|  |  |
| --- | --- |
|  | **Учебно-Научно Производственный Комплекс**  Международный Университет Кыргызстана  Кыргызстан Эл Аралык Университети  International University of Kyrgyzstan |

**Кафедра Компьютерных информационных систем и управления**

Курсовая работа

**Тема**: Разработка нейросетевой системы распознавания объектов

**Группа: ИВТ 1-21**

**Исполнитель: Узакбаев У.У.**

**Руководитель: Мусакулова Ж.А.**

Бишкек 2023

**ОГЛАВЛЕНИЕ**

[1. Постановка задачи 3](#_Toc135653808)

[2. Назначение и область применения программного продукта 5](#_Toc135653809)

[3. Выбор и обоснование выбора инструментальных средств 7](#_Toc135653810)

[4. Теоретический материал – особенности программирования в данной области 9](#_Toc135653811)

[5. Структурное описание разработки 10](#_Toc135653812)

[6. Описание программы 20](#_Toc135653813)

[7. Описание алгоритмов и методов решения. Функциональное описание. 21](#_Toc135653814)

[8. Описание пользовательского интерфейса 23](#_Toc135653815)

[9. Руководство пользователя 25](#_Toc135653816)

[10. Заключение 27](#_Toc135653817)

[11. Список используемой литературы 28](#_Toc135653818)

# Постановка задачи

Введение:

Распознавание объектов на изображениях является одной из ключевых задач в области компьютерного зрения. С возрастанием доступности больших объемов данных и развитием глубокого обучения, нейронные сети стали мощным инструментом для автоматического распознавания и классификации объектов на изображениях. Это привело к развитию нейросетевых систем распознавания, способных с высокой точностью и скоростью обрабатывать визуальную информацию.

Анализ проблемы:

В задаче распознавания объектов на изображениях и других медиа материалов как видео, существуют несколько сложностей, включая вариации в положении, размере и форме объектов, наличие шума и перекрытие объектов друг с другом. Для решения этих проблем требуется использование мощных инструментов и алгоритмов компьютерного зрения.

Цели и задачи:

Целью данной работы является разработка эффективной нейросетевой системы для распознавания объектов на изображениях и видеоматериалах. Также с возможностью подключения к камере и распознавания объектов в режиме реального времени. В итоге мы должны продемонстрировать работу нейронных сетей распознавания и их возможное применение.

Для достижения этой цели ставятся следующие задачи:

* Изучить основы нейронных сетей и алгоритмов распознавания объектов.
* Провести анализ и выбрать наиболее подходящие архитектуры нейронных сетей для задачи распознавания объектов.
* Собрать и подготовить набор данных, содержащий изображения с различными объектами.
* Разработать и обучить нейросетевую модель на основе выбранных архитектур с использованием набора данных.
* Оценить производительность и точность разработанной модели на тестовом наборе данных.
* Интегрировать разработанную модель в графический пользовательский интерфейс для удобного использования.

Методология:

Для разработки нейросетевой системы распознавания объектов будет использоваться следующая методология:

* Изучение основных понятий и принципов нейронных сетей, включая сверточные нейронные сети (CNN), рекуррентные нейронные сети (RNN) и архитектуры сегментации объектов.
* Исследование существующих алгоритмов и моделей для распознавания объектов, таких как YOLO (You Only Look Once), Mask R-CNN (Facebook AI Research) и Pixellib или ImageAI.
* Анализ особенностей и преимуществ каждой модели, их способности к точному локализации объектов и высокой скорости обработки.
* Выбор наиболее подходящей модели, учитывая требования к точности, производительности и доступность пред обученных весов моделей.
* Разработка системы нейросетевой модели.
* Интеграция модели в графический интерфейс для удобного пользования

# Назначение и область применения программного продукта

Наш программный продукт - нейросетевая система распознавания объектов. Программа представляет собой интерактивное окно с выбором задач на выполнение и загрузкой данных. Программа будет полезна для начинающих студентов, для ознакомления с нейронными сетями. Преимущества: простота и удобное использование.

Назначение нейросетевой системы распознавания заключается в автоматическом обнаружении и классификации объектов на изображениях с использованием алгоритмов глубокого обучения. Основной целью системы является распознавание и идентификация объектов на основе их визуальных характеристик, что позволяет автоматизировать процессы анализа изображений, сократить время и усилия, требуемые для ручной обработки данных.

Основные задачи, решаемые нейросетевой системой распознавания, включают:

* Детектирование объектов: Система может обнаруживать и локализовывать объекты на изображении, указывая их границы или охватывающие прямоугольники.
* Классификация объектов: Система может определить, к какому классу принадлежит объект, например, определить, что на изображении изображен автомобиль, кошка или стул.
* Сегментация объектов: Система может разделить изображение на сегменты, относящиеся к различным объектам, что позволяет более точно определить границы и маски объектов.
* Распознавание лиц и эмоций: Система может идентифицировать лица на изображениях и даже определять эмоциональное состояние людей на основе их выражений.
* Автоматическая индексация и анализ изображений: Система может использоваться для индексации и классификации больших наборов изображений, что позволяет упростить поиск и анализ графического контента.

Программный продукт может быть применен во многих областях, включая:

* Компьютерное зрение и робототехнику:

Такая система может использоваться для распознавания объектов на изображениях, что является важным компонентом для автономных роботов, систем видеонаблюдения, а также в области робототехники для выполнения задач, связанных с восприятием окружающей среды.

* Автоматическое вождение:

Распознавание объектов является ключевым компонентом систем автоматического вождения, позволяющим автомобилю обнаруживать и классифицировать различные объекты на дороге, такие как автомобили, пешеходы, дорожные знаки и светофоры.

* Системы безопасности:

Наша система может быть использована для обнаружения и распознавания объектов в системах видеонаблюдения и обеспечения безопасности. Это позволяет эффективно отслеживать нежелательную активность, наприме, обнаружение движения на охраняемых объектах или распознавание лиц на видеозаписях.

* Медицинская диагностика:

Распознавание объектов может быть применено в области медицинской диагностики, например, для автоматического обнаружения и классификации структур на медицинских изображениях, включая рентгеновские снимки, магнитно-резонансные томограммы (МРТ) и компьютерную томографию (КТ).

Это лишь несколько примеров областей, в которых наша нейросетевая система распознавания объектов может быть применена. Ее использование также может быть расширено в другие области, где требуется распознавание объектов на изображениях.

Важно отметить, интерфейс Tkinter обеспечивает удобный способ взаимодействия с разработанной системой, позволяя пользователю загружать изображения, запускать процесс распознавания объектов и получать результаты в удобном формате. Наша система предоставляет гибкое решение для распознавания объектов на изображениях, что открывает новые возможности для автоматизации, повышения эффективности анализа данных.

# Выбор и обоснование выбора инструментальных средств

Выбор инструментальных средств для реализации нашего продукта включает использование следующих компонентов:

* Язык программирования Python: Python является одним из наиболее популярных языков программирования для разработки и исследований в области машинного обучения и нейронных сетей. Он обладает широким набором библиотек и инструментов, которые значительно облегчают разработку и реализацию нейросетевых систем.
* Среда разработки PyCharm: PyCharm - это мощная интегрированная среда разработки (IDE) для Python, предоставляющая редактор кода с подсветкой синтаксиса и автодополнением, отладчик, управление проектом, интеграцию с инструментами и библиотеками, автоматизацию задач, функции рефакторинга и анализа кода. Она позволяет разработчикам эффективно работать над проектами на Python.
* Библиотека Pixellib и ImageAI: Pixellib и ImageAI являются мощными библиотеками компьютерного зрения для Python. Pixellib предоставляет удобные инструменты для сегментации объектов на изображениях, а ImageAI содержит реализации различных алгоритмов машинного обучения для обнаружения и классификации объектов на изображениях. Сочетание этих библиотек позволит эффективно обрабатывать изображения и распознавать объекты на них.
* Модуль Pointrend: Pointrend является модулем, разработанным для сегментации объектов на изображениях с использованием нейронных сетей. Он предоставляет методы для точечного представления объектов и точечной регрессии, что улучшает качество сегментации и распознавания объектов.
* OpenCV: OpenCV (Open Source Computer Vision Library) является мощным инструментом для компьютерного зрения и обработки изображений. Он предоставляет широкий набор функций и алгоритмов для работы с изображениями, включая загрузку, обработку, анализ и отображение. OpenCV будет полезен для предварительной обработки изображений перед их подачей на вход нейросетевой системе.
* Графический интерфейс Tkinter: Tkinter является стандартной библиотекой Python для создания графического пользовательского интерфейса (GUI). Он обладает простым и понятным API, позволяющим создавать окна, кнопки, поля ввода и другие элементы интерфейса. Tkinter удобно использовать для создания пользовательского интерфейса, через который пользователь сможет загружать изображения, запускать процесс распознавания и получать результаты.
* И библиотеку “os” для реализации возможности подключения к файлам и взаимодействия с ними в операционной системе Windows. Чтобы пользователь мог загружать, открывать свои изображения.

Обоснование выбора данных инструментальных средств основывается на их популярности, функциональности и подходящей для задачи распознавания объектов на изображениях. Python является промышленным стандартом для машинного обучения и обработки изображений, а библиотеки Pixellib, ImageAI и OpenCV предоставляют мощные инструменты для работы с изображениями и нейронными сетями. Модуль Pointrend специально разработан для улучшения сегментации объектов на изображениях. Tkinter предоставляет удобный способ создания графического интерфейса для взаимодействия с нейросетевой системой.

Ожидаемые результаты:

Ожидается, что разработанная нейросетевая система будет эффективно распознавать объекты на изображениях, иметь высокую точность и оптимальное время обработки.

# Теоретический материал – особенности программирования в данной области

Программирование в области нейросетевого распознавания объектов на изображениях имеет свои особенности.

* Концепция нейросетевого распознавания объектов: Нейросетевое распознавание объектов на изображениях основано на использовании глубоких нейронных сетей. Ключевыми компонентами являются сверточные нейронные сети (Convolutional Neural Networks, CNN), которые способны выделять важные признаки из изображений и классифицировать объекты. Понимание работы и структуры CNN является важным для разработки и обучения моделей распознавания объектов.
* Обучение нейросетевых моделей: Для успешного распознавания объектов необходимо обучить нейросетевую модель на большом наборе размеченных данных. Процесс обучения включает выбор и настройку архитектуры нейронной сети, определение функции потерь и выбор метода оптимизации. Также требуется разделение данных на обучающую, проверочную и тестовую выборки для оценки производительности модели.
* Аугментация данных: Для улучшения обобщающей способности нейросетевой модели полезно применять аугментацию данных. Это включает в себя применение различных трансформаций к изображениям, таких как повороты, сдвиги, масштабирование и изменение яркости. Аугментация данных помогает модели обучаться на разнообразных вариантах изображений и повышает ее способность к обобщению.
* Выбор функции потерь: Функция потерь определяет, как нейросетевая модель оценивает разницу между предсказанными и фактическими метками. Для задачи распознавания объектов на изображениях распространенными функциями потерь являются категориальная кросс-энтропия или среднеквадратическая ошибка. Выбор подходящей функции потерь влияет на качество и скорость обучения модели.
* Постобработка результатов: После распознавания объектов на изображениях могут быть применены методы постобработки для уточнения результатов или применения дополнительных алгоритмов, например, для фильтрации ложных обнаружений или объединения сегментов объектов.

# Структурное описание разработки

Программный продукт “Нейросетевой системы распознавания” разработан в среде разработки PyCharm с использованием языка программирования Python. Для анализа и распознавания использовались библиотеки Pixellib и ImageAI. А также библиотеки для загрузки и обработки изображений такие как OpenCV и os. Для удобного использования системы интегрируем графический интерфейс Tkinter.

Архитектура программы состоит из : главного окна, строки меню, модуль

вывода изображения, также кнопки загрузки изображения,загрузки видео или подключения веб-камеры для распознавания в реальном режиме. Так как, использовались библиотеки с классами, в процессе написания кода использовались принципы объектно-ориентированного программирования, что позволило сделать программу более гибкой и масштабируемой.

Процесс разработки программы включал в себя несколько этапов: проектирование архитектуры, написание кода, отладку, тестирование и оптимизацию.

Для использования библиотеки pixellib и ее зависимостей, необходимо установить их с помощью инструмента pip.

Чтобы установить pixellib, вы можете выполнить следующую команду в командной строке:

pip install pixellib

pip - это инструмент установки пакетов для языка программирования Python. Он предоставляет простой способ загрузки и установки дополнительных модулей и библиотек, которые не входят в стандартную библиотеку Python.

С помощью pip вы можете легко устанавливать пакеты из Python Package Index (PyPI) - официального репозитория пакетов Python. Вы также можете указать конкретные версии пакетов или установить их из локальных файлов или других источников.

После успешной установки всех необходимых библиотек вы сможете импортировать и использовать модули и классы из pixellib, imageai.Detection и pixellib.torchbackend.instance в своем коде.

import pixellib

from imageai.Detection import VideoObjectDetection

from pixellib.torchbackend.instance import instanceSegmentation

Функция import позволяет подключить нужные библиотеки к нашему проекту:

* Мы импортируем модуль pixellib, который предоставляет функциональность для работы с нейросетевыми моделями и выполнения задач компьютерного зрения.
* Затем мы импортируем класс VideoObjectDetection из модуля imageai.Detection. Этот класс позволяет нам выполнять обнаружение объектов на видео с использованием предобученных моделей глубокого обучения.
* Мы также импортируем класс instanceSegmentation из модуля pixellib.torchbackend.instance. Этот класс обеспечивает функциональность сегментации объектов, то есть разделение изображения на отдельные сегменты, соответствующие различным объектам.

import cv2

from tkinter import \*

from tkinter import ttk

from tkinter.filedialog import askopenfilename

from PIL import ImageTk, Image

import os

import subprocess

* import cv2: Мы импортируем модуль cv2 из библиотеки OpenCV. OpenCV - это библиотека компьютерного зрения, которая предоставляет множество функций для обработки изображений и видео.
* from tkinter import \*: Мы импортируем все классы и функции из модуля tkinter. Tkinter - это стандартная библиотека Python для создания графического интерфейса пользователя (GUI).
* from tkinter import ttk: Мы импортируем модуль ttk из tkinter. Модуль ttk предоставляет набор тематических виджетов для создания более современного и стильного интерфейса.
* from tkinter.filedialog import askopenfilename: Мы импортируем функцию askopenfilename из модуля filedialog. Эта функция позволяет открыть диалоговое окно для выбора файла с помощью стандартного диалога операционной системы.
* from PIL import ImageTk, Image: Мы импортируем классы ImageTk и Image из модуля PIL (Python Imaging Library). Эти классы предоставляют функциональность для работы с изображениями, включая загрузку, отображение и манипуляции с изображениями.

* import os: Мы импортируем модуль os, который предоставляет функции для взаимодействия с операционной системой, такие как работа с файловой системой, путями и окружением.
* import subprocess: Мы импортируем модуль subprocess, который предоставляет функции для запуска внешних процессов и выполнения команд командной строки.

Эти импорты позволяют нам использовать функциональность библиотек для создания графического интерфейса с помощью Tkinter, обработки изображений с помощью OpenCV и других операций, связанных с файлами и командной строкой. Например, мы можем открыть диалоговое окно для выбора файла изображения, отображать изображения на экране, выполнять операции над изображениями с помощью OpenCV и запускать внешние процессы через командную строку.

def main():

После идет главная функция в которой идет объявление графического интерфейса окна Windows:

myWindow = Tk()

myWindow.title("Нейросетевая система распознавания")

и кнопок с которыми будет взаимодействовать пользователь:

button1 = Button(frame,text="Выбрать изображение", command=open\_file,relief="solid",

background='red', foreground='white', font=('Arial', 14),

activebackground="grey",activeforeground="red",cursor="plus")

button1.pack(padx=5,pady=5)

.pack() это метод который позволяет размещать виджеты, такие как кнопки текстовые поля и другие, в окне Windows.

text="Выбрать изображение": Это свойство задает текст, отображаемый на кнопке. В данном случае, текст кнопки будет "Выбрать изображение".

command=open\_file: Это свойство определяет функцию, которая будет вызываться при нажатии на кнопку. В данном случае, при нажатии на кнопку будет вызвана функция open\_file.

relief="solid": Это свойство определяет стиль рамки кнопки. Значение "solid" означает, что рамка будет сплошной и иметь одинаковую толщину по всему периметру кнопки.

background='red': Это свойство определяет цвет фона кнопки. В данном случае, фон кнопки будет красного цвета.

foreground='white': Это свойство определяет цвет текста на кнопке. В данном случае, текст на кнопке будет белого цвета.

font=('Arial', 14): Это свойство определяет шрифт и размер шрифта текста на кнопке. В данном случае, используется шрифт Arial с размером 14 пунктов.

activebackground="grey": Это свойство определяет цвет фона кнопки в активном состоянии, когда кнопка нажата. В данном случае, фон кнопки будет серого цвета при нажатии.

activeforeground="red": Это свойство определяет цвет текста на кнопке в активном состоянии, когда кнопка нажата. В данном случае, текст на кнопке будет красного цвета при нажатии.

cursor="plus": Это свойство определяет внешний вид курсора при наведении на кнопку. В данном случае, курсор будет выглядеть как плюс.

y = Image.open("logo.png")

y.thumbnail((560,450))

y1 = ImageTk.PhotoImage(y)

lb1 = Label(foto, image=y1,relief="ridge",pady=15)

lb1.pack()

Создается виджет Label с именем lb1, который будет содержать отображаемое изображение(логотип университета). Свойство image устанавливается в объект y1, чтобы отобразить изображение на виджете. Дополнительные параметры задают внешний вид виджета, включая стиль рамки (relief="ridge") и отступы (pady=15).

menu = Menu(myWindow)

help\_menu = Menu(menu,tearoff=False)

menu.add\_command(label="Помощь (F1)",command=open\_help)

menu.add\_separator()

menu.add\_command(label="О программе (F2)", command=open\_version)

menu.add\_separator()

menu.add\_command(label="Об авторе (F3)", command=open\_about)

В этом фрагменте кода создается меню для графического интерфейса с помощью библиотеки Tkinter. Давайте рассмотрим каждую строку:

1. menu = Menu(myWindow): Создается объект Menu с именем menu. Он связан с главным окном myWindow (это предполагается, что myWindow - это объект осовного окна графического интерфейса).
2. help\_menu = Menu(menu, tearoff=False): Создается подменю с именем help\_menu с помощью Menu. Оно привязано к родительскому меню menu. Параметр tearoff=False указывает, что подменю не может быть вырвано из главного меню.
3. menu.add\_command(label="Помощь (F1)", command=open\_help): Добавляется команда (опция) в главное меню. В данном случае, команда имеет название "Помощь (F1)". Когда эта команда будет выбрана, будет вызвана функция open\_help.
4. menu.add\_separator(): Добавляется разделительная линия в главное меню. Она используется для визуального разделения опций в меню.
5. menu.add\_command(label="О программе (F2)", command=open\_version): Добавляется еще одна команда в главное меню. Она имеет название "О программе (F2)". Когда эта команда будет выбрана, будет вызвана функция open\_version.
6. menu.add\_separator(): Добавляется еще одна разделительная линия в главное меню.
7. menu.add\_command(label="Об авторе (F3)", command=open\_about): Добавляется третья команда в главное меню. Она имеет название "Об авторе (F3)". Когда эта команда будет выбрана, будет вызвана функция open\_about.

Этот код создает меню с тремя командами ("Помощь (F1)", "О программе (F2)" и "Об авторе (F3)") и связывает их с соответствующими функциями. Когда пользователь выбирает опцию из меню, будет вызываться соответствующая функция для выполнения определенных действий, например, открытия окна справки, информации о программе или информации об авторе.

myWindow.bind("<Escape>", lambda event: myWindow.quit())

myWindow.bind("<F1>",lambda event: open\_help())

myWindow.bind("<F2>", lambda event: open\_version())

myWindow.bind("<F3>", lambda event: open\_about())

В данном фрагменте кода определены привязки (bindings) клавиш к определенным функциям:

1. myWindow.bind("<Escape>", lambda event: myWindow.quit()): Эта строка привязывает клавишу Escape к функции myWindow.quit(). Когда пользователь нажимает клавишу Escape, будет выполнена функция myWindow.quit(), которая закрывает главное окно приложения.
2. myWindow.bind("<F1>", lambda event: open\_help()): Здесь клавиша F1 привязана к функции open\_help(). Когда пользователь нажимает клавишу F1, будет выполнена функция open\_help(), которая, вероятно, открывает окно со справкой или предоставляет помощь пользователю.
3. myWindow.bind("<F2>", lambda event: open\_version()): Клавиша F2 привязана к функции open\_version(). При нажатии клавиши F2 будет выполнена функция open\_version(), которая, возможно, открывает окно с информацией о версии программы.
4. myWindow.bind("<F3>", lambda event: open\_about()): Здесь клавиша F3 привязана к функции open\_about(). При нажатии клавиши F3 будет выполнена функция open\_about(), которая, скорее всего, открывает окно с информацией об авторе программы или приложения.

Таким образом, данные строки кода устанавливают обработчики событий для определенных клавиш, чтобы выполнять определенные действия при их нажатии в графическом интерфейсе.

def button\_click():

global exec\_path

exec\_path = os.getcwd()

if picture != "":

findimage(picture)

else:

answer.config(text="Пожалуйста выберите изображение!",bg="#FFD700",fg="black")

button2.config(state=DISABLED)

В целом, функция button\_click() выполняет обработку нажатия на кнопку. Если выбрано изображение (проверяется через переменную picture), вызывается соответствующая функция для обработки изображения. В противном случае, выводится сообщение и кнопка становится неактивной. Функция вызывается при нажатии кнопки ”Распознать объекты на изображении”

def findimage(picture):

ins = instanceSegmentation()

ins.load\_model("pointrend\_resnet50.pkl")

result = ins.segmentImage(picture, show\_bboxes=True, output\_image\_name="output\_image.jpg",

text\_thickness=2, box\_thickness=1, segment\_target\_classes=None,text\_size=0.8)

objects\_count = len(result[0]["scores"])

x = Image.open("output\_image.jpg")

x.thumbnail((560, 300))

x1 = ImageTk.PhotoImage(x)

lb1.config(image=x1 , justify="center")

lb1.image = x1

answer.config(text=f"Обработка произошла успешно!\n"

f"Найдено обьектов: {objects\_count}", bg="#00BB00")

os.startfile("output\_image.jpg")

Функция findimage(picture) выполняет следующие действия:

1. Создает экземпляр класса instanceSegmentation() и загружает модель pointrend\_resnet50.pkl, используемую для сегментации объектов на изображении.
2. Сегментирует изображение picture с помощью метода segmentImage(), который возвращает результаты сегментации, включая рамки объектов и оценки уверенности.
3. Выводит количество обнаруженных объектов из результатов сегментации.

def open\_file():

global picture

filename = askopenfilename(filetypes=[("All Files", "\*.\*")])

picture = os.path.basename(filename)

image = Image.open(filename)

image.thumbnail((550, 300))

image\_tk = ImageTk.PhotoImage(image)

lb1.config(image=image\_tk)

lb1.image = image\_tk

button2.config(state=NORMAL)

answer.config(text="Отлично, вы выбрали! Начнем?", bg="#FF0800",fg="white")

функция open\_file() выполняет операции, связанные с выбором и отображением изображения, а также обновляет состояние кнопки и текстового виджета на основе выбранного файла. Открывает диалоговое окно для выбора файла с помощью askopenfilename(). Пользователь может выбрать любой тип файла.

def open\_video():

global picture

answer.config(text="Отлично, выберите и ожидайте!", bg="#FF0800",fg="white")

filename = askopenfilename(filetypes=[("MP4 videos", "\*.mp4")])

picture = os.path.basename(filename)

if picture == "":

answer.config(text="Файл не выбран!",bg="#FFD700",fg="black")

else:

findvideo(picture)

answer.config(text="Обработка произошла успешно!", bg="#00BB00")

функция open\_video() выполняет те же операции,что и функция open\_file() Открывает диалоговое окно для выбора файла с помощью askopenfilename(). Пользователь может выбрать файл типа mp4 видео а затем вызывает функцию для обработки выбранного видео. Отображается соответствующий текстовый ответ, указывающий на результат обработки видео.

def findvideo(picture):

global exec\_path

exec\_path = os.getcwd()

vids = VideoObjectDetection()

vids.setModelTypeAsYOLOv3()

vids.setModelPath( os.path.join(exec\_path ,"yolov3.pt"))

vids.loadModel()

result= vids.detectObjectsFromVideo(input\_file\_path=os.path.join(exec\_path, picture),

output\_file\_path=os.path.join(exec\_path, "detected")

, frames\_per\_second=10,log\_progress=True,frame\_detection\_interval=2)

os.startfile("detected.mp4")

Функция findvideo(picture)

1. Объявляет переменную exec\_path как глобальную, чтобы она могла быть использована и изменена внутри функции.
2. Устанавливает значение переменной exec\_path равным текущему рабочему каталогу с помощью os.getcwd().
3. Создает объект vids класса VideoObjectDetection.
4. Устанавливает тип модели как YOLOv3 с помощью setModelTypeAsYOLOv3().
5. Устанавливает путь к модели YOLOv3 с помощью setModelPath(). Путь к модели объединяется с exec\_path для получения полного пути к модели.
6. Загружает модель YOLOv3 с помощью loadModel().
7. а затем выполняет обработку видео с использованием модели YOLOv3 и сохраняет результат в файле "detected.mp4".

def open\_help():

subprocess.run(['python', "helpWindow.py"])

def open\_version():

subprocess.run(['python', "version.py"])

def open\_about():

subprocess.run(['python', "about.py"])

Каждая из этих функций используется для открытия вспомогательных окон или выполнения определенных действий, вызывая соответствующие скрипты Python с помощью subprocess.run().

Они вызываются с помощью привязки событий к клавишам F1, F2, F3

# Описание программы

**“**Нейросетевая система распознавания объектов” представляет собой программу, которая использует нейросетевые модели для распознавания объектов на изображениях и видео. Программа имеет графический интерфейс, реализованный с помощью библиотеки Tkinter. При запуске программы открывается главное окно с кнопками для работы с нейросетью. вверху окна есть меню для открытия дополнительных окон.

Основные функции программы:

* Открытие изображения: Пользователь может выбрать изображение с помощью кнопки "Выбрать изображение". Выбранное изображение отображается в окне программы.
* Распознавание объектов на изображении: После выбора изображения пользователь может нажать кнопку "Распознать объекты на изображении", чтобы запустить процесс распознавания объектов на выбранном изображении. Результат распознавания отображается в окне программы, где каждый обнаруженный объект помечается рамкой.
* Распознавание объектов на видео: Пользователь может выбрать видеофайл с помощью кнопки "Распознать объекты на видео". После выбора видеофайла происходит процесс распознавания объектов на видео. Результат сохраняется в видеофайле "detected.mp4".
* Реальное время: При нажатии кнопки "В реальном времени" программа открывает камеру и начинает распознавать объекты в режиме реального времени. Результат записывается в видеофайл "online.mp4".
* Помощь, информация о программе и об авторе: В меню программы доступны опции для открытия вспомогательных окон, которые предоставляют помощь, информацию о программе и информацию об авторе.
* Программа использует различные библиотеки, такие как pixellib, imageai, cv2, tkinter, PIL, os и subprocess, чтобы реализовать функциональность распознавания объектов и взаимодействия с пользователем через графический интерфейс.

# Описание алгоритмов и методов решения. Функциональное описание.

Программа использует несколько алгоритмов и методов для решения задачи распознавания объектов на изображениях и видео. Вот их функциональное описание:

1. Алгоритм распознавания объектов на изображении:

При выборе изображения пользователем, вызывается функция open\_file(), которая открывает диалоговое окно для выбора файла и загружает выбранное изображение.

Затем пользователь может нажать кнопку "Распознать объекты на изображении". Это вызывает функцию button\_click(), которая проверяет, выбрано ли изображение, и вызывает функцию findimage(picture) для распознавания объектов на изображении.

Функция findimage(picture) использует модель инстансного сегментирования instanceSegmentation() из библиотеки pixellib для распознавания объектов. Изображение передается в функцию, и происходит процесс сегментации и распознавания объектов. Результаты распознавания сохраняются в файле "output\_image.jpg". Количество обнаруженных объектов извлекается из результатов и отображается пользователю.

1. Алгоритм распознавания объектов на видео:

При выборе видеофайла пользователем, вызывается функция open\_video(), которая открывает диалоговое окно для выбора файла и загружает выбранное видео.

Если файл выбран, функция open\_video() вызывает функцию findvideo(picture), которая использует модель объектного обнаружения на видео VideoObjectDetection() из библиотеки imageai. Происходит процесс обнаружения объектов на видео, и результаты сохраняются в видеофайле "detected.mp4".

Затем выводится сообщение об успешной обработке видео.

1. Алгоритм распознавания объектов в реальном времени:

При нажатии кнопки "В реальном времени" вызывается функция online(), которая открывает камеру и начинает процесс обнаружения объектов в реальном времени.

Функция использует модель инстансного сегментирования instanceSegmentation() для обработки кадров с камеры. Результаты сегментации и обнаружения объектов записываются в видеофайл "online.mp4".

После завершения обработки, видеофайл "online.mp4" открывается для просмотра результатов.

1. Дополнительные функции:

Программа также предоставляет меню с опциями "Помощь", "О программе" и "Об авторе", которые открывают соответствующие окна с дополнительной информацией.

Также предусмотрено использование клавиш F1, F2 и F3 для вызова окон помощи, информации о программе и информации об авторе соответственно.

Для открытия вспомогательных окон используется модуль subprocess для запуска соответствующих скриптов внешних файлов.

# 8. Описание пользовательского интерфейса

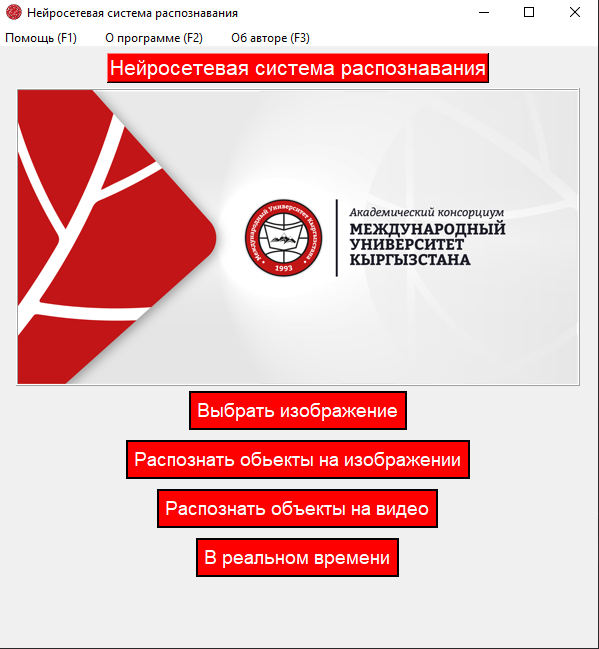


рис 1.1 - главный экран

При запуске программы, вы окажетесь в главном экране, где можете управлять нейросетью.

В вверху меню с нужными разделами, нажав на них вы можете перейти в то или иное окно. или нажав соответствующую кнопку(F1,F2,F3).

Меню «О программе» находится в главном меню программы и позволяет пользователям получить дополнительную информацию о приложении и разработчиках. При нажатии на эту кнопку, вы попадете в раздел, где будет отображена информация о программе.

Меню “Помощь” также находится в главном меню и позволяет получит подсказки к программе.

Меню “Об авторе” это раздел где вы можете связатся с автором и найти его контакты

Дальше название программы, а затем окно с изображением где уже установлен логотип университета по умолчанию. А после нажатия кнопки “Выбрать изображение”Б открывается загрузчик системы и в это окно грузиться выбранное изображение.

Кнопка “Распознать объекты на изображении” начинает распознавание только на изображениях, если это видео появится окно с предупреждением.

Кнопка “Распознать объекты на видео” обрабатывает только видео. Поэтому при нажатии на кнопку откроется загрузчик где дозволено грузить только видео. загруженное видео не будет показано до тех пор пока не закончится распознание.

Кнопка “В реальном времени” подключается к веб=камере и открывается окно с трансляцией. где объекты сегментированы. После закрытия окна видео сохраняется.

# 9. Руководство пользователя

При запуске программы "Нейросетевая система распознавания" вы окажетесь на главном экране, где можно управлять нейросетью и выполнить различные операции распознавания объектов. Вот руководство для пользователя:

1. Выбор изображения:
2. Нажмите кнопку "Выбрать изображение".
3. В открывшемся диалоговом окне выберите изображение, которое вы хотите обработать.
4. Выбранное изображение будет отображаться в окне программы.
5. Распознавание объектов на изображении:
6. После выбора изображения, нажмите кнопку "Распознать объекты на изображении".
7. Программа выполнит распознавание объектов на выбранном изображении и отобразит результаты в окне программы.
8. Результаты будут включать обнаруженные объекты и количество найденных объектов.
9. Распознавание объектов на видео:
10. Нажмите кнопку "Распознать объекты на видео".
11. В открывшемся диалоговом окне выберите видеофайл, который вы хотите обработать.
12. Программа выполнит распознавание объектов на выбранном видео и сохранит результаты в видеофайле "detected.mp4".
13. По окончании обработки видеофайла, результаты будут отображены в окне программы, и видеофайл с результатами будет автоматически открыт.

4. Реальное время:

1. Нажмите кнопку "В реальном времени".
2. Программа подключится к веб-камере и откроет окно с трансляцией изображения с веб-камеры.
3. В реальном времени будет выполняться сегментация объектов на изображении.
4. По окончании трансляции, программа сохранит видеофайл с результатами сегментации под названием "online.mp4".

5. Меню "О программе":

В главном меню программы находится раздел "О программе".

1. Чтобы получить дополнительную информацию о программе, нажмите на соответствующий пункт в меню.
2. В открывшемся окне будет отображена информация о программе, включая версию и автора.

6. Меню "Помощь":

1. В главном меню программы находится раздел "Помощь".
2. Чтобы получить подсказки и дополнительную информацию о программе, нажмите на соответствующий пункт в меню.
3. В открывшемся окне будет предоставлена справочная информация и инструкции по использованию программы.

7. Меню "Об авторе":

1. В главном меню программы находится раздел "Об авторе".
2. Чтобы связаться с автором программы или найти его контактные данные, нажмите на соответствующий пункт в меню.
3. В открывшемся окне будет предоставлена информация об авторе и его контактные данные.

Следуя указаниям и использованию доступных функций в пользовательском интерфейсе программы, вы сможете управлять нейросетевой системой распознавания и получить результаты обработки изображений и видео.

# Заключение

В данной курсовой работе была разработана нейросетевая система распознавания, которая позволяет обрабатывать изображения и видео с целью обнаружения объектов на них. Были применены различные алгоритмы и методы машинного обучения, включая глубокие нейронные сети, для достижения высокой точности распознавания.

В рамках работы были выполнены следующие задачи:

* Изучение существующих методов и алгоритмов для распознавания объектов на изображениях и видео.
* Разработка архитектуры нейросетевой системы, включая выбор подходящих моделей и алгоритмов.
* Обучение модели на обучающих данных с использованием методов глубокого обучения.
* Реализация программного интерфейса пользователя для управления системой и визуализации результатов распознавания.
* Тестирование системы на различных наборах данных и оценка ее производительности и точности.

В результате проведенных исследований и разработок была создана нейросетевая система распознавания, способная эффективно обрабатывать изображения и видео, и обнаруживать на них различные объекты. Система показала высокую точность распознавания и удобство использования благодаря разработанному пользовательскому интерфейсу.

Однако, в ходе работы выявились некоторые ограничения и возможности для улучшения системы. Например, можно провести дальнейшие исследования для оптимизации алгоритмов и улучшения производительности системы. Также, можно расширить функциональность системы, добавив поддержку распознавания других типов объектов и возможность обучения на пользовательских данных.

В целом, данная курсовая работа вносит вклад в обучение в области компьютерного зрения и нейросетевых систем распознавания. Она предоставляет основу для дальнейших исследований и разработок в этой области и может быть использована в различных практических приложениях, таких как автоматическое распознавание объектов на видеонаблюдении, автономные системы, медицинская диагностика и другие.

# Список используемой литературы

1. <https://pixellib.readthedocs.io/en/latest/>

2. <https://imageai.readthedocs.io/en/latest/>

3. https://www.youtube.com/watch?v=mop6g-c5HEY&t=44924s