МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И. Сатпаева

Институт автоматики и информационных технологий

Кафедра «Программная инженерия»

Қуандықова Сара Равилқызы

Сравнительный анализ методов кластерного анализа в решении задач распознавания

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

5В070400 – Вычислительная техника и программное обеспечение

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И. Сатпаева

Институт автоматики и информационных технологий

Кафедра «Программная инженерия»

ДОПУЩЕН К ЗАЩИТЕ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____А.Н. Молдагулова
«20» 05 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

На тему: «Сравнительный анализ методов кластерного анализа в решении задач распознавания»

по специальности 5В070400 – Вычислительная техника и программное обеспечение

Выполнила Куандыкова С. Р.

Рецензент
старший преподаватель, доктор Ph.D.
_____ Даркенбаев Д. К.
«_____» _____ 2022 г.

Научный руководитель
сениор лектор, доктор Ph.D.
_____ Черикбаева Л.Ш.
«19» 05 2022 г.

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И. Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

5В070400 – Вычислительная техника и программное обеспечение

УТВЕРЖДАЮ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«20» 05 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся *Куандыковой Саре Равлқызы*

Тема: *Сравнительный анализ методов кластерного анализа в решении задач распознавания*

Утверждена *приказом проректора по академической работе* № 489-П/О
от "24" 12 *2021 г.*

Срок сдачи законченного проекта: "25" 05 *2022 г.*

Исходные данные к дипломному проекту: *сбор теоретического материала, данные анализа по данной теме.*

Краткое содержание дипломной работы:

а) Обзор методов кластерного анализа

б) Обзор нейронных сетей

в) Применение нейронных сетей в кластеризации

г) Сравнительный анализ

Перечень подлежащих разработке в дипломном проекте вопросов (с точным указанием обязательных чертежей) *представлены __ слайда презентации.*

Рекомендуемая основная литература: *из __ наименований*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю и консультантам	Примечание
1. Обзор методов кластерного анализа	20.01.22	Выполнено
2. Обзор нейронных сетей	15.02.22	Выполнено
3. Применение нейронных сетей в кластеризации	20.03.22	Выполнено
4. Сравнительный анализ	18.04.22	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты, И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись
Нормоконтролер	Жекамбаева М.Н. доктор Ph.D., ассоциированный профессор	17.05.22г	
Программное обеспечение	Марғұлан Қ. магистр техн. наук, лектор	17.05.22г	

Научный руководитель 19.05.2022 Черикбаева Л.Ш.

Задание принял к исполнению обучающийся Куандыкова С.Р.

Дата 19.05.2022. «17» 11 2021 г.

АННОТАЦИЯ

Основной целью дипломной работы является сравнение методов кластерного анализа в решении задач распознавания.

При выполнении работы было выполнено исследование предметной области: изучение алгоритмов кластеризации, исследованы методы кластерного анализа. Изучены модели нейронной сети и использованы в целях применения их в кластеризации.

Для выполнения данной работы используется язык программирования Python с использованием библиотек tensorflow, sklearn. Среда выполнения Google Colab.

Дипломная работа состоит из введения, четырех основных разделов (исследовательский, технический, проектный, экспериментальный), заключения, списка использованной литературы и приложений.

В введении раскрыта актуальность дипломного проекта.

В исследовательском разделе поставлена цель разработки, определены термины и сокращения. Описан теоретический материал.

В технологическом разделе рассмотрен стек технологий, используемый в дипломном проекте.

В проектном разделе описывается проектная часть работы.

В экспериментальном разделе приводятся результаты работы.

Дипломная работа состоит из 39 страниц, 26 рисунков и 2 приложений.

АҢДАТПА

Дипломдық жұмыстың негізгі мақсаты тану есептерін шешуде кластерлік талдау әдістерін салыстыру болып табылады.

Жұмысты орындау кезінде пәндік саланы зерттеу жүргізілді: кластерлеу алгоритмдерін зерттеу, кластерлік талдау әдістері зерттелді. Нейрондық желі модельдері зерттелді және оларды кластерлеуде қолдану үшін қолданылды.

Бұл жұмысты орындау үшін TensorFlow, sklearn кітапханаларын қолдана отырып, Python бағдарламалау тілі қолданылады. Google Colab жұмыс уақыты.

Дипломдық жұмыс кіріспеден, төрт негізгі бөлімнен (зерттеу, техникалық, жобалау, эксперименттік), қорытындыдан, пайдаланылған әдебиеттер тізімі мен қосымшалардан тұрады.

Өзгерістер енгізу ашылуы өзектілігі дипломдық жоба.

Зерттеу бөлімінде даму мақсаты қойылған, терминдер мен қысқартулар анықталған. Теориялық материал сипатталған.

Технологиялық бөлімде дипломдық жобада қолданылатын технологиялар стегі қарастырылған.

Жобалау бөлімінде жұмыстың жобалық бөлігі сипатталады.

Эксперименттік бөлімде жұмыс нәтижелері келтірілген.

Дипломдық жұмыс 39 беттен, 26 суреттен және 2 қосымшадан тұрады.

ANNATATION

The main purpose of the thesis is to compare the methods of cluster analysis in solving recognition problems.

During the execution of the work, a study of the subject area was carried out: the study of clustering algorithms, methods of cluster analysis were investigated. Neural network models have been studied and used for their application in clustering.

To perform this work, the Python programming language is used using Tensorflow and sklearn libraries. The Google Colab runtime environment.

The thesis consists of an introduction, four main sections (research, technical, design, experimental), a conclusion, a list of references and appendices.

The introduction reveals the relevance of the graduation project.

The research section sets the goal of development, defines terms and abbreviations. The theoretical material is described.

In the technological section, the stack of technologies used in the graduation project is considered.

The project section describes the project part of the work.

The experimental section presents the results of the work.

The thesis consists of 39 pages, 26 drawings and 2 appendices.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Распознавание изображений – это способность программы понимать, что находится на изображении и определять его класс. Можно сказать, что распознавание изображений относится к задачам классификации. В рамках этой цели нейронные сети идентифицируют объекты на изображении и присваивают им одну из предопределенных групп. При классификации используются размеченные данные (обучение с учителем). Однако чаще всего случается так, что размеченные данные сложнее получить. В таких случаях, используется кластеризация (обучение без учителя), которая размечает данные, и дает возможность дальнейшего анализа.

В действительности, кластеризацию можно рассматривать как процесс группировки объектов в кластеры, таким образом, что изображения, попадающие в один и тот же кластер, должны иметь схожие визуальные характеристики, чем изображения из разных кластеров. Зачастую, количество кластеров изначально не известно, и задаются либо вручную, либо самим алгоритмом.

Распознавание образов применяется во многим сферах, в таких как:

- маркетинг;
- видеонаблюдении;
- медицине;
- автомобилях;
- дронах и т.д.

1 Исследовательский раздел

1.1 Цель разработки

Цель дипломной работы – проведение сравнительного анализа методов кластерного анализа в решении задач распознавания.

Задачи дипломной работы:

- обзор методов кластерного анализа;
- обзор нейронных сетей;
- выбор средств разработки;
- разведочный анализ данных;
- выбор моделей кластеризации;
- выбор моделей нейронной сети;
- проведение кластерного анализа;
- проведение сравнительного анализа результатов.

1.2 Определения, термины и сокращения

В таблице 1 приведены термины и сокращения, которые использовались в предметной области проекта, а также специфические термины, связанные с используемыми технологиями.

Таблица 1 – Определение сокращений и терминов

Сокращение или термин	Определение
k-means	Алгоритм кластеризации (к-средние)
AP	(сокр. от англ. affinity propagation). Метод распространения близости
AC	(сокр. от англ. aagglomerative clustering). Агломеративная кластеризация
BIRCH	(сокр. от англ. balanced iterative reducing and clustering using hierarchies). Метод сбалансированного итеративного сокращения и кластеризации с использованием иерархии
CFT	(сокр. от англ. cluster feature tree). Дерево функций кластеризации
CovNet/CNN	(сокр. от англ. convolution neural network). Сверточная нейронная сеть
VGG16	Модель CNN
ResNet50	(сокр. от англ. residual networks 50). Модель CNN

Продолжение таблицы 1

InceptionV3	Модель нейронной сети CNN
RGB	(сокр. от англ. red green blue)
PCA	(сокр. от англ. principal component analysis). Метод главных компонент
ЯП	Язык программирования
Центроид	Центр тяжести кластера

1.3 Предметная область

Предметной областью дипломного проекта, являются: машинное обучение, глубокое обучение, кластеризация изображений.

1.4 Методы кластерного анализа

Кластерный анализ можно условно разделить на 3 категории:

1. Иерархический метод;
2. Метод основанный на плотности;
3. Вероятностный метод.

Иерархическая кластеризация направлена на создание иерархии кластеров. Выделяют два метода иерархической кластеризации:

- *восходящий метод*, когда каждый элемент выборки считается отдельным кластером, объединяясь создаются новые кластеры. Таким образом создается дерево иерархии от листьев к корню;
- *нисходящий метод*, противоположность восходящему методу. Кластеры формируются от корня к листьям.

Метод, основанный на плотности, вычисляет площадь сосредоточения точек отделяя их от полых и редких областей. Точки, которые не относятся к какому-либо кластеру помечаются как шум. При этом если область концентрирует внутри себя достаточно точек и становится плотной, то эта область считается новым кластером [1].

Вероятностный метод подразумевает, что все элементы относятся к какому-либо кластеру [2].

K-means. Алгоритм можно описать пятью основными этапами:

1) Определение k центров кластеров.
2) Определение принадлежности объектов кластерам.
3) Определение центроидов k кластеров.
4) Сравнение центров и центроидов кластеров

5) Повторить итерацию [3].

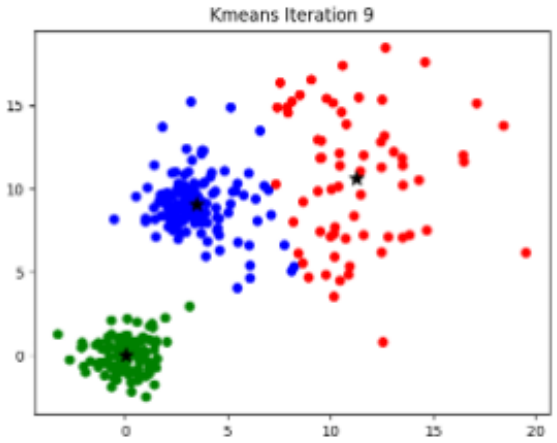


Рисунок-1.4.1 – Визуализация алгоритма k-means

Метод распространения близости (Affinity Propagation). Данный алгоритм вводит три матрицы:

- схожести S (i, k), по которой можно сказать, насколько похожи соседние точки;
- ответственности R (i, k), которая показывает, насколько i элемент хочет видеть k элемент своим лидером;
- доступности A (i, k).

При вычислении r (i, k) происходит передача сообщений от точек данных к возможным лидерам – рассматриваются матрицы S и A вдоль строк.

При вычислении a (i, k) происходит передача сообщений от возможных лидеров ко всем остальным точкам – рассматриваются матрицы S и R вдоль столбцов.

Данный алгоритм принимает 2 обязательных параметра: предпочтение и коэффициент демпфирования. Недостатком алгоритма является его сложность $O(N^2)$. Это делает алгоритм наиболее подходящим для малых или средних наборов данных [1].

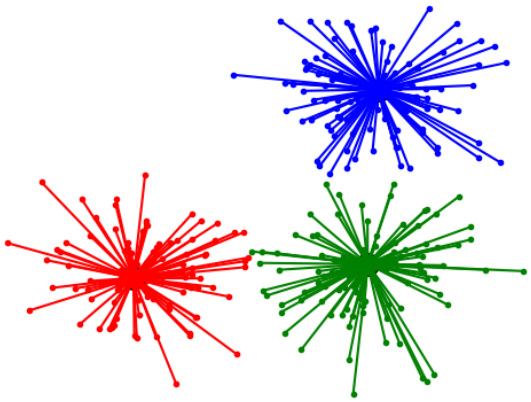


Рисунок-1.4.2 – Визуализация метода распространения близости

Агломеративная кластеризация. Основана на иерархической кластеризации, которая используется для формирования иерархии кластеров. Алгоритм является восходящим, т.е. начинается с создания множества небольших кластеров, и таким образом дерево создается от листьев к корню. Сложность алгоритма $O(N^2)$ [1].

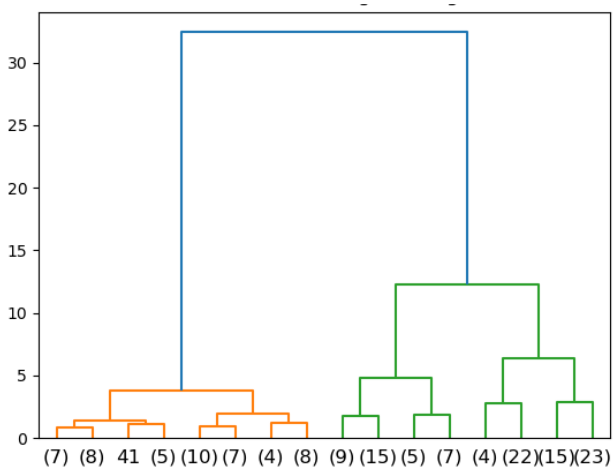


Рисунок-1.4.3 – Визуализация агломеративной кластеризации

Сбалансированное итеративное сокращение и кластеризация с использованием иерархии (BIRCH). Алгоритм строит дерево, которое имеет название – дерево функций кластеризации (CFT). Clustering Feature (CF Nodes) – это сжатые до набора узлов данные с потерями. Алгоритм исключает выпадения и объединяет заполненные кластеры в большие кластеры, применяя имеющийся метод кластеризации для всех листов – агломерирующий иерархический алгоритм. Алгоритм обеспечивает гибкость, предоставляя пользователям возможность указывать нужное количество кластеров, либо нужный предел диаметра кластеров [1].

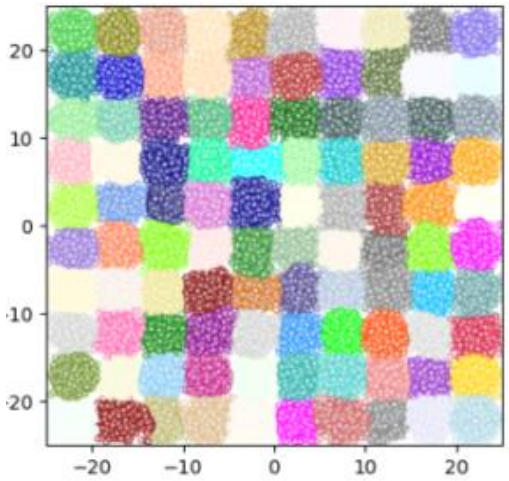


Рисунок-1.4.4 – Визуализация алгоритма BIRCH

1.5 Нейронная сеть

Нейронными сетями называются подмножества машинного обучения, которые лежат в основе алгоритмов глубокого обучения. Название и структура были вдохновлены человеческим мозгом и имитируют биологические нейроны, которые подают сигналы друг другу.

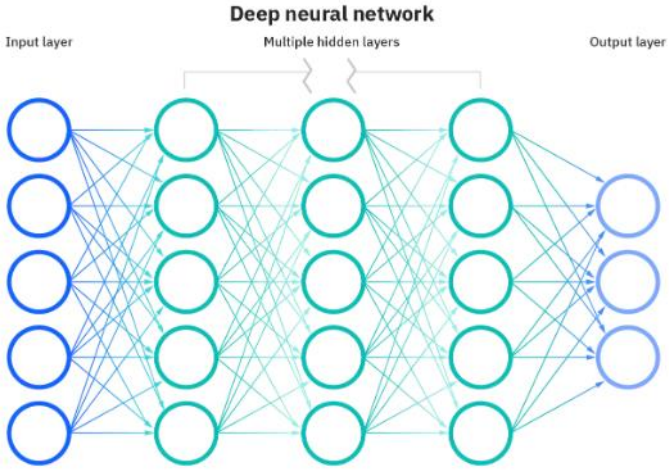


Рисунок-1.5.1 – Графическое представление нейронной сети

Нейронная сеть состоит из входного и выходного слоев. Между ними располагаются скрытые слои. Количество скрытых слоев может достигать от нескольких десятков до нескольких тысяч. Полученная матрица скрытого слоя служит матрицей входных данных для следующего слоя. Результат содержит только последний выходной слой.

Одним из алгоритмов глубоко обучения является сверточная нейронная сеть (CovNet, CNN). Алгоритм принимает на вход изображение, а скрытые слои извлекают признаки и выполняют различные вычисления и операции. CNN работает с фильтрами (свертками), которые определяют характеристики изображений. Свертка проходится по изображению и распознает существует ли некая искомая характеристика в определенном месте этого изображения [4].

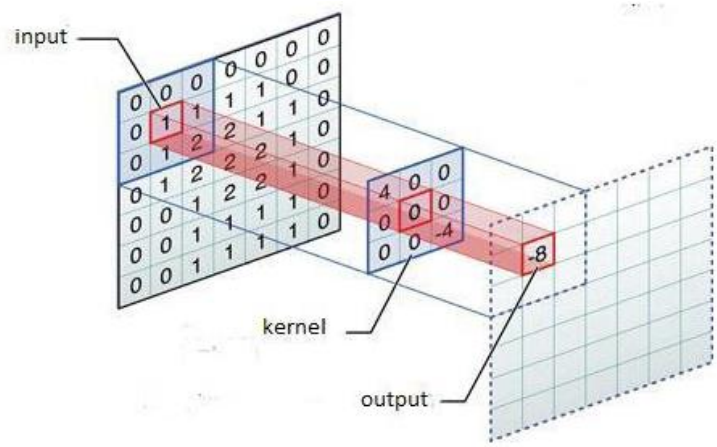


Рисунок-1.5.2 – Свертка нейронной сети

В моментах, когда эта искомая характеристика находится в части изображения, на выходе процесса свертки получается значение со сравнительно большим коэффициентом подобия. В случае если характеристика отсутствует, на выходе будет маленькое значение коэффициента [4].

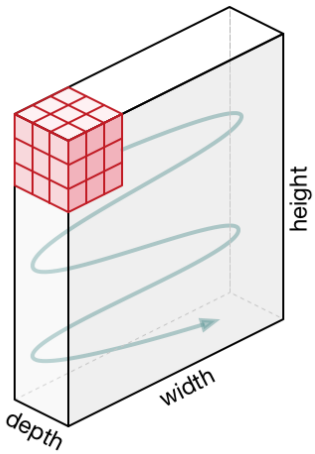


Рисунок-1.5.3 – Перемещение свертки

1.6 Архитектура VGG16

Сверточная нейронная сеть, также популярная как ConvNet, является одной из разновидностью нейронных сетей. VGG16 – это тип CNN, который является одной из наилучшей моделей компьютерного зрения. Создатели этой модели заменили большие фильтры на несколько маленьких размером 3x3, идущих друг за другом, тем самым увеличив показатели результатов по сравнению с другими моделями. На входном слое conv1 считываются RGB изображения размером 224x224. После изображения проходят через стек фильтров. Далее идут три полносвязных

слоя: каждый из двух первых слоев имеют 4096 каналов, а третий 1000 каналов. Последним идет softmax слой [5, 6].

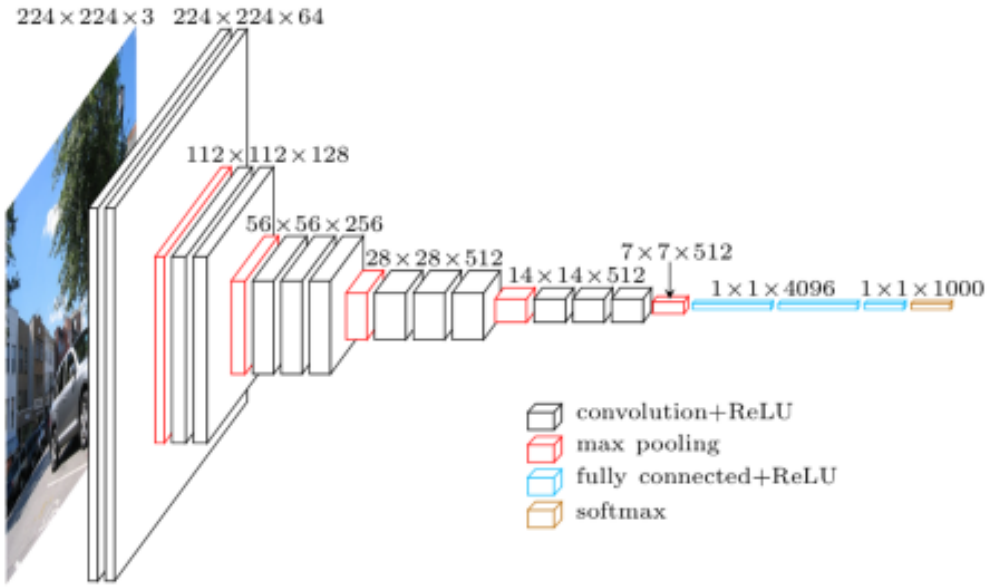


Рисунок-1.6 – Архитектура модели VGG16

1.7 Архитектура ResNet50

Residual Network 50 (ResNet50) – остаточная нейронная сеть с 50-ю слоями. Она была обучена на наборе данных ImageNet, объём которого составлял 1.2 миллиона изображений, разделенных на 1000 категорий. Residual Learning переводится как «остаточное обучение», когда слои стремятся корректировать ошибки, допущенные слоями до него, вместо того чтобы просто изменять входы предыдущих слоев [7].

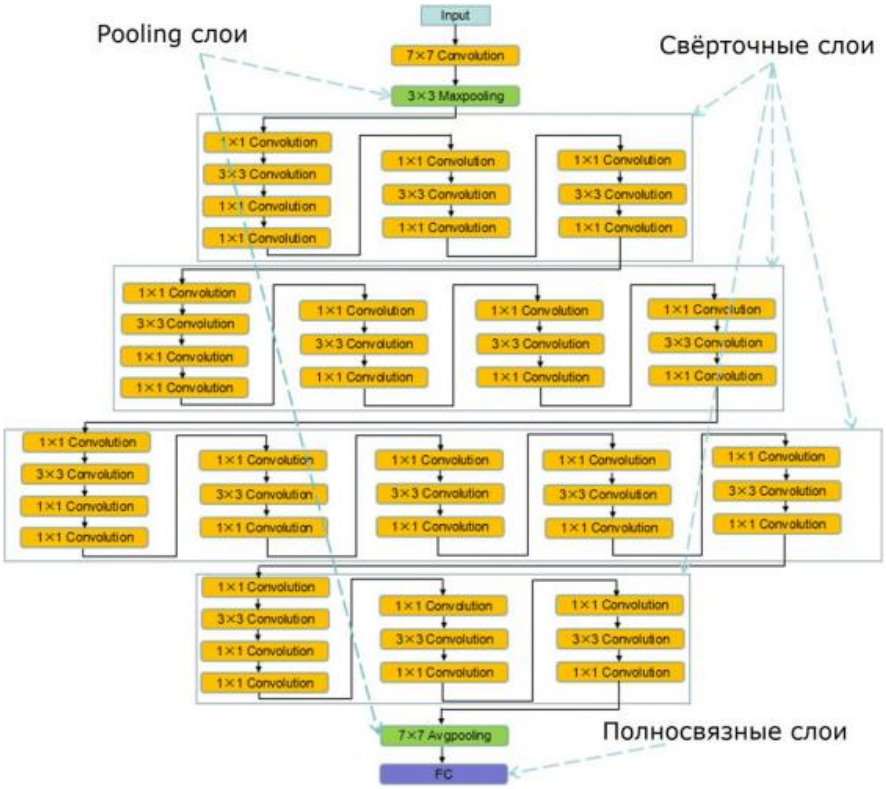


Рисунок-1.7 – Архитектура модели ResNet50

1.8 Архитектура InceptionV3

InceptionV3 – это сверточная нейронная сеть от компании Goggle. Сеть состоит из симметричных и асимметричных слоев, в том числе сверток, среднего пула, максимального пула, конкататов, выпадений и полносвязных слоев [8].

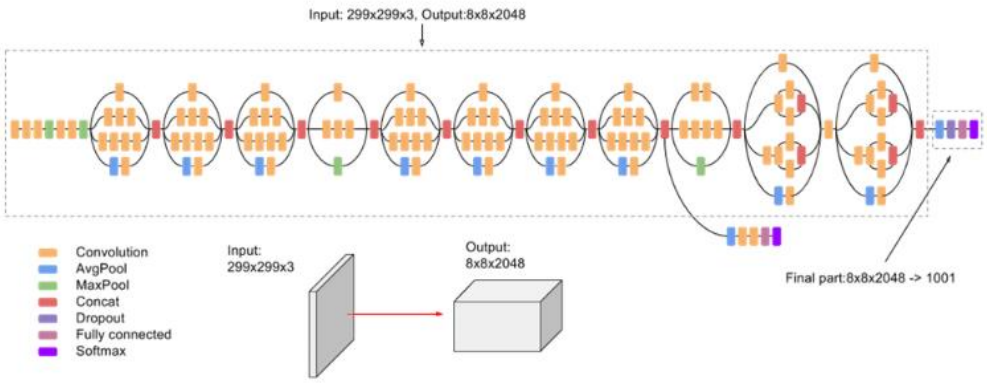


Рисунок-1.8 – Архитектура InceptionV3

2 Технологический раздел

2.1 Исходные данные

Сравнительный анализ проводился на наборе данных цветов. Датасет состоит из 3670 изображений, поделенных на 5 классов: розы, тюльпаны, маргаритки, одуванчики, подсолнухи. Класс роз включает в себя 641 изображений, тюльпаны – 799 изображений, маргаритки – 633, одуванчики – 898, подсолнухи – 699 изображений.

2.2 Python

В качестве языка программирования был выбран Python. Объясняется это тем, что данный язык считается одним из ведущим языком программирования для анализа данных, во многом благодаря своим библиотекам и фреймворкам. Также питон является довольно лаконичным и простым в написании кода. У данного языка огромное сообщество, что является непосредственным плюсом. Библиотеки, использованные в анализе: numpy, pandas, matplotlib. Также был использован пакет sklearn, для непосредственно самой кластеризации

Рисунок-2.2.1 – Топ 5 языков программирования на 2022 год

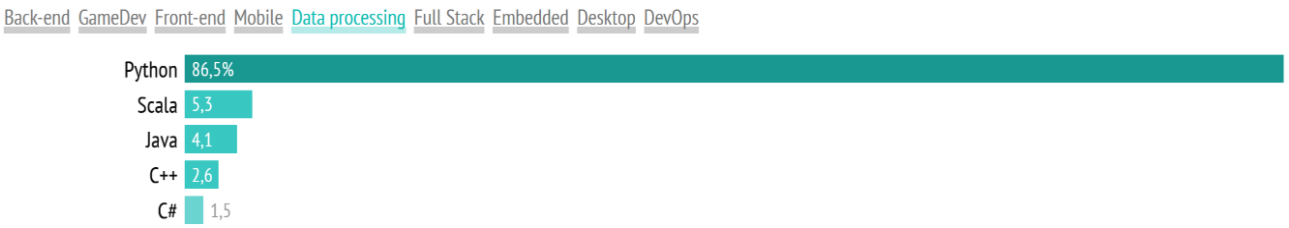


Рисунок-2.2.2 – Топ 5 языков программирования для обработки данных на 2022 год

Как показано на рисунках 2.2.1 и 2.2.2 Python занимает одно из лидирующих мест в разработке, а в обратке данных занимает твердое первое место.

2.3 Google Colaboratory

Весь код писался в среде разработки Colaboratory (Colab). Это облачный сервис от компании Google, который позволяет запускать Jupyter Notebook удаленно через браузер. Colab работает на высокопроизводительных серверах Google, и также имеет модули GPU. GPU ускоряет вычислительные процессы при обучении нейронных сетей, а также при больших объемах данных. Все файлы хранятся в Google Disk с расширением ipynb. Данные загружаются как с локальной машины, так и с гугл диска.

2.4 Tensorflow

Tensorflow – это библиотека для глубокого обучения разработанная Google. Библиотека уже предустановлена в Colab. Tensorflow используется для построения и обучения нейронных сетей.

3 Проектная часть

3.1 Общая архитектура проекта

Архитектуру проекта можно визуально отобразить с помощью четырех блоков:

- Блок с входными данными, изображениями;
- Блок с предобработкой данных;
- Блок с кластеризацией данных;
- Блок с высчитыванием метрик.

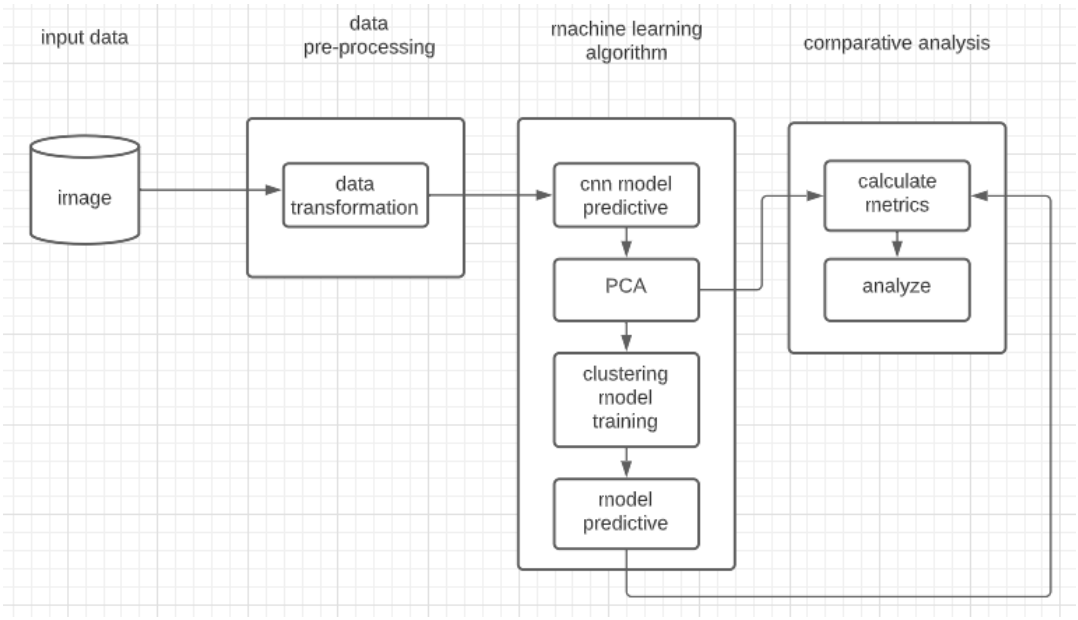


Рисунок-3.1.1 – Архитектура проекта

В первую очередь, на вход подаются изображения. Далее, для дальнейшей работы с ними, необходимо провести обработку данных (изменить размер изображений, нормализовать данные). После обработки, данные проходят через модель нейронной сети для изъятия характеристик изображений. Для получения характеристик, у модели нейронной сети удаляются верхние (классификационные) слои. Поскольку на выходе получается большой размер данных, используется метод главных компонентов для сжатия данных. Сжатые данные проходят через модель кластеризации и на выходе имеем список кластеров. В конце вычисляются качественные метрики кластеризации и на основе полученных результатов метрики проводится сравнительный анализ.

3.2 Диаграмма активности

Диаграмма активности используется для визуализации последовательных и параллельных действий в системе. Ее главная цель просто и наглядно показать, что должно происходить в системе. Благодаря диаграмме активности наглядно виден и понятен алгоритм работы системы

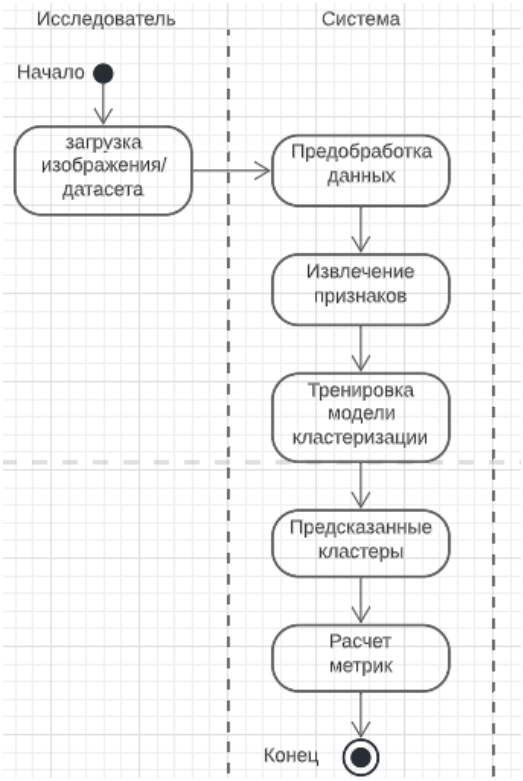


Рисунок-3.1.2 – Диаграмма активности

Благодаря диаграмме активности, изображенной на рисунке 3.1.2 можно увидеть работу алгоритма кластеризации изображений. Первым делом исследователь загружает набор изображений для кластеризации. Далее система сама эти данные прогоняет через несколько этапов: предобработка данных, извлечение признаков, использование модели кластеризации, расчет метрика. Благодаря полученным метрикам производится сравнительный анализ алгоритмов кластеризации.

3.6 Применение нейронной сети в кластеризации

Для лучших результатов, кластеризацию используют в связке с нейронными сетями. Изображение подается на вход к предобученной модели нейронной сети (CNN). Обучались модели на наборе ImageNet состоящая из 1.2 миллиона изображений. Модель преобразует входные данные в вектор признаков и пропускает их через свои скрытые слои [9, 10]. Переходя из одного

скрытого слоя в другой размер изображения с каждым разом уменьшается, но увеличивается размер признаков.

Последний классификационный слой удаляется и в результате выводит вектор-признаки изображения. Эти наборы признаков позже подаются через модели кластеризации.

4 Экспериментальный раздел

4.1 Входные данные и предобработка

В качестве входных данных выступают изображения из набора данных flowers.

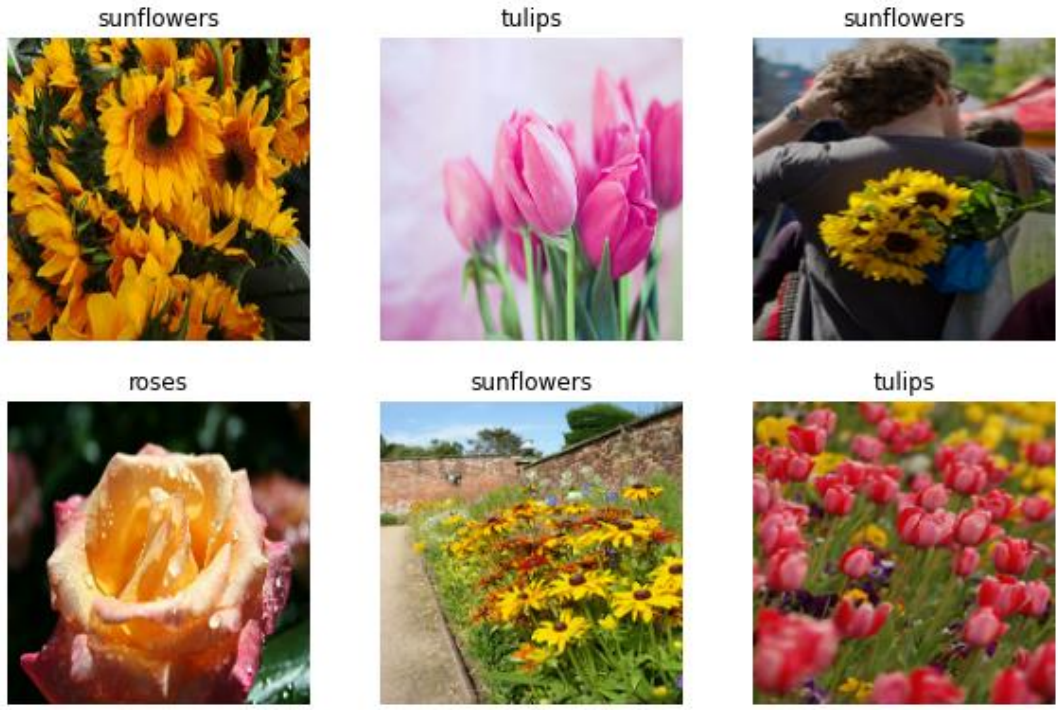
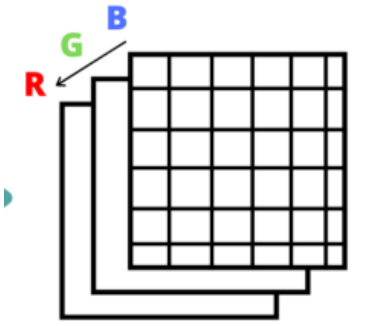


Рисунок-4.1.1 – Визуализация изображений (6) из набора данных

В самом начале массив имеет только 3 измерения (ширина, высота, канал), и т.к. модель работает с пакетами выборок нужно расширить массив, чтобы добавить измерение, которое позволит модели узнать, сколько изображений ей дается. На вход модели нейронной сети подается изображение размером (3670, 180, 180, 3) (количество изображений, ширина, высота, глубина). Данные также были предобработаны путем нормализации. Теперь значения варьируются в диапазоне от 0 до 1 включительно.



Color channels representation

Рисунок-4.1.2 - Представление цветового канала

4.2 Модели нейронной сети

Теперь, когда данные прошли предобработку [11, 12], их можно пропускать через готовые, предобученные модели сверточных нейронных сетей. Модели обучались на наборе данных ImageNet, которая состоит из 14 млн изображений, относящихся к 21 841 категориям. Из моделей удаляются верхние (классифицированные) слои, для получения векторов-признаков.

```
vgg16 = VGG16()
vgg16 = Model(inputs = vgg16.inputs, outputs = vgg16.layers[-2].output)

inc_v3 = InceptionV3()
inc_v3 = Model(inputs = inc_v3.inputs, outputs = inc_v3.layers[-2].output)

resnet = ResNet50()
resnet = Model(inputs = resnet.inputs, outputs = resnet.layers[-2].output)
```

Рисунок-4.2 Модели CNN

4.3 Метод главных компонент

Для сокращения размерности признаков изображений с минимальной потерей полезной информации, используют технологию PCA [13].

```
VGG components before PCA : 4096
VGG components after PCA: 603

InceptionV3 components before PCA : 2048
InceptionV3 components after PCA: 603

ResNet50 components before PCA : 2048
ResNet50 components after PCA: 603
```

Рисунок-4.3 – Количество признаков до/после прохождения через PCA

4.4 Обучение моделей кластеризации

После сжатия данных они готовы пройти через модель кластеризации. Обучение происходит благодаря методу fit().

```
vgg_kmeans =  KMeans(n_clusters=10).fit(vgg_pca_output)
vgg_birch = Birch(n_clusters=10).fit(vgg_pca_output)
vgg_agg = AgglomerativeClustering(n_clusters=10).fit(vgg_pca_output)
vgg_af = AffinityPropagation().fit(vgg_pca_output)
```

Рисунок-4.4 – Модели кластеризации НС VGG16

Как видно на рисунке 4.4 в качестве параметра передаются сжатые PCA данные.

4.5 Использованные метрики

Для анализа алгоритмов кластеризации необходимо вывыести качественную оценку. В своей работе я использовала 3 метрики для оценки:

- Silhouette Score
- Davies-Bouldin
- Calinski-Harabasz

Silhouette Score показывает качества метода кластеризации. Значение оценки варьируется в диапазоне от -1 до 1. Коэффициент 1 указывает на то, что кластеры расположены далеко и четко различимы. Коэффициент 0 говорит о то, что расстояние между кластерами не существенно, а коэффициент -1 указывает на неправильную кластеризацию выборок. Вычисляется коэффициент как с

среднее внутрикластерное расстояние на среднее расстояние до ближайшего кластера по каждому образцу:

$$silhouette = \frac{b - a}{\max(a, b)},$$

где b - расстояния до ближайшего кластера;

a - внутрикластерное расстояние [14, 15, 16].

Метрика Дэвис-Боулдина (*davis-bouldin)*.Компактность вычисляется расстоянием объектов кластера до центроидов. Отделимость высчитывается расстоянием между центроидами кластеров. Метрика принимает значение больше 0 и является средним отношением внутрикластерных разбросов к расстояниям между кластерами. Низкий показатель метрики говорит о лучшем разделении между кластерами [14, 17].

Calinski-Harabasz также распространенный как оценка отношения дисперсиия, применятся для оценки модели. Высокий коэффициент принадлежит моделям с наиболее определенными кластерами [14, 17].

```
################ For VGG16 ################
For K-means:
Silhouette Score: 0.07086873799562454
Davies-Bouldin Score: 2.691264577994461
Calinski-Harabasz Score: 25.60353186051561

For BIRCH:
Silhouette Score: 0.05937385931611061
Davies-Bouldin Score: 2.7603769279438986
Calinski-Harabasz Score: 23.76491532029737

For Agglomerative:
Silhouette Score: 0.05937385931611061
Davies-Bouldin Score: 2.7603769279438986
Calinski-Harabasz Score: 23.76491532029737

For Affinity:
Silhouette Score: 0.06205463036894798
Davies-Bouldin Score: 2.1963440157895535
Calinski-Harabasz Score: 9.159278847308968
```

Рисунок-4.5 – Вывод метрик для VGG16 в программе

4.6 Результаты кластеризации

4.6.1 Визуализация результатов

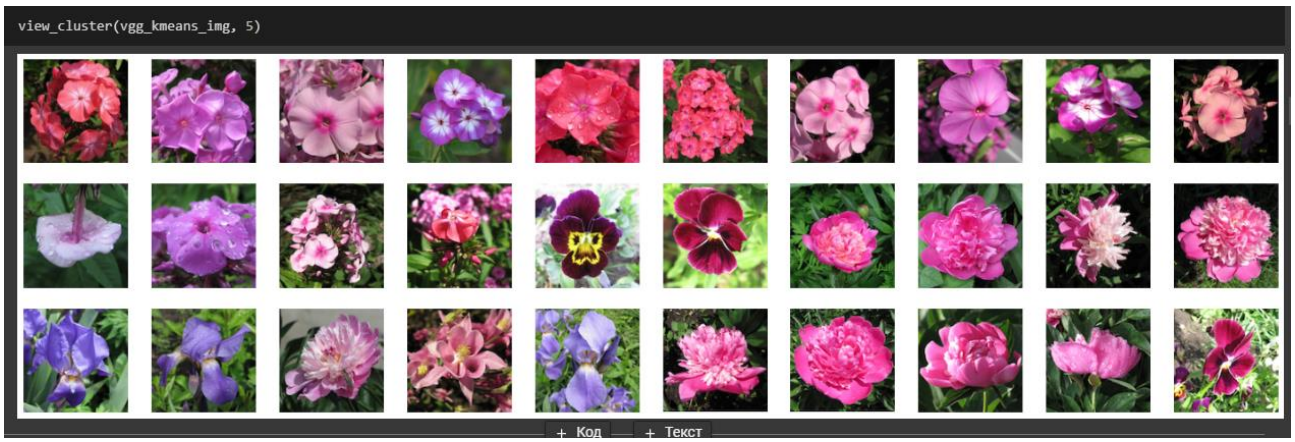


Рисунок-4.6.1.1 – Алгоритм k-means для модели VGG16

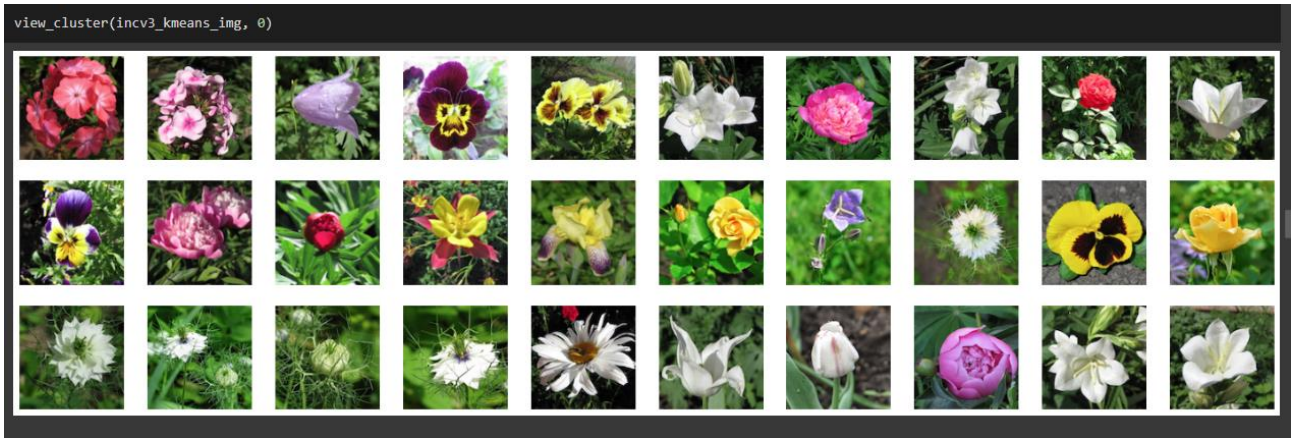


Рисунок-4.6.1.2 – Алгоритм k-means для модели InceptionV3

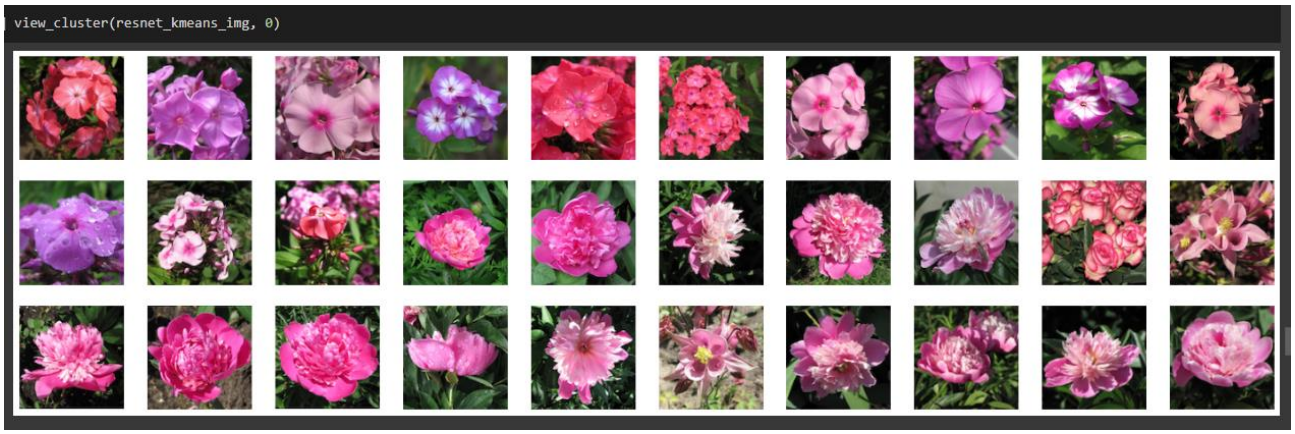


Рисунок-4.6.1.3 – Алгоритм k-means для модели ResNet50

Рисунок-4.6.1.4 – Алгоритм agglomerative для модели VGG16

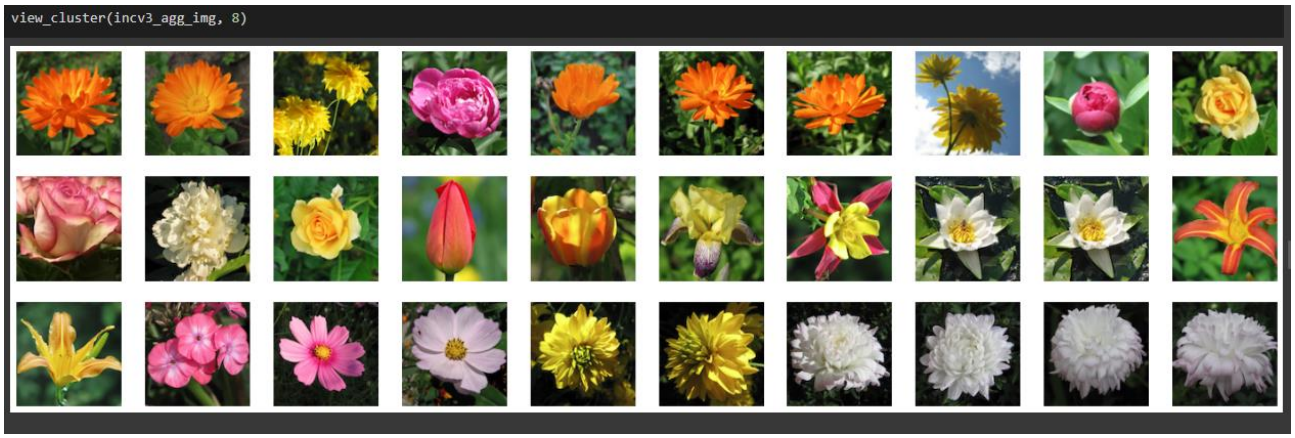


Рисунок-4.6.1.5 – Алгоритм agglomerative для модели InceptionV3

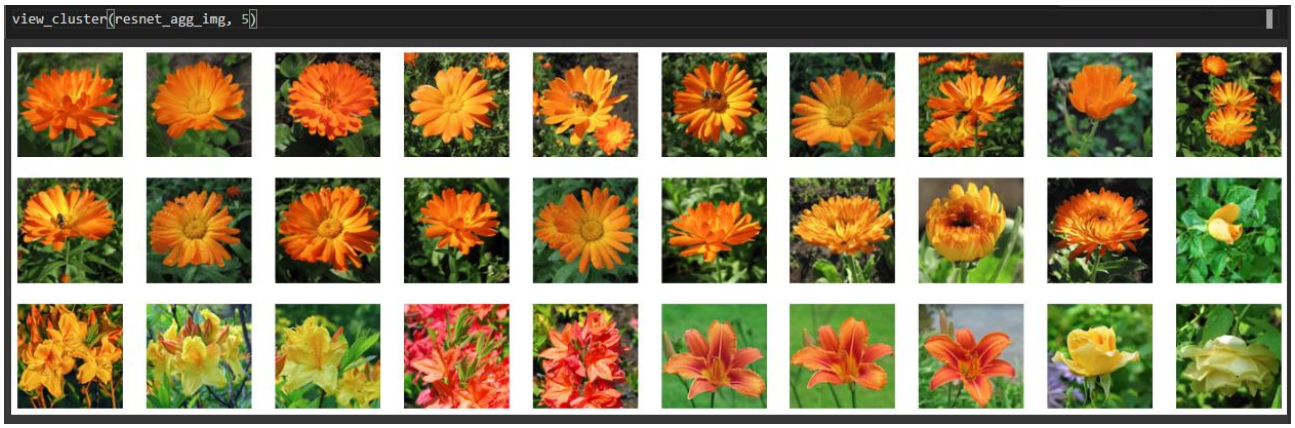


Рисунок-4.6.1.6 – Алгоритм agglomerative для модели ResNet50

На рисунках выше можно увидеть работу алгоритмов k-means и agglomerative для разных моделей нейронной сети (VGG16, InceptionV3, ResNet50). Как можно заметить алгоритмы, действительно, кластеризуют изображения не по определенному классу (тюльпаны, розы, ромашки и т.д.), как

классификатор, а по качественным признакам изображения (цвета, линии, углы и т.д.)

4.6.2 Сравнительный анализ

В таблице 2 приведены результаты работы алгоритмов кластеризации посчитанных с помощью внутренних метрик качества.

Таблица 2. Качественная оценка алгоритмов кластерного анализа

	Алгоритм	**Silhouette Score**	**Davies-Bouldin**	**Calinski-Harabasz**
VGG16	k-means	0.06962	2.67177	25.33340
	BIRCH	0.05937	2.76038	23.76492
	Agglomerative	0.05937	2.76038	23.76492
	Affinity Propagation	0.06205	2.19634	9.15928
ResNet	k-means	0.08635	2.58393	28.64442
	BIRCH	0.07483	2.75572	26.57498
	Agglomerative	0.07483	2.75572	26.57498
	Affinity Propagation	0.08196	2.19723	10.80910
InceptionV3	k-means	0.12184	2.49073	140.48422
	BIRCH	0.07237	2.51595	131.09730
	Agglomerative	0.07237	2.51595	131.09730
	Affinity Propagation	0.05090	1.90128	39.83761

ЗАКЛЮЧЕНИЕ

В рамках данного дипломного проекта поставленная цель была выполнена. В ходе работы были рассмотрены методы кластеризации. Была произведена подготовка данных к анализу посредством их нормализации, а также сжатие данных методом PCA.

Модели, использованные при кластеризации:

- k-means;
- BIRCH;
- agglomerative clustering;
- affinity propagation.

Также были изучены и проанализированы модели нейронной сети CNN, такие как:

- VGG16;
- ResNet50;
- InceptionV3.

По результатам сравнительного анализа, основанного на значениях метрик, не совсем корректно высказываться, что один метод кластерного анализа превосходит другой. Связано это с тем, что каждый алгоритм обусловлен своими внутренними метриками для вычислений, также не существует универсального алгоритма кластеризации, поскольку у каждого алгоритма есть своя специфическая область применения.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Кластеризация [Электронный ресурс] – Режим доступа: https://scikit-learn.ru/clustering, свободный.

2 Кластеризация [Электронный ресурс] – Режим доступа: https://blog.skillfactory.ru/glossary/klasterizacziya-klasternyj-analiz, свободный.

3 Кораблев Н.М., Фомичев А.А. Кластеризация данных методом k-means с использованием искусственных иммунных систем // Бионика Интеллекта. – 2011 - №3 – С. 102-106

4 Богдан Б., Сверточные нейронные сети с нуля [Электронный ресурс] – Режим доступа: https://medium.com/@balovbohdan/4d5a1f0f87ec, свободный.

5 Милютин И., VGG16 — сверточная сеть для выделения признаков изображений [Электронный ресурс] – Режим доступа: https://neurohive.io/ru/vidy-nejrosetej/vgg16-model, свободный.

6 Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognitional. ICLR 2015

7 Литвинов С. ResNet (34, 50, 101): «остаточные» CNN для классификации изображений [Электронный ресурс] – Режим доступа: https://neurohive.io/ru/vidy-nejrosetej/resnet-34-50-101, свободный.

8 Диагностика меланомы с использованием нейронной сети InceptionV3 / Д.А. Гаврилов, А.В. Мелерзанов, Н.Н. Щелкунов, Э.И. Закиров

9 Xu Ji, F. Henriques, Andrea Vedaldi. Invariant Information Clustering for Unsupervised Image Classification and Segmentation

10 CNN features are also great at unsupervised classification / Joris Gu´erin, Olivier Gibaru, St´ephane Thiery, and Eric Nyiri.

11 Y. Peng, X. He, and J. Zhao. Object-part attention model for fine-grained image classification. IEEE Transactions on Image Processing, 27(3):1487–1500, March 2018.

12 Image clustering using Transfer learning [Электронный ресурс] – Режим доступа: https://towardsdatascience.com/image-clustering-using-transfer-learning-df5862779571, свободный.

13 Метод главных компонент (Principal component analysis) [Электронный ресурс] – Режим доступа: https://wiki.loginom.ru/articles/principal-component-analysis.html, свободный.

14 Сивоголовко Е.В. Методы оценки качества четкой кластеризации // Компьютерные инструменты в образовании, 2011.

15 Bhardwaj A., Silhouette Coefficient [Электронный ресурс] – Режим доступа: https://towardsdatascience.com/silhouette-coefficient-validating-clustering-techniques-e976bb81d10c#:~:text=Silhouette, свободный.

16 Кластеризуем лучше, чем «метод локтя» [Электронный ресурс] – Режим доступа: https://habr.com/ru/company/jetinfosystems/blog/467745, свободный.

17 Оценка эффективности кластеризации [Электронный ресурс] – Режим доступа: https://scikit-learn.ru/clustering/#clustering-performance-evaluation, свободный.

ПРИЛОЖЕНИЕ А
(обязательное)

Техническое задание

А.1.5 Техническое задание на сравнительный анализ методов кластеризации в решении задач распознавания.

Настоящее техническое задание распространяется на сравнительный анализ методов кластеризации с использование машинного обучения и нейронных сетей.

А.1.5.1 Основание для разработки

Сравнительный анализ разрабатывается на основании устного распоряжения дипломного руководителя.

А.1.5.2 Назначение

Назначение анализа – это выявление общих метрик кластеризации для дальнейшего их сравнения. По результатам анализа выявляется наиболее эффективный алгоритм кластерного анализа для распознавания изображений.

А.1.5.3 Требования к функциональным характеристикам

В ходе анализа выполняются следующие функции:

- подача на вход изображения jpg, png форматов;
- предобработка данных;
- изъятие вектор-признаков из изображения;
- сжатие размера данных;
- кластеризация изображений;
- сравнение и анализ полученных метрик кластерного анализа;
- визуализация кластеров.

А.1.5.4 Требования к надежности

Продолжение приложения А

На выходе, в результате сравнительного анализа, должна выводиться достоверная информация.

ПРИЛОЖЕНИЕ Б
(обязательное)

Текст программы

```
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import preprocess_input

# for cnn models
from keras.applications.vgg16 import VGG16
from tensorflow.keras.applications.resnet50 import ResNet50
from keras.applications.inception_v3 import InceptionV3
from keras.models import Model

# for clustering
from sklearn.cluster import KMeans
from sklearn.cluster import Birch
from sklearn.cluster import AgglomerativeClustering
from sklearn.cluster import AffinityPropagation
from sklearn.decomposition import PCA

#for metrics
from sklearn.metrics import silhouette_score
from sklearn.metrics import davies_bouldin_score
from sklearn.metrics import calinski_harabasz_score

import os
import numpy as np
import matplotlib.pyplot as plt
from random import randint
import pandas as pd
import pickle
import glob
from pathlib import Path

dir = Path('/content/drive/MyDrive/diploma/flowers')

flowers = []
for files in dir.glob('*.png'):
    flowers.append(files)

vgg16 = VGG16()
```

```
vgg16 = Model(inputs = vgg16.inputs, outputs = vgg16.layers[-2].output)

inc_v3 = InceptionV3()
inc_v3 = Model(inputs = inc_v3.inputs, outputs = inc_v3.layers[-2].output)
resnet = ResNet50()
resnet = Model(inputs = resnet.inputs, outputs = resnet.layers[-2].output)

def get_features(file, model, weight, height):
    img = load_img(file, target_size=(weight,height))
    img = np.array(img)
    reshaped_img = img.reshape(1,weight,height,3)
    imgx = preprocess_input(reshaped_img)
    features = model.predict(imgx)
    return features

def get_list(model,weight,height):
  data = {}
  for flower in flowers:
        ft = get_features(flower,model,weight, height)
        data[flower] = ft
  filenames = np.array(list(data.keys()))
  ft = np.array(list(data.values()))
  ft = ft.reshape(ft.shape[0],ft.shape[2])
  return filenames, ft

incv3_file, incv3_feat = get_list(inc_v3, 299, 299)
vgg_file, vgg_feat = get_list(vgg16, 224, 224)
resnet_file, resnet_feat = get_list(resnet, 224, 224)

def create_pca(data):
    pca = PCA()
    pca_result = pca.fit_transform(data)
    return pca_result

vgg_pca_output = create_pca(vgg_feat)
incv3_pca_output = create_pca(incv3_feat)
resnet_pca_output = create_pca(resnet_feat)

print(f"VGG components before PCA : {vgg_feat.shape[1]}")
print(f"VGG components after PCA: {vgg_pca_output.shape[1]}")
print(f"\nInceptionV3 components before PCA : {incv3_feat.shape[1]}")
print(f"InceptionV3 components after PCA: {incv3_pca_output.shape[1]}")
```

```
print(f"\nResNet50 components before PCA : {resnet_feat.shape[1]}")
print(f"ResNet50 components after PCA: {resnet_pca_output.shape[1]}")

def grouping(files, labels, arr):
  for file, cluster in zip(files, labels):
      if cluster not in arr.keys():
          arr[cluster] = []
          arr[cluster].append(file)
      else:
          arr[cluster].append(file)

def view_cluster(files_group,label):
    plt.figure(figsize = (25,25));
    files = files_group[label]
    if len(files) > 30:
        files = files[:30]
    for index, file in enumerate(files):
        plt.subplot(10,10,index+1);
        img = load_img(file)
        img = np.array(img)
        plt.imshow(img)
        plt.axis('off')

vgg_kmeans =  KMeans(n_clusters=10).fit(vgg_pca_output)
vgg_birch = Birch(n_clusters=10).fit(vgg_pca_output)
vgg_agg = AgglomerativeClustering(n_clusters=10).fit(vgg_pca_output)
vgg_af = AffinityPropagation().fit(vgg_pca_output)
vgg_kmeans_img = {}
vgg_birch_img = {}
vgg_agg_img = {}
vgg_af_img = {}
grouping(vgg_file, vgg_kmeans.labels_, vgg_kmeans_img)
grouping(vgg_file, vgg_birch.labels_, vgg_birch_img)
grouping(vgg_file, vgg_agg.labels_, vgg_agg_img)
grouping(vgg_file, vgg_af.labels_, vgg_af_img)

incv3_kmeans =  KMeans(n_clusters=10).fit(incv3_pca_output)
incv3_birch = Birch(n_clusters=10).fit(incv3_pca_output)
incv3_agg = AgglomerativeClustering(n_clusters=10).fit(incv3_pca_output)
incv3_af = AffinityPropagation().fit(incv3_pca_output)
incv3_kmeans_img = {}
incv3_birch_img = {}
```

```
incv3_agg_img = {}
incv3_af_img = {}
grouping(incv3_file, incv3_kmeans.labels_, incv3_kmeans_img)
grouping(incv3_file, incv3_birch.labels_, incv3_birch_img)
grouping(incv3_file, incv3_agg.labels_, incv3_agg_img)
grouping(incv3_file, incv3_af.labels_, incv3_af_img)

resnet_kmeans =  KMeans(n_clusters=10).fit(resnet_pca_output)
resnet_birch = Birch(n_clusters=10).fit(resnet_pca_output)
resnet_agg = AgglomerativeClustering(n_clusters=10).fit(resnet_pca_output)
resnet_af = AffinityPropagation().fit(resnet_pca_output)
resnet_kmeans_img = {}
resnet_birch_img = {}
resnet_agg_img = {}
resnet_af_img = {}
grouping(resnet_file, resnet_kmeans.labels_, resnet_kmeans_img)
grouping(resnet_file, resnet_birch.labels_, resnet_birch_img)
grouping(resnet_file, resnet_agg.labels_, resnet_agg_img)
grouping(resnet_file, resnet_af.labels_, resnet_af_img)

def metrics(pca_output,kmeans,birch,agg,af):
  print('For K-means:')
  print(f'Silhouette Score: {silhouette_score(pca_output, kmeans)}')
  print(f'Davies-Bouldin Score: {davies_bouldin_score(pca_output, kmeans)}')
  print(f'Calinski-Harabasz Score: {calinski_harabasz_score(pca_output, kmeans)}')

  print('\nFor BIRCH:')
  print(f'Silhouette Score: {silhouette_score(pca_output, birch)}')
  print(f'Davies-Bouldin Score: {davies_bouldin_score(pca_output, birch)}')
  print(f'Calinski-Harabasz Score: {calinski_harabasz_score(pca_output, birch)}')

  print('\nFor Agglomerative:')
  print(f'Silhouette Score: {silhouette_score(pca_output, agg)}')
  print(f'Davies-Bouldin Score: {davies_bouldin_score(pca_output, agg)}')
  print(f'Calinski-Harabasz Score: {calinski_harabasz_score(pca_output, agg)}')

  print('\nFor Affinity:')
  print(f'Silhouette Score: {silhouette_score(pca_output, af)}')
  print(f'Davies-Bouldin Score: {davies_bouldin_score(pca_output, af)}')
  print(f'Calinski-Harabasz Score: {calinski_harabasz_score(pca_output, af)}')

print('################ For VGG16 ################')
```

```
metrics(vgg_pca_output,vgg_kmeans.labels_,vgg_birch.labels_,vgg_agg.labels_,vgg_af.labels_)

print('################ For InceptionV3 ################')
metrics(incv3_pca_output,incv3_kmeans.labels_,incv3_birch.labels_,incv3_agg.labels_,incv3_af.labels_)

print('################ For ResNet50 ################')
metrics(resnet_pca_output,resnet_kmeans.labels_,resnet_birch.labels_,resnet_agg.labels_,resnet_af.labels_)
```