МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра математической кибернетики и компьютерных наук

СЕРВИС АВТОМАТИЗИРОВАННОЙ ОЦЕНКИ СТЕПЕНИ ЗАГРУЗКИ КУЗОВА ГРУЗОВЫХ АВТОМОБИЛЕЙ ПО ФОТОГРАФИИ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Белова Александра Евгеньевича

Научный руководитель
доцент _____ Д. С. Пантелеев

Заведующий кафедрой
доцент, к. ф.-м. н. _____ С. В. Миронов

Саратов 2026

ВВЕДЕНИЕ

Автоматизация визуального контроля является актуальной задачей для производственных и строительных процессов, где значительная часть решений всё ещё принимается оператором по внешним признакам. Такой подход прост, но субъективен: результат зависит от опыта работника, условий наблюдения, освещения, ракурса съёмки и степени усталости. В результате одинаковые ситуации могут оцениваться по-разному, что снижает воспроизводимость контроля и может приводить к ошибкам в технологическом процессе.

Одним из прикладных примеров является оценка степени загрузки кузова грузового автомобиля или самосвала. Во время погрузки оператору необходимо быстро определить, достаточно ли загружен кузов, требуется ли продолжить операцию или можно завершать процесс. Недогруз снижает эффективность рейса и производительность работ, а перегруз может приводить к повышенному износу техники, росту эксплуатационных затрат и дополнительным рискам безопасности. Поэтому система, способная автоматически анализировать изображение кузова и определять степень его заполнения, может использоваться как вспомогательный инструмент при принятии решения.

Развитие методов компьютерного зрения и машинного обучения позволяет рассматривать программные решения как дополнение к традиционным средствам контроля. Весовые системы дают точный результат, но требуют отдельной инфраструктуры. Датчиковые решения позволяют получать данные автоматически, однако связаны с установкой оборудования, настройкой и последующим обслуживанием. Подходы на основе анализа изображений имеют более низкий порог внедрения, поскольку в качестве источника данных может использоваться обычная фотография или кадр с камеры.

Цель выпускной квалификационной работы – разработать программный прототип системы автоматизированной оценки степени загрузки кузова грузового автомобиля по фотографии и подготовить воспроизводимую инженерную основу для экспериментальной проверки такого подхода.

Для достижения поставленной цели в работе были рассмотрены существующие подходы к контролю загрузки кузова, определена постановка задачи, сформирован набор изображений, выполнена подготовка данных к обучению, обучена модель компьютерного зрения и реализован демонстрационный программный прототип с backend- и frontend-частями.

В качестве материалов исследования использовался набор изображений кузовов грузовых автомобилей, сформированный специально для данной работы. Поскольку готового специализированного датасета для такой задачи в открытом доступе не удалось найти, данные пришлось собирать из разных источников и дополнительно отбирать вручную. Для каждого изображения задавался один из трёх классов загрузки: пустой кузов, частично загруженный кузов и полностью загруженный кузов.

Научная новизна работы состоит не в создании принципиально нового математического метода компьютерного зрения, а в формировании целостной и воспроизводимой постановки прикладной задачи для предметной области, слабо обеспеченной открытыми размеченными данными. Практическая значимость заключается в реализации программного прототипа, который объединяет обученную модель, серверную часть, пользовательский интерфейс и механизм визуального выделения области кузова.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Первая глава посвящена анализу предметной области и постановке задачи автоматизированного определения степени загрузки кузова. В ней рассматривается проблема визуального контроля при погрузке и перевозке материалов. Показано, что операторская оценка является простой, но субъективной: она зависит от ракурса наблюдения, освещённости, опыта работника, состояния груза и видимости кузова. Особенно неоднозначными являются ситуации, когда груз распределён неравномерно или часть кузова перекрыта элементами техники.

В работе рассмотрены основные подходы к решению задачи. Весовые системы позволяют определить массу автомобиля до и после погрузки, однако требуют весовой площадки или иной специализированной инфраструктуры. Датчиковые системы фиксируют отдельные параметры работы техники, но требуют установки дополнительного оборудования и обычно привязаны к конкретной машине. Подходы на основе компьютерного зрения используют изображение как основной источник информации и позволяют построить более гибкое программное решение. Для дипломного прототипа этот вариант является наиболее реалистичным, поскольку не требует изменения конструкции транспортного средства и может работать с обычной фотографией.

Решаемая задача формализована как многоклассовая классификация изображений. На вход системе поступает одиночное изображение кузова грузового автомобиля или самосвала. После обработки оно должно быть отнесено к одному из трёх классов: `empty`, `partial` или `full`. Такой выбор связан с ограниченным объёмом данных и высокой субъективностью процентной разметки. Если бы задача решалась как регрессия, для каждого изображения потребовалось бы указывать точный процент заполнения, который в реальных условиях трудно определить однозначно. Классовая шкала является более устойчивой и лучше соответствует демонстрационному характеру прототипа.

Отдельной проблемой стало отсутствие специализированного датасета. В открытых источниках можно найти изображения самосвалов, грузовых автомобилей и строительной техники, однако такие данные обычно не имеют разметки по степени заполнения кузова. Кроме того, многие изображения оказываются непригодными для обучения: кузов может быть снят сбоку, частично закрыт ковшом, находиться слишком далеко или занимать малую часть кадра. Поэтому формирование набора данных стало самостоятельным этапом работы. При от-

боре сохранялись только те изображения, на которых можно было достаточно уверенно определить состояние кузова.

При подготовке датасета использовались как реальные изображения, так и дополнительные материалы, полученные из видеозаписей и сгенерированных сцен. Такой подход позволил расширить выборку и добавить больше примеров для разных состояний загрузки. При этом синтетические изображения рассматривались не как полноценная замена реальным данным, а как практический способ дополнить обучающую выборку в условиях ограниченного количества подходящих фотографий.

Вторая глава посвящена проектированию и реализации программного прототипа. Серверная часть реализована на языке Python с использованием FastAPI. Backend принимает изображение, выполняет базовую проверку входных данных, передаёт файл в модуль машинного обучения и возвращает результат классификации во frontend. Информация об активной модели хранится отдельно в файле `active-model.json`, где указываются архитектура, путь к checkpoint-файлу, список классов, размер входного изображения, параметры нормализации и основные метрики качества. Такой подход упрощает замену модели без существенного изменения кода серверной части.

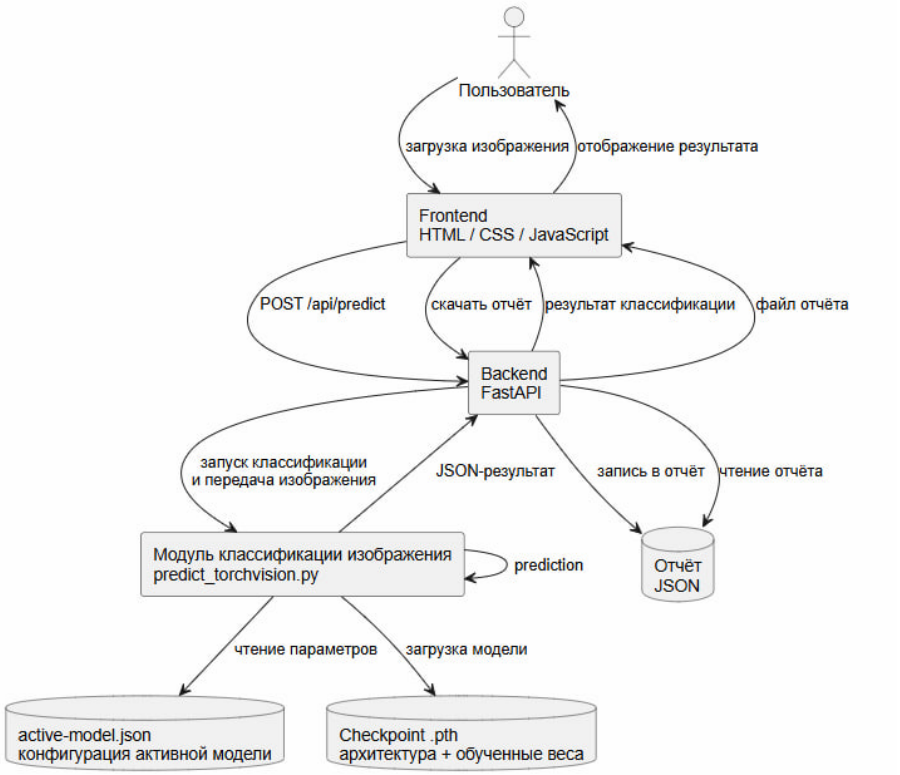

Рисунок 1 – Архитектурная схема программного прототипа

Архитектурно система разделена на несколько логических частей. Пользователь взаимодействует с web-интерфейсом, через который загружает изображение. Backend принимает файл и передаёт его в модуль классификации, где выполняется подготовка изображения и запуск модели. После этого результат возвращается пользователю в понятном виде. Такое разделение позволяет не смешивать интерфейсную часть, серверную обработку и код машинного обучения.

После ручного отбора данные фиксировались в manifest-файле. В нём для каждого изображения сохранялись путь, метка класса, принадлежность к выборке и статус принятия. Разбиение выполнялось отдельно для каждого класса: основная часть изображений использовалась для обучения, меньшая часть – для валидации, а оставшиеся данные – для итоговой проверки качества. Для воспроизводимости использовалось фиксированное значение `random.seed(42)`.

Перед подачей в модель изображения проходили предварительную обработку. Сначала файл открывался с помощью библиотеки Pillow, затем применялась корректировка EXIF-ориентации и преобразование в формат RGB. После этого изображение приводилось к размеру 224 на 224 пикселя, переводилось в тензор и нормализовалось по средним значениям и стандартным отклонениям ImageNet. Для обучающей выборки дополнительно применялись аугментации, которые имитировали изменение ракурса, освещения, перспективы и частичные перекрытия. Это позволило повысить устойчивость модели без физического увеличения числа файлов в датасете.

Для классификации были рассмотрены модели ResNet18 и ResNet34. Обе архитектуры относятся к сверточным нейронным сетям с остаточными связями, однако отличаются глубиной. ResNet18 является более лёгкой и быстрее обучается, но может хуже выделять сложные признаки. ResNet34 глубже и способна учитывать более сложные визуальные особенности: форму кузова, распределение материала, границу между грузом и бортами, а также различия между частичной и полной загрузкой.

Обучение выполнялось с использованием подхода transfer learning. В качестве основы была взята ResNet34 с предобученными весами ImageNet. Последний полносвязный слой, рассчитанный на 1000 классов, был заменён на новый слой с тремя выходами. В итоговой конфигурации большая часть параметров модели была заморожена, а обучение выполнялось для блока `layer4` и нового

слоя `fc`. Такой вариант fine-tuning позволяет сохранить универсальные признаки, полученные при предварительном обучении, и адаптировать верхние слои к задаче классификации загрузки кузова.

В качестве функции потерь использовалась `CrossEntropyLoss`, подходящая для многоклассовой классификации. Для учёта возможного дисбаланса классов применялись веса, рассчитанные по обучающей выборке. Оптимизация выполнялась алгоритмом Adam только по тем параметрам, для которых был включён признак `requires_grad=True`. После каждой эпохи модель проверялась на валидационной выборке, и при улучшении accuracy текущее состояние весов сохранялось как лучшее.

Результаты сравнения моделей показали преимущество ResNet34. Значение accuracy на тестовой выборке составило 0.8958, macro F1 – 0.8967, precision – 0.9127. ResNet18 также показала приемлемый результат, однако уступила ResNet34 по основным метрикам.

Таблица 1 – Сравнение моделей ResNet18 и ResNet34

Модель	**Val accuracy**	**Test accuracy**	**Macro F1**	**Precision**
ResNet18	0.7955	0.8125	0.8094	0.8217
ResNet34	0.8636	0.8958	0.8967	0.9127

После завершения обучения модель была сохранена в checkpoint-файл `system/checkpoints/best_3class_resnet34.pth`. В него записывались архитектура модели, количество классов, `state_dict`, соответствие классов и размер входного изображения. При инференсе создаётся такая же архитектура ResNet34, после чего в неё загружаются обученные веса. Новое изображение проходит ту же предобработку, затем передаётся в модель. С помощью `softmax` выходные значения преобразуются в вероятности, а `argmax` выбирает класс с максимальным значением.

Дополнительно в прототип была включена связка GroundingDINO и SAM для визуального выделения области кузова. GroundingDINO используется для поиска объекта по текстовому запросу, а SAM строит маску найденной области. Данный механизм не заменяет классификационную модель ResNet34, а используется как вспомогательная часть системы, позволяющая наглядно показать пользователю область изображения, связанную с анализируемым объектом.

Frontend-интерфейс реализован с использованием HTML, CSS и JavaScript.

Пользователь может выбрать изображение на странице, отправить его на сервер и увидеть результат классификации. Помимо итогового класса, интерфейс может отображать уверенность модели и вероятности по классам. Это полезно в спорных случаях, когда значения для классов `partial` и `full` близки и изображение требует дополнительной проверки.

Ключевое ограничение разработанного решения связано с объёмом и составом датасета. Качество модели зависит от того, насколько подготовленные изображения соответствуют будущим условиям эксплуатации. Кроме того, система определяет только один из трёх классов, но не вычисляет точный объём или массу груза.

ЗАКЛЮЧЕНИЕ

В ходе выпускной квалификационной работы была рассмотрена задача автоматизированного определения степени загрузки кузова грузового автомобиля по фотографии. В качестве основного варианта решения была выбрана трёхклассовая классификация изображений, при которой система относит кузов к одному из состояний: пустой, частично загруженный или полностью загруженный. Такой подход оказался удобным для демонстрационного прототипа, поскольку пользователь получает понятный результат, не требующий спорной процентной оценки заполнения.

Для решения задачи был сформирован специализированный набор изображений кузовов грузовых автомобилей. Данные были вручную отобраны, размечены по трём классам и разделены на обучающую, валидационную и тестовую части. Перед обучением изображения приводились к единому формату, а для повышения устойчивости модели использовались аугментации, имитирующие реальные изменения условий съёмки.

Для классификации были обучены и сравнены модели ResNet18 и ResNet34. По результатам экспериментов лучшей стала ResNet34: на тестовой выборке accuracy составила 0.8958, macro F1 – 0.8967, precision – 0.9127. На основании сравнения ResNet34 была выбрана в качестве основной модели и сохранена в checkpoint-файл для дальнейшего использования в приложении.

Также был реализован программный прототип, включающий backend на FastAPI, модуль классификации изображения и frontend-интерфейс на HTML, CSS и JavaScript. Пользователь может загрузить изображение через web-страницу, после чего backend передаёт его в модель и возвращает результат классификации. Дополнительно в проект включено визуальное выделение области кузова с использованием GroundingDINO и SAM.

Полученный результат можно считать законченным демонстрационным прототипом, который показывает полный путь обработки изображения: загрузка файла, подготовка данных, применение модели, получение класса загрузки и отображение результата пользователю. Для дальнейшего развития системы необходимо расширить реальный датасет, проверить модель на большем числе производственных сценариев, добавить обработку нерелевантных изображений и реализовать более строгую оценку надёжности предсказаний.