**Государственное бюджетное общеобразовательное учреждение**

**города Москвы «Школа №1347»**

**Разработка системы автоматического анализа фото, видео с распознаванием объектов, текста и речи**

Выполнил:

Учащийся 10 «Б» класса ГБОУ Школа № 1347

Смирнов Михаил Павлович

Руководитель работы:

Учитель информатики ГБОУ Школа № 1347

Кузнецов Илья Николаевич

**Москва, 2024**

Оглавление

[Введение 3](#_Toc65352580)

[Цель, постановка задачи 5](#_Toc65352581)

[Методика выполнения 6](#_Toc65352582)

[Практическая и теоретическая значимость 15](#_Toc65352583)

[Результаты и выводы 16](#_Toc65352584)

[Перспективы развития 17](#_Toc65352585)

[Список литературы 18](#_Toc65352586)

# Введение

Современные технологии анализа видео, фото или аудио записи позволяют автоматизировать многие процессы, включая контроль безопасности, анализ поведения, автоматическое создание субтитров и машинный перевод речи. Важно создавать доступные и удобные решения, способные обрабатывать видео и аудио потоки и извлекать полезную информацию.

# Обоснование выбора темы

Распознавание объектов, текста и речи является ключевой задачей компьютерного зрения и искусственного интеллекта. Создание приложения, объединяющего эти функции, поможет автоматизировать обработку видео в различных сферах: от мониторинга городской среды до анализа образовательного контента.

Существующие системы анализа видео зачастую узкоспециализированы: одни решения ориентированы только на детекцию объектов, другие – на распознавание речи, третьи – на обработку текста. Комплексное решение, объединяющее эти технологии, будет полезным инструментом для анализа видеоданных.

# Цель, постановка задачи

**Цель**

Разработать программный инструмент для анализа видео, включающий распознавание объектов, текста и речи.

**Задачи**

1. Реализовать систему детекции объектов с использованием нейросетевой модели YOLOv8.
2. Внедрить алгоритмы распознавания текста на изображениях и кадрах видео (OCR).
3. Добавить функционал обработки аудиодорожки видео и конвертации речи в текст.
4. Разработать графический интерфейс пользователя (GUI) на PyQt5.
5. Реализовать возможность сохранения полученных результатов.
6. Провести тестирование системы на различных видеоданных.

# Методика выполнения работы

## **Программное обеспечение и инструменты**

1. **Язык программирования:** Python 3.12
2. **Библиотеки:**
   * OpenCV – обработка изображений и видео
   * Ultralytics YOLO – детекция объектов
   * EasyOCR – распознавание текста
   * Deep Translator – машинный перевод
   * SpeechRecognition – распознавание речи
   * Pydub – обработка аудиофайлов
   * Avconv – обработка видео/аудио
3. **Фреймворк для GUI:** PyQt5
4. **Среда разработки:**PyCharm
5. **Операционные системы:** Windows

# Аппаратное обеспечение

1. Компьютер с GPU (желательно для ускорения работы YOLO).
2. Видеокамера (для тестирования в реальном времени).
3. Микрофон (для тестирования распознавания речи).

# Методы работы

* Предварительная обработка кадров и применение YOLO для детекции объектов.
* Оптическое распознавание текста (OCR) на отдельных кадрах.
* Конвертация аудиодорожки в текст с помощью SpeechRecognition.
* Внедрение механизмов перевода текста.
* Тестирование системы на различных видеофайлах.

# Место и сроки выполнения работы

Разработка проводилась в ИТ полигоне ГБОУ Школа 1347 в течении 3 месяцев.

# Результаты работы

**Функциональные возможности программы**

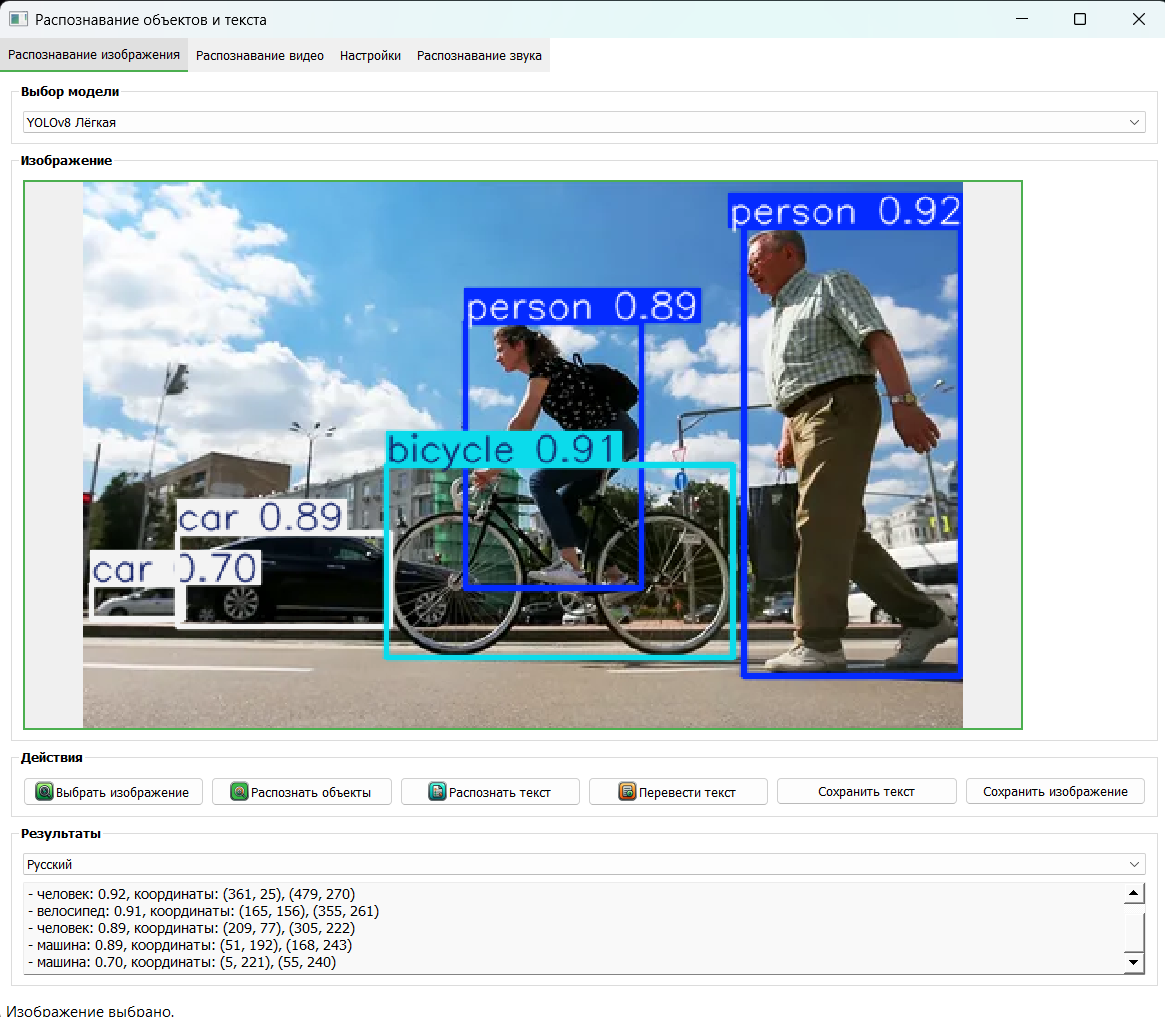
* Графический интерфейс позволяет загружать и анализировать видеофайлы.
* Детекция объектов в кадрах видео с возможностью выбора модели YOLOv8.
* Автоматическое распознавание текста и перевод.
* Анализ аудиофайлов с извлечением текста речи.
* Поддержка различных форматов видео и аудио.
* Возможность сохранения обработанных данных.

**Примеры работы**

Распознавание объектов (рис.1):

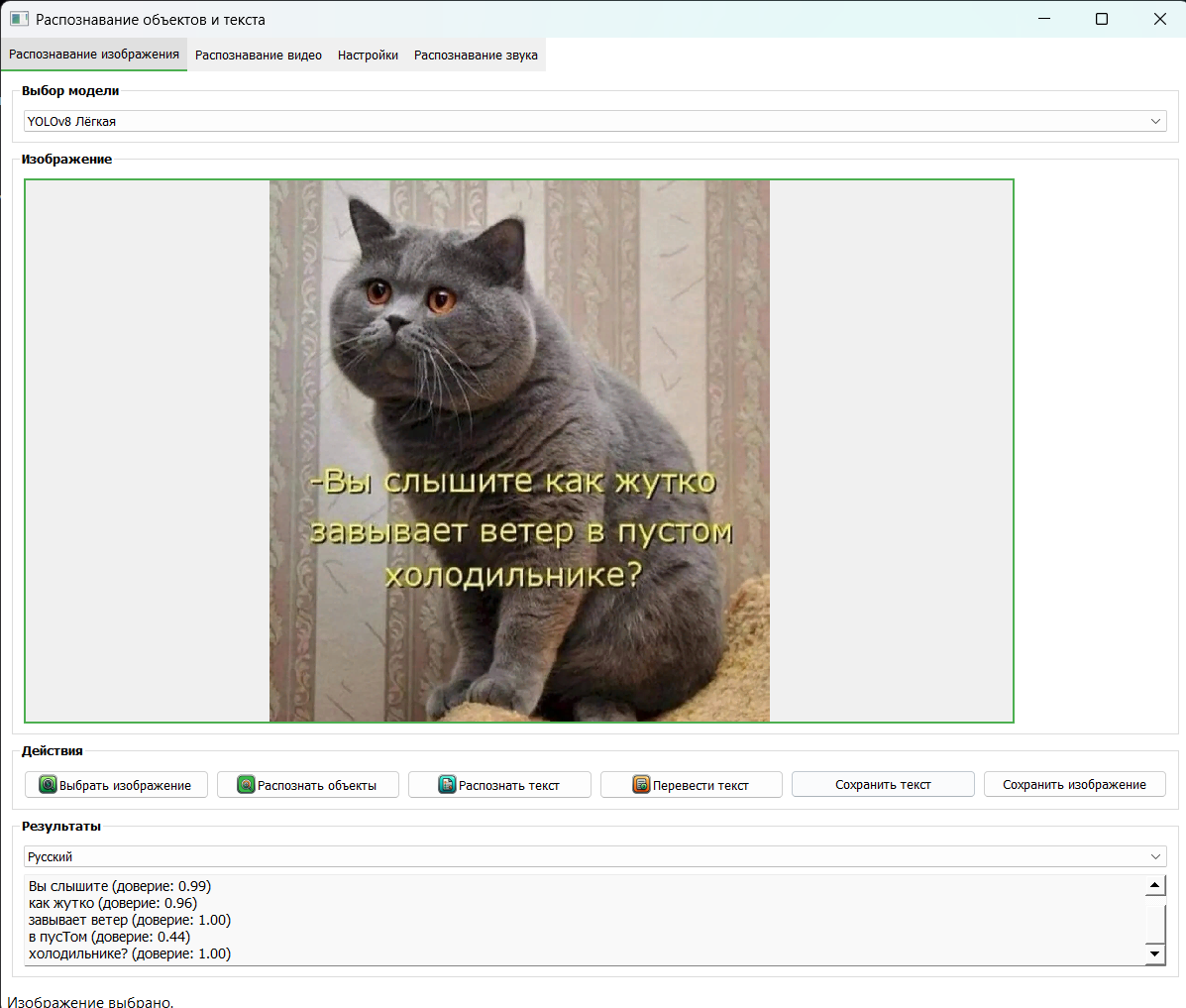
def detect\_objects(self):  
 if not self.image\_path:  
 self.result\_text.setText("Сначала выберите изображение.")  
 return  
  
 if not self.model:  
 self.result\_text.setText("Модель не загружена. Выберите модель из списка.")  
 return  
  
 image = cv2.imread(self.image\_path)  
 results = self.model.predict(image**,** conf=self.conf\_threshold**,** imgsz=self.input\_size)  
 annotated\_image = results[**0**].plot()  
  
 # Сохраняем аннотированное изображение для последующего сохранения  
 self.annotated\_image = cv2.cvtColor(annotated\_image**,** cv2.COLOR\_RGB2BGR) # Преобразование в формат BGR для OpenCV  
  
 # Отображаем изображение  
 height**,** width**,** channel = annotated\_image.shape  
 bytes\_per\_line = **3** \* width  
 qt\_image = QImage(annotated\_image.data**,** width**,** height**,** bytes\_per\_line**,** QImage.Format\_RGB888).rgbSwapped()  
  
 pixmap = QPixmap.fromImage(qt\_image)  
 self.image\_label.setPixmap(  
 pixmap.scaled(self.image\_label.width()**,** self.image\_label.height()**,** Qt.KeepAspectRatio))  
  
 detected\_objects = results[**0**].boxes.data.cpu().numpy()  
 self.result\_text.clear()  
 if detected\_objects.size > **0**:  
 self.result\_text.append("Распознанные объекты:")  
 for box in detected\_objects:  
 x1**,** y1**,** x2**,** y2**,** confidence**,** class\_id = box  
 class\_name = self.model.names[int(class\_id)]  
 self.result\_text.append(  
 f"- {translate\_text(class\_name**,** 'ru')}: {confidence:.2f}, координаты: ({x1:.0f}, {y1:.0f}), ({x2:.0f}, {y2:.0f})")  
 else:  
 self.result\_text.append("Объекты не обнаружены.")

В данном фрагменте кода показано распознавание объектов на фото, для удобства в функции detect\_objects добавлена возможность изменять исходное фото для удобства восприятия информации.

**

Распознавание текста (рис.2):

def recognize\_text\_with\_easyocr(self):  
 if not self.image\_path:  
 self.result\_text.setText("Сначала выберите изображение.")  
 return  
  
 try:  
 reader = easyocr.Reader(['ru'**,** 'en']**,** gpu=False)  
 results = reader.readtext(self.image\_path**,** detail=**1**)  
  
 self.result\_text.clear()  
 if results:  
 self.result\_text.append("Распознанный текст:")  
 for bbox**,** text**,** confidence in results:  
 self.result\_text.append(f"{text} (доверие: {confidence:.2f})")  
 else:  
 self.result\_text.append("Текст не обнаружен.")  
 except Exception as e:  
 self.result\_text.setText(f"Ошибка распознавания текста: {e}")

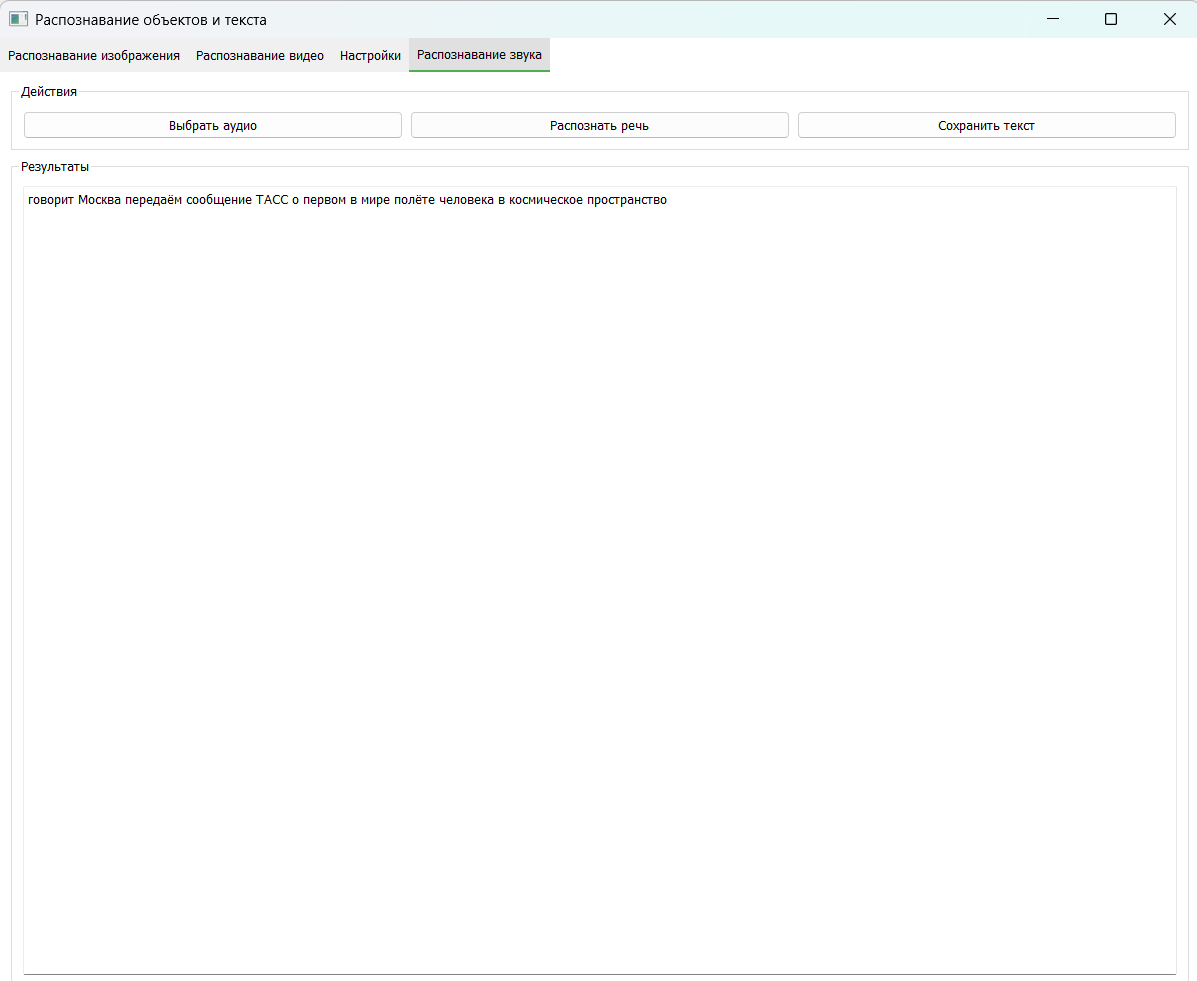
с помощью библиотеки easyocr можно распознавать текст на фото  
**

Распознавание речи (рис.3):

<https://disk.yandex.ru/d/T3CPckxaxyywvg>

здесь для распознавания текста из аудио файлов используется библиотека speech\_recognition, для распознавания текста требуется создание отдельного файла с расширением \*.wav

def recognize\_audio(self):  
 if not hasattr(self**,** 'audio\_path') or not self.audio\_path:  
 self.status\_bar.showMessage("Сначала выберите аудиофайл."**, 5000**)  
 return  
  
 recognizer = sr.Recognizer()  
 sound = AudioSegment.from\_mp3(self.audio\_path)  
 sound.export("result.wav"**,** format="wav")  
 try:  
 with sr.AudioFile("result.wav") as source:  
 audio\_data = recognizer.record(source)  
 recognized\_text = recognizer.recognize\_google(audio\_data**,** language='ru-RU')  
 self.audio\_result\_text.setPlainText(recognized\_text)  
 self.status\_bar.showMessage("Распознавание речи завершено."**, 5000**)  
 except Exception as e:  
 self.audio\_result\_text.setPlainText(f"Ошибка распознавания: {e}")  
 self.status\_bar.showMessage(f"Ошибка распознавания: {e}"**, 5000**)

**

**Выводы и перспективы дальнейшей работы**

**Выводы**

1. Разработано программное обеспечение, которое объединяет анализ видео, текста и речи.
2. Система успешно выполняет распознавание объектов, текстов и аудио.
3. Интеграция с PyQt5 обеспечивает удобный пользовательский интерфейс.
4. Использование YOLOv8 позволяет достичь высокой точности детекции объектов.

**Перспективы дальнейшей работы**

* Добавление поддержки онлайн-анализа видеопотоков (например, с камер наблюдения).
* Оптимизация модели YOLO для работы на мобильных устройствах.
* Улучшение алгоритмов распознавания речи и добавление поддержки нескольких языков.

**Список используемой литературы**

1. Redmon J., Farhadi A. "YOLOv3: An Incremental Improvement", 2018.
2. Ultralytics. "YOLOv8 Documentation". https://docs.ultralytics.com
3. Smith R. "An Overview of the Tesseract OCR Engine", 2007.
4. Python SpeechRecognition Library. <https://pypi.org/project/SpeechRecognition/>
5. PyQt5 Documentation. https://www.riverbankcomputing.com/software/pyqt/intro