МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра дискретной математики и информационных технологий

РАЗРАБОТКА ИИ БОТА «АССИСТЕНТ ПОКУПАТЕЛЯ»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Хорькова Даниила Игоревича

Научный руководитель
профессор, д. э. н. _____ Л. В. Кальянов

Заведующий кафедрой
к. ф.-м. н., доцент _____ Л. Б. Тяпаев

Саратов 2026

ВВЕДЕНИЕ

Современный этап развития электронной коммерции характеризуется высокой динамикой цен, широким ассортиментом и необходимостью оперативного получения информации о товарах. Покупатели всё чаще обращаются к онлайн-платформам, однако процесс поиска актуальной цены, особенно при большом количестве позиций, остаётся трудоёмким, а защитные механизмы маркетплейсов (в частности, графические капчи) существенно затрудняют автоматизированный сбор данных, делая ручной мониторинг ещё более затратным по времени. В этой связи разработка инструмента, который способен автоматизировать взаимодействие с маркетплейсом и предоставлять пользователю нужную информацию в удобном формате, становится актуальной задачей. Telegram-боты благодаря своей популярности, простоте использования и богатому API представляют собой идеальную платформу для создания такого помощника, а интеграция бота с методами машинного обучения и компьютерного зрения позволяет ему самостоятельно распознавать капчи, извлекать ценовые данные и выполнять другие действия, имитирующие поведение человека. Таким образом, разработка подобного интеллектуального инструмента является **актуальной задачей**.

Целью работы является разработка интеллектуального Telegram-бота «Ассистент покупателя», использующего методы машинного обучения для автоматизации получения информации с маркетплейса, и оценка его эффективности на основе анализа работы и востребованности у пользователей.

Для достижения поставленной цели решены следующие **задачи**:

1. изучены современные методы машинного обучения и компьютерного зрения, применимые для автоматизации взаимодействия с веб-интерфейсами;
2. рассмотрены инструменты и технологии создания Telegram-ботов и их интеграции с внешними сервисами;
3. определён технологический стек для разработки системы автоматизации;
4. разработан прототип Telegram-бота «Ассистент покупателя», способного обрабатывать ссылки на товары и текстовую информацию, распознавать капчи и извлекать ценовую информацию;
5. проведено тестирование разработанного прототипа на реальных данных маркетплейса;

6. выполнен статистический анализ и визуализация результатов работы для оценки влияния бота на покупательскую активность.

Объектом исследования являются процессы автоматизации взаимодействия с маркетплейсами, а также поведение покупателей в условиях использования интеллектуальных систем. В рамках работы рассматриваются как технические аспекты автоматизации (взаимодействие с веб-интерфейсами, обход защитных механизмов), так и поведенческие аспекты (частота обращений, предпочтения пользователей, их реакция на скорость получения информации). Исследование охватывает полный цикл — от момента отправки пользователем запроса до получения итоговой цены и фиксации этого события в логах для последующего анализа.

Предметом исследования выступают методы машинного обучения, в частности алгоритмы компьютерного зрения и классификации изображений, а также программные средства и технологические решения, применяемые для создания Telegram-бота, способного автоматизировать сбор данных с маркетплейсов. Особое внимание уделяется эффективности применения свёрточных нейронных сетей для распознавания капчи, методам оптимизации процессов автоматизации, а также подходам к сбору и статистической обработке данных о работе бота с целью оценки его влияния на покупательскую активность.

Практическая значимость работы заключается в создании полезного инструмента, который может быть использован покупателями для упрощения поиска цен, а также в получении объективных данных о востребованности и эффективности подобной автоматизации.

Работа состоит из введения, трёх глав, заключения, списка использованных источников и приложений с исходным кодом ключевых модулей. В первой главе рассматриваются теоретические основы методов машинного обучения, необходимых для реализации функций бота: принципы работы нейронных сетей, архитектуры свёрточных сетей для классификации изображений, концепции интеллектуальных агентов. Вторая глава посвящена обзору технологий и инструментов, необходимых для разработки: средств создания Telegram-ботов, библиотек компьютерного зрения и глубокого обучения, систем оптического распознавания символов и изображений, а также инструментов автоматизации взаимодействия с мобильными устройствами. В

третьей главе описывается реализация прототипа Telegram-бота «Ассистент покупателя»: его архитектура, ключевые модули и их взаимодействие, экспериментальное тестирование на реальных данных и статистический анализ влияния бота на поведение пользователей. Работа содержит 1 таблицу и 16 рисунков.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первая глава посвящена теоретическим основам методов машинного обучения, необходимых для реализации функций бота.

Рассмотрены три основные парадигмы обучения: обучение с учителем (классификация и регрессия на парах «объект — правильный ответ»), обучение без учителя (поиск скрытой структуры в данных, кластеризация, понижение размерности) и обучение с подкреплением (выработка стратегии поведения агента, максимизирующей суммарную награду). Описана работа искусственного нейрона: каждый нейрон вычисляет взвешенную сумму входных сигналов $s = \sum_{i=1}^{n} w_i x_i + b$ и пропускает её через нелинейную функцию активации $y = f(s)$; рассмотрены наиболее распространённые функции активации — сигмоида, гиперболический тангенс и ReLU. Описан механизм обучения сети методом обратного распространения ошибки в сочетании с градиентными оптимизаторами (SGD, Adam, RMSProp), а также методы борьбы с переобучением — Dropout, L1/L2-регуляризация, ранняя остановка и аугментация данных.

Отдельно рассмотрена задача классификации изображений и архитектуры свёрточных нейронных сетей (CNN). Показано, что свёрточный слой применяет набор обучаемых фильтров, формирующих карты признаков по формуле $S(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n)$, а пулинговые слои снижают размерность и повышают устойчивость признаков к локальным искажениям. Прослежена эволюция архитектур CNN — от AlexNet и VGGNet до архитектуры ResNet, ключевой особенностью которой являются остаточные (skip) соединения, решающие проблему затухания градиента и позволяющие обучать сети с сотнями слоёв. Обоснована эффективность переноса обучения (Transfer Learning) для задач с ограниченным объёмом размеченных данных. Рассмотрены метрики качества классификации — Accuracy, Precision, Recall, F1-score, ROC-AUC — и обоснован выбор архитектуры **ResNet18** как оптимального баланса между точностью распознавания и вычислительной эффективностью для задачи распознавания капчи.

Также рассмотрено понятие интеллектуального агента как программной сущности, обладающей автономностью, реактивностью, проактивностью и социальностью, и понятие мультиагентной системы как совокупности взаимодействующих агентов с механизмами координации и коммуникации

(KQML, FIPA ACL). Показано, что в разработанном боте отдельные программные модули (приём сообщений, распознавание капчи, управление устройством, логирование) реализуют принципы агентного подхода, а координацию осуществляет центральный модуль-диспетчер. Описан инструмент Android Debug Bridge (ADB) как средство, позволяющее агенту воздействовать на среду — эмулировать касания, вводить текст, делать снимки экрана — и тем самым имитировать поведение реального пользователя.

Вторая глава посвящена обзору технологий и инструментов, необходимых для разработки.

Проведён сравнительный анализ фреймворков для создания Telegram-ботов на Python — синхронной библиотеки `python-telegram-bot` и асинхронной библиотеки `aiogram`. Выбор сделан в пользу `aiogram`, архитектура которого построена на диспетчере, цепочке middleware и машине состояний (FSM); асинхронная модель позволяет одновременно обслуживать множество пользователей без блокировки, что критично при выполнении длительных операций (запуск ADB-команд, распознавание изображений).

Рассмотрены библиотеки компьютерного зрения и машинного обучения: **PyTorch** — фреймворк глубокого обучения, используемый для инференса предобученной модели ResNet18; **OpenCV** — библиотека для предварительной обработки скриншотов (выделение областей, изменение размера, бинаризация); **Tesseract OCR** — система оптического распознавания символов, применяемая для извлечения цены из области экрана. Подробно разобрана структура сети ResNet18: последовательность слоёв `Layer1`–`Layer4`, устройство остаточных блоков BasicBlock, изменение размерности тензоров признаков (с $(64, H, W)$ до $(512, H/8, W/8)$) и завершающий слой глобального усредняющего пулинга `AdaptiveAvgPool2d`.

Описан инструмент автоматизации взаимодействия с мобильными устройствами ADB, его клиент-серверная архитектура и возможности (запуск приложений, захват экрана, эмуляция жестов и ввода текста), а также способ управления ADB из Python через модуль `subprocess` с обработкой таймаутов и повторных попыток. Рассмотрены средства сбора, обработки и визуализации данных — формат логирования CSV, библиотека **Pandas** для статистической агрегации и библиотеки **Matplotlib/Seaborn** для построения линейных, столбчатых, круговых диаграмм и тепловых карт. По результатам

сравнительного анализа выбран итоговый технологический стек: `aiogram`, `PyTorch`, `OpenCV`, `Tesseract OCR`, `ADB`, `Pandas`, `Matplotlib`, `Seaborn`.

Третья глава содержит описание реализации прототипа, его тестирования и анализа результатов.

Описан процесс регистрации бота через официальный сервис `@BotFather`: получение токена доступа, настройка описания и команд `/start`, `/help`, выбор идентификатора `dwm_store_bot`. Представлена модульная архитектура системы, состоящая из трёх компонентов: `telegram_bot.py` — главный модуль обработки сообщений на основе `aiogram`; `ML10.py` — модуль автоматизации, реализующий взаимодействие с устройством через ADB, обнаружение и распознавание капчи и извлечение цены через OCR; `Calculator_bot.py` — модуль расчёта итоговой стоимости с получением курса юаня с сайта ЦБ РФ. Такая структура обеспечивает гибкость, масштабируемость и удобство отладки.

Описан пользовательский интерфейс бота: главная клавиатура с кнопками «Расчёт по ссылке», «Расчёт по цене» и «Помощь», автоматическое распознавание ссылок и чисел в произвольном тексте, инлайн-кнопки «Сделать заказ» и «Новый запрос», а также механизмы обработки ошибок ввода.

Подробно рассмотрен **модуль расчёта по ручному вводу**: извлечение числа из текста регулярным выражением, проверка диапазона (от 1 до 10 000 000 юаней), получение актуального курса юаня к рублю с сайта Центрального банка РФ (с резервным значением 12,5 руб./юань при недоступности источника) и вычисление итоговой цены по формуле

$$\text{Цена в рублях} = (\text{Курс} + 0{,}8) \times \text{Цена в юанях} + 1200,$$

где коэффициент 0,8 учитывает комиссию за конвертацию, а константа 1200 — фиксированные расходы на доставку до Москвы.

Подробно описан **модуль автоматического получения цены по ссылке** (класс `CaptchaAnalyzer`). Обоснована необходимость создания собственного датасета изображений капчи (вместо CIFAR-10, STL-10 или ImageNet) ввиду специфического набора классов (автомобиль, собака, кошка, велосипед, поезд, корабль, лошадь, диван, баскетбольный мяч) и характерного стилизованного векторного оформления изображений. Собрано около 1500 размеченных изображений, разделённых в соотношении 80/10/10 на обучающую, валидационную и тестовую выборки; применена аугментация

(повороты $\pm 15^{\circ}$, сдвиги, нормализация). Модель ResNet18 дообучена на 25 эпохах с оптимизатором Adam ($lr = 0{,}001$, размер пакета 32, функция потерь — перекрёстная энтропия); итоговая точность на тестовой выборке составила **94,2%**, усреднённый F1-score — 0,94.

Описан полный алгоритм работы модуля: открытие ссылки на Android-устройстве через ADB, захват скриншота, обнаружение капчи методом скользящего окна OpenCV, определение целевого слова сопоставлением с шаблонами (`cv2.matchTemplate`), классификация изображений-вариантов нейросетью `NeuralNetworkPredictor`, эмуляция перетаскивания выбранной картинки в бланк ответа (`drag_to_blank` через `adb shell input swipe`) и извлечение цены из области экрана средствами Tesseract OCR после предварительной бинаризации изображения методом Отсу.

Представлены результаты экспериментального тестирования прототипа на реальных данных маркетплейса в течение 32 дней (с 6 марта по 6 апреля 2026 года), в ходе которого велось логирование всех запросов в формате CSV (метка времени, идентификатор пользователя, тип запроса, успешность, статус решения капчи, цена, время обработки, факт перехода к менеджеру). За этот период бот обработал **847 запросов** от **201 уникального пользователя** (в среднем около 26 запросов в день, пик — 49 запросов в субботу 22 марта). Запросы по ссылке составили 68% (576), ручной ввод — 32% (271). Капча возникала в 412 из 576 ссылочных запросов (71,5%); из них решена с первой попытки в 90% случаев, ещё 5,3% потребовали повторных попыток, а итоговая успешность автоматического решения капчи с учётом повторов достигла **95,4%**. Среднее время ответа для ссылочных запросов составило 26,9 секунды, для ручного ввода — 1,2 секунды.

Выполнено сравнение ключевых показателей до и после внедрения бота (за аналогичный 32-дневный период). Число обращений к менеджеру снизилось со 192 до 113, при этом число реальных заказов выросло с 31 до 68, а конверсия из обращений в заказы увеличилась с **16,1% до 60,1%**. Это свидетельствует о том, что автоматизация расчётов не только привлекла новую аудиторию, но и существенно повысила качество лидов: обращающиеся пользователи стали более целевыми и готовыми к покупке, а сам процесс получения цены товара стал для них значительно более удобным и прозрачным.

ЗАКЛЮЧЕНИЕ

Современный этап развития электронной коммерции характеризуется высокой динамикой цен и широким внедрением защитных механизмов, таких как графические капчи, которые существенно затрудняют автоматизированное получение информации о товарах. В этих условиях разработка интеллектуальных помощников, способных имитировать действия реального пользователя и применять методы компьютерного зрения для обхода ограничений, приобретает особую актуальность.

В рамках данной работы были решены все поставленные задачи. Проведён анализ современных методов машинного обучения и компьютерного зрения, применимых для автоматизации взаимодействия с веб-интерфейсами; рассмотрены базовые парадигмы обучения, архитектуры свёрточных нейронных сетей, включая ResNet, и концепция интеллектуальных агентов. Изучены и систематизированы инструменты и технологии создания Telegram-бота и его интеграции с внешними сервисами, определён итоговый технологический стек (aiogram, PyTorch, OpenCV, Tesseract OCR, ADB).

Разработан прототип Telegram-бота с модульной архитектурой, реализующий два сценария работы — ручной ввод цены и автоматическое получение цены по ссылке с решением графической капчи. Для распознавания капчи сформирован собственный датасет из примерно 1500 изображений, на котором дообучена модель ResNet18, достигшая точности классификации 94,2% на тестовой выборке.

Проведено тестирование разработанного прототипа на реальных данных маркетплейса в течение 32 дней: бот обработал 847 запросов от 201 уникального пользователя, успешность автоматического решения капчи с учётом повторных попыток достигла 95,4%, среднее время ответа на ссылочные запросы составило 26,9 секунды. Сравнение ключевых показателей до и после внедрения бота показало снижение числа обращений к менеджеру со 192 до 113 при росте числа реальных заказов с 31 до 68, а конверсия из обращений в заказы увеличилась с 16,1% до 60,1%.

Таким образом, цель бакалаврской работы — разработка интеллектуального Telegram-бота «Ассистент покупателя» с применением методов машинного обучения и оценка его эффективности — полностью достигнута. Полученные результаты подтверждают практическую пригодность

разработанного инструмента и целесообразность применения методов машинного обучения для автоматизации взаимодействия с маркетплейсами.

Основные источники информации:

1 Flach P. Machine Learning: The Art and Science of Algorithms that Make Sense of Data. – Cambridge University Press, 2012. – 396 p.

2 Shalev-Shwartz S., Ben-David S. Understanding Machine Learning: From Theory to Algorithms. – Cambridge University Press, 2014. – 449 p.

3 Goodfellow I., Bengio Y., Courville A. Deep Learning. – MIT Press, 2016. – 800 p.

4 He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. – 2016. – P. 770–778.

5 Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks // Advances in Neural Information Processing Systems. – 2012. – Vol. 25. – P. 1097–1105.

6 Aiogram Documentation [Электронный ресурс]. – Электронные данные. – Режим доступа: URL.: https://docs.aiogram.dev/ (дата обращения: 21.03.2026).

7 PyTorch Documentation [Электронный ресурс]. – Электронные данные. – Режим доступа: URL.: https://pytorch.org/docs/ (дата обращения: 24.03.2026).