 |
| Генератор защитного поля (DEFGENERATOR) | 325 |
| Двигатель (ENGINE) | 345 |
| Топливный бак (FUELTANKS) | 322 |
| Радар (RADAR) | 317 |
| Ремонтный дроид (REPAIRROBOT) | 312 |
| Сканер (SCANER) | 312 |
| Корпус (HULL) | 310 |
| Захват (CARGOHOOK) | 277 |

Аугментация проводилась с помощью пайплайна из случайных изменений:

* Поворот (*rotate*) от -8 до +8 градусов с вероятностью 0.5;
* Изменение масштаба (*zoom*) с коэффициентом от 0.8 до 1.01 с вероятностью 0.5;
* Отражение по горизонтали (*flip\_left\_right*) с вероятностью 0.5;
* Случайное искажение (*random\_distortion*) с вероятностью 0.7 (пример искажения представлен на рисунке ниже).

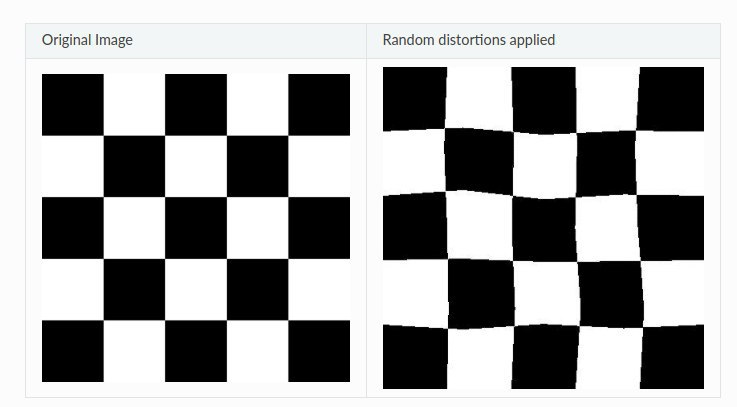


Рис. Пример случайного искажения (слева – оригинальное изображение, справа - искаженное)

Количество сгенерированных изображений выбиралось исходя из двух предпосылок:

1. Чем больше обучающая выборка будет, тем лучшего качества можно будет достичь при обучении модели;
2. Слишком большое количество данных может повлечь слишком долгое обучение (подробнее это описано в разделе построения baseline модели).

**3. Построение baseline модели**

**3.1 Предпосылки**

Изображения разных классов не сильно различаются между собой. Можно сказать, что даже для человеческого глаза изображения разных классов трудно различимы, так как все модели игры имеют довольно разнообразный внешний вид.

Исходя из этого, было принято решение строить baseline на основе техники Bag-of-words для computer vision [2-4].

**3.2 Описание метода**

Для извлечения признаков из изображений использовалось масштабно-инвариантное преобразование функций SIFT (Scale Invariant Feature Transform) [5-[6](https://medium.com/data-breach/introduction-to-sift-scale-invariant-feature-transform-65d7f3a72d40)].

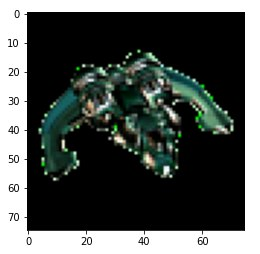
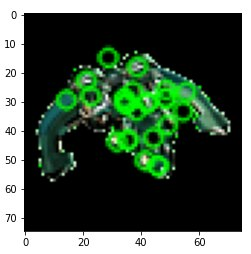
 

Рис. Определение ключевых точек методом SIFT (слева – исходное изображение, справа с отрисованными выделенныи точками)

После этого на основе выделенных дескрипторов каждого изображения строится соответствующее векторное пространство. Из множества дескрипторов для изображений получается набор векторов в этом пространстве с соответствующими координатами. Однако так как векторов много и нужно как-то выделить «ключевые точки», которые и будут представлять собой слова в словаре, нужно все это пространство кластеризовать. Проводится эта процедра с помощью K-Means алгоритма, который данное векторное пространство разбивает на K кластеров – опорных точек пространства. Эти опорные точки представляют собой «словарь» в представлении «Bag-of-words».

Следующим шагом исходные изображения описывались с помощью полученных опорных точек. Для каждого изображения определяется какие опорные точки лучше всего его описывают (какие кластеры находятся ближе всего к каждому из векторов его векторного представления). Таким образом, каждое изображение мы представляем как набор «слов из словаря».

После этого для каждого такого набора слов можно построить гистограмму. Таким образом, для каждого изображения мы получим по одной гистограмме. Именно их мы и будет классифицировать с помощью метода опорных векторов.

Описанный метод классификации накладывает значительное ограничение на объем обучающей выборки. K-Means алгоритм в процессе работы вычисляет матрицу расстояний, из-за этого время обучения на больших объемах данных может сильно возрастать.

**3.3 Метрики и полученное качество**

Поставленная задача представляет собой задачу многоклассовой классификации. Однако baseline не предполагает построения нескольких моделей для бинарной классификации. Вместо этого строится одна модель, которая на выходе может выдавать один из 9 классов.

В качестве метрики качества для такой модели будет использоваться Accuracy, задающаяся как:

В собранных данных классы распределены почти что равномерно, поэтому адекватное применение Accuracy возможно.

Эмпирически были получены следующие метрики качества:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Количество кластеров | Размер обучающей выборки | Размер тестовой выборки | Качество на train | Качество на test |
| 200 | 900 | 2100 | 1.0 | 0.614 |
| 1500 | 1500 | 1.0 | 0.677 |
| 2250 | 750 | 0.974 | 0.742 |
| 500 | 900 | 2100 | 1.0 | 0.797 |
| 1500 | 1500 | 1.0 | 0.849 |
| 2250 | 750 | 1.0 | 0.897 |

Столбец «Количество кластеров» отражает величину K в методе K-Means.

Так как метод работает остаточно долго, K-Means в приведённой выше таблице проходил всего одну итерацию. При попытке выставить хотя бы 5 итераций качество классификации, к сожалению, не выросло:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Количество итераций | Количество кластеров | Размер обучающей выборки | Размер тестовой выборки | Качество на train | Качество на test |
| 1 | 500 | 2250 | 750 | 1.0 | 0.897 |
| 5 | 500 | 2250 | 750 | 1.0 | 0.889 |

**4. Улучшение модели**

Следующие шаги могут привести к улучшению качества текущего решения:

* 1. Провести подбор гиперпараметров модели;
  2. Применить другую модель. Например, градиентный бустинг;
  3. Подобрать на кроссвалидации параметры для модели кластеризации;
  4. Применить другой подход к кластеризации: dbscan, иерархическую кластеризацию;
  5. Применить другой алгоритм для извлечения признаков из изображения. Это могут быть следующие алгоритмы:
  6. ORB (Oriented FAST and Rotated BRIEF) [7];
  7. SURF (Speeded Up Robust Features) [8].

**4.1 Calibrated SVM**

Возьмем построенные модели SVM и посмотрим на вероятности, с которыми модель определяет класс для каждого тестового изображения. После калибровки вероятностей и выбора наиболее вероятного класса увидим, что такое незатейливое улучшение сильно увеличивает качество классификации:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Количество кластеров | Размер обучающей выборки | Размер тестовой выборки | Accuracy baseline | Улучшенное Accuracy |
| 200 | 900 | 2100 | 0.614 | 0.824 |
| 1500 | 1500 | 0.677 | 0.849 |
| 2250 | 750 | 0.742 | 0.912 |
| 500 | 900 | 2100 | 0.797 | 0.915 |
| 1500 | 1500 | 0.849 | 0.933 |
| 2250 | 750 | 0.897 | 0.948 |

**Заключение**

В рамках первого этапа проекта выполнены следующие шаги:

1. Сбор и подготовка данных
2. Разработано baseline решение для задачи классификации изображений
3. Разработан план по улучшению текущей модели

В рамках второго этапа проекта было выполнено:

В результате работы были изучены некоторые существующие способы обработки изображений и извлечения из них признаков для классификации. Был спроектирован и реализован baseline метод для решения поставленной задачи классификации моделей оборудования из видеоигры.

**Литература**

# Космические Рейнджеры Вики, ULR: https://rangers.fandom.com/ru/wiki/ResEditor

# Википедия, Bag-of-words model in computer vision, 2021, URL:https://en.wikipedia.org/wiki/Bag-of-words\_model\_in\_computer\_vision

# PyImageSearch, The bag of (visual) words model, URL: https://customers.pyimagesearch.com/the-bag-of-visual-words-model/

# Sachin Mohan, Image Classification using Bag of Visual Words Model, 2020, URL:<https://machinelearningknowledge.ai/image-classification-using-bag-of-visual-words-model/>

# Deepanshu Tyagi, Introduction To Feature Detection And Matching, 2019, URL:<https://medium.com/data-breach/introduction-to-feature-detection-and-matching-65e27179885d>

# Deepanshu Tyagi, Introduction to SIFT (Scale Invariant Feature Transform), 2019, URL:<https://medium.com/data-breach/introduction-to-sift-scale-invariant-feature-transform-65d7f3a72d40>

# Deepanshu Tyagi, Introduction to ORB (Oriented FAST and Rotated BRIEF), 2019, URL:<https://medium.com/data-breach/introduction-to-orb-oriented-fast-and-rotated-brief-4220e8ec40cf>

# Herbert Bay, Tinne Tuytelaars, Luc Van Gool, SURF: Speeded Up Robust Features, 2006, URL:<https://people.ee.ethz.ch/~surf/eccv06.pdf>

**Приложение**

Изображения классифицируемых моделей класса «CARGOHOOK»



Изображения классифицируемых моделей класса «DEFGENERATOR»



Изображения классифицируемых моделей класса «ENGINE»



Изображения классифицируемых моделей класса «FUELTANKS»



Изображения классифицируемых моделей класса «HULL»



Изображения классифицируемых моделей класса «RADAR»



Изображения классифицируемых моделей класса «REPAIRROBOT»



Изображения классифицируемых моделей класса «SCANER»



Изображения классифицируемых моделей класса «WEAPON»

