DOI: 10.58168/PPSPM2025_372-379
УДК 519.713.1

ОСОБЕННОСТИ РЕАЛИЗАЦИИ ФРОНТЕНДА ДЛЯ ПРИЛОЖЕНИЙ С МАШИННЫМ ОБУЧЕНИЕМ

Е.А. Кирин, И.Р. Евчук

ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

Аннотация: статья посвящена современным подходам к frontend - разработке, интегрированных с моделями машинного обучения для задач распознавания документов. Рассматриваются ключевые аспекты построения пользовательского интерфейса: организация процессов загрузки и предобработки файлов, визуализация результатов классификации и извлечения текстовой информации, а также обеспечение интерактивного взаимодействия с моделью. Приводится технический анализ особенностей интеграции клиентской части с API машинного обучения, включая вопросы оптимизации запросов, асинхронной обработки и отображения промежуточных состояний. Особое внимание уделяется проблемам юзабилити, связанным с неопределённостью ответов ИИ и необходимостью валидации результатов пользователем. В заключение обозначается значимость frontend-разработки как посредника между пользователем и сложной интеллектуальной системой, что определяет его роль в успешной практической реализации решений на основе машинного обучения.

Ключевые слова: frontend-разработка, машинное обучение, распознавание документов, интерактивный интерфейс, интеграция с API, асинхронная обработка, визуализация данных, UX.

FEATURES OF FRONT-END IMPLEMENTATION FOR MACHINE LEARNING APPLICATIONS

E.A. Kirin, Evchuk I.R.

Voronezh State University of Forestry and Technologies named after G.F. Morozov

Abstract: This article is dedicated to modern approaches in front-end development integrated with machine learning models for document recognition tasks. Key aspects of building the user

interface are considered, including the organization of file upload and preprocessing processes, visualization of classification results and extracted textual information, as well as ensuring interactive interaction with the model. A technical analysis of the features of integrating the client side with machine learning APIs is provided, including optimization of requests, asynchronous processing, and display of intermediate states. Special attention is given to usability challenges related to the uncertainty of AI responses and the need for user validation of results. In conclusion, the significance of front-end development as an intermediary between the user and a complex intelligent system is highlighted, which determines its role in the successful practical implementation of machine learning-based solutions.

Keywords: front-end development, machine learning, document recognition, interactive interface, API integration, asynchronous processing, data visualization, UX.

Современные информационные системы всё активнее интегрируют технологии машинного обучения (ML), которые позволяют автоматизировать задачи, ранее требовавшие значительных человеческих ресурсов. Одним из востребованных направлений применения является распознавание и интеллектуальная обработка документов — от извлечения текста и классификации до анализа структуры и поиска ключевых данных. Такие решения применяются в банковской сфере, логистике, образовании и государственных учреждениях, где требуется ускорить работу с большими объёмами бумажных и электронных документов. Однако успешное внедрение подобных систем невозможно без создания удобного и функционального пользовательского интерфейса, который обеспечивает взаимодействие человека с моделью машинного обучения. Именно frontend-разработка становится связующим звеном между сложными алгоритмами обработки данных и конечным пользователем, делая результаты работы модели понятными и доступными.

Классические веб-приложения ориентированы преимущественно на статическое взаимодействие с сервером: получение данных, отображение интерфейсов и обработку простых пользовательских запросов. При интеграции моделей машинного обучения специфика frontend-разработки существенно меняется. При реализации системы важно не только обеспечить корректное отображение результата, но и учесть особенности работы модели. Ключевым аспектом является асинхронность выполнения задач, которая требует организации отображения промежуточных статусов для пользователя, таких как загрузка файла, обработка, распознавание и так далее. Кроме того, необходимо учитывать возможность получения неопределённых или частично

корректных ответов, что обуславливает потребность в дополнительных инструментах для их валидации со стороны пользователя. Высокая нагрузка на систему при обработке больших файлов диктует необходимость оптимизации всех взаимодействий с API для обеспечения стабильности. Отдельное внимание следует уделить вопросам защиты данных, поскольку обрабатываемые документы могут содержать конфиденциальную или персональную информацию. Следовательно, специфика создания пользовательского интерфейса в системах, связанных с ML, выходит за рамки классической разработки и требует отдельного рассмотрения.

Frontend-разработка в системах распознавания документов — это не просто оболочка, а критически важный компонент, который выступает посредником между пользователем и сложной логикой машинного обучения. Главная задача заключается в том, чтобы сделать этот процесс прозрачным. На этапе загрузки и подготовки данных должен быть предоставлен понятный интерфейс, к примеру, через перетаскивание или классический диалог выбора файлов. Система обязана проводить предварительную валидацию: проверять тип файла (формат PDF, JPG, JPEG, PNG, TIF, TIFF, GIF, GIFF, BMP, WEBP, DOCX, DOC, RTF, ODT, PPTX, PPT, ODP, XLSX, XLS, ODS), его размер (чтобы не перегрузить сервер) и целостность. Полезной функцией является предпросмотр загруженного документа, чтобы пользователь мог убедиться, что отсканировал нужную страницу и она не перевернута, не обрезана и не размыта. На этом же этапе предлагается базовая клиентская обработка: коррекция ориентации изображения или его обрезка.

Также важной задачей frontend-разработки является интерактивная визуализация результатов. Просто вывести распознанный текст недостаточно. Пользователь должен видеть, что и с какой уверенностью распознала модель. Для этого применяется техника привязки распознанного текста (OCR) к исходным областям изображения. Текст выделяться цветными рамками, где цвет указывает на уровень уверенности модели: зеленый для высоких значений, желтый для средних и красный для низких. Это сразу направляет внимание пользователя на потенциальные проблемы. Аналогично, для классификации документов полезно визуализировать вероятности, в виде графика или процентов рядом с предполагаемым типом документа («Паспорт — 97%, Водительские права — 3%»). Такой подход превращает результат модели в понятный инструмент.

Поскольку модели не идеальны, фронтенд должен быть спроектирован для обработки неопределенности. Он не должен требовать от системы стопроцентной точности, а вместо этого давать пользователю простые и быстрые инструменты для исправления ошибок. Распознанные данные должны отображаться в интерактивных полях ввода, которые можно легко редактировать. Для полей с низкой уверенностью можно автоматически подсвечивать их. Эффективным решением является интерфейс сплит-вида, где справа отображается оригинал документа, а слева форма с распознанными данными, что позволяет быстро сверять информацию и вносить правки.

Интеграция с моделями машинного обучения — это нетривиальная задача, напрямую влияющая на удобство работы пользователя. Процесс распознавания документов может занимать от нескольких секунд до нескольких минут, поэтому интерфейс не должен «зависать» в ожидании ответа. Нужно реализовать механизм обработки долгих запросов через web-sockets. Пользователь должен видеть понятные статусы — «распознавание выполняется», «готово» или «произошла ошибка».

Важно также предусмотреть надёжную обработку ошибок — от таймаутов и сбоев соединения до внутренних ошибок ML-модели и показывать понятные сообщения, объясняющие, что произошло и что можно сделать дальше.

Все эти функции должны быть объединены в удобный и доступный интерфейс. Он должен быть понятным даже для новичков. Каждый шаг процесса нужно сопровождать ясными инструкциями и подсказками. Все элементы интерфейса должны быть доступны с клавиатуры, а визуальные подсказки продублированы текстом для людей с ограниченными возможностями. Хорошо продуманный интерфейс упрощает сложные технологии, делая их использование легким и предсказуемым — это и обеспечивает успех решения.

Создание подобного интерфейса требует больших ресурсов и высокой квалификации разработчиков. Поэтому на этапе проектирования архитектуры и выбора технологий важно тщательно изучить уже существующие решения. Анализ их возможностей, ограничений, особенностей лицензирования и архитектуры помогает лучше понять рынок, определить сильные и слабые стороны конкурентов и сформулировать требования к собственной системе. Такой подход позволяет использовать проверенные практики и избежать типичных ошибок при разработке.

На основе актуальных данных был проведен сравнительный анализ уже существующих решений на рынке систем распознавания документов, каждый из которых демонстрирует уникальный подход к архитектуре и набору поддерживаемых функций.

ContentaAI представляет собой корпоративное решение, ориентированное на автоматизацию ввода и классификации документов. Система поставляется в виде готового коробочного продукта и может быть развернута как локально, так и в облачной среде. Гибкая система шаблонов позволяет создавать и редактировать правила извлечения данных без привлечения разработчиков или специалистов от вендора, что делает систему удобной для масштабного корпоративного использования. К недостаткам можно отнести ограниченную поддержку форматов исходных документов — система не работает напрямую с файлами Microsoft Office, требуя предварительного преобразования в графический.

ContentReader Engine, также от ContentaAI, представляет собой комплект (SDK) для интеграции возможностей оптического распознавания текста (OCR) в сторонние приложения и системы. Система способна работать более чем с 200 языками, включая редкие и исторические, а также языки программирования. Поскольку ContentReader Engine не имеет готового пользовательского интерфейса и инструментов визуальной настройки, интеграция требует участия квалифицированных разработчиков. Клиенту необходимо самостоятельно реализовывать интерфейс, логику обработки исключений, управление задачами и хранение результатов. Это увеличивает затраты на внедрение и повышает требования к технической команде.

Dbrain облачная платформа, специализирующаяся на автоматической верификации документов и проверке подлинности личности. В отличие от классических систем OCR, Dbrain делает акцент не столько на извлечении данных из документов, сколько на подтверждении их достоверности и предотвращении мошенничества. Одним из главных преимуществ Dbrain является наличие встроенного антифрод-модуля, способного определять признаки подделки документов, несанкционированных изменений изображений и подмены данных. Платформа Dbrain работает в облаке и предоставляет готовый пользовательский интерфейс станции верификации, через который операторы могут просматривать результаты проверки, подтверждать или отклонять документы. К недостаткам можно отнести ограничение у пользователей, которые не могут самостоятельно создавать или настраивать

шаблоны документов — этот процесс жёстко контролируется поставщиком, что снижает адаптивность системы под специфические нужды организации.

SmartEngines является узкоспециализированным вендором с фокусом на безопасности и проверке подлинности. Компания специализируется на высокоточном анализе изображений и безопасности, что делает её продукты востребованными в сферах, где критически важно качество верификации — в банках, аэропортах, государственных системах идентификации и крупных корпоративных службах безопасности. SmartEngines поддерживает более 3000 типов документов. Система также отличается высокой скоростью обработки и низкими требованиями к ресурсам, что позволяет внедрять её даже на мобильных устройствах или встроенных терминалах. Продукт поставляется исключительно в виде SDK, требующих интеграции в существующие программные продукты. Для внедрения такого решения необходима команда разработчиков и системных архитекторов, что увеличивает затраты и время развертывания.

На фоне представленных аналогов проект Атом.ОКО демонстрирует готовую платформу для корпоративных решений. Проект ориентирован на создание гибкой, настраиваемой среды для интеллектуальной обработки документов, где пользователь может не только работать с готовыми функциями, но и самостоятельно выстраивать бизнес-логику под конкретные задачи. Ключевыми компонентами системы являются классификатор документов, обучаемый на пользовательских данных, шаблонизатор для извлечения структурированной информации, настройка атрибутов, файловая система с папками и персональными документами пользователя и интерактивная среда построения бизнес-процессов, реализованная на основе React Flow. Последняя позволяет визуально конструировать логику обработки — от этапа загрузки и распознавания до сохранения результатов и интеграции с внешними системами.

Кроме того, система включает административный модуль, обеспечивающий управление пользователями, ролями и правами доступа к различным типам документов. Это делает платформу масштабируемой и пригодной к использованию как в небольших компаниях, так и в крупных организациях.

К его ключевым преимуществам можно отнести российское происхождение, возможность работы в изолированном контуре и отсутствие ограничений по масштабированию.

Основными зонами роста для Атом.ОКО являются расширение поддержки языков распознавания за пределы кириллицы, латиницы и китайского, а также увеличение количества готовых шаблонов документов по сравнению с такими вендорами, как SmartEngines. При этом проект представляет собой перспективное отечественное решение, которое благодаря высокой гибкости и активному развитию имеет все шансы занять стабильную позицию на рынке. Особенно это актуально для корпоративных заказчиков, ценящих возможность глубокой кастомизации процессов и независимость от сторонних API.

Таким образом, современные системы распознавания документов объединяют технологии машинного обучения и удобный пользовательский интерфейс. Фронтенд в таких решениях уже не просто показывает результаты работы алгоритма, а играет активную роль в организации взаимодействия пользователя с системой. Успех системы во многом зависит от того, насколько легко и удобно пользователь может выполнять свои задачи — от загрузки документа до проверки и корректировки извлечённой информации.

Проведенный анализ показывает, что проектирование такого интерфейса имеет свои особенности по сравнению с обычной фронтенд-разработкой. Необходимо учитывать промежуточные статусы распознавания, работать с областями документа, в которых модель не уверена, и создавать инструменты для простой проверки и исправления данных. Эти принципы были учтены при сравнении существующих решений на рынке, где можно выделить как готовые платформы с развитым интерфейсом, так и специализированные SDK, которые требуют глубокой интеграции в собственные системы.

Список литературы

1. Nadukuda, N. Automating Front-End Development with AI: From Code Generation to Intelligent Debugging // International Journal of Artificial Intelligence & Applications (IJAIAP). – 2023. – сайт URL: https://iaeme.com (дата обращения: 06.09.2025).

2. Goh, H. A. Front-end Deep Learning Web Apps Development and Deployment // International Journal of Computational Intelligence Systems. – Springer, 2023.

3. Благодельский, А. С. Применение искусственного интеллекта в фронтенд-разработке: от теории к практике // Информационные технологии и

программирование. – 2024. – сайт URL: https://cyberleninka.ru (дата обращения: 06.09.2025).

4. Ведьманов, И. С. Интеллектуальные технологии для формирования взаимодействия компонентов системы: архитектура с фронтендом, бэкендом и ML-сервисом // Информационные системы и модели процессов. – 2025. – сайт URL: https://mathnet.ru (дата обращения: 06.09.2025).

5. Давлетов, А. Р. Главные трудности при интеграции машинного обучения в коммерческую эксплуатацию // Современные технологии. – 2023. – сайт URL: https://cyberleninka.ru (дата обращения: 06.09.2025).

6. Машинное обучение в продуктовой разработке, где его не ожидают: сайт URL: https://habr.com/ (дата обращения: 06.09.2025)

7. AI для frontend: модели для генерации интерфейса: сайт URL: https://tproger.ru (дата обращения: 06.09.2025)

References

1. Nadukuda, N. Automating Front-End Development with AI: From Code Generation to Intelligent Debugging // *International Journal of Artificial Intelligence & Applications (IJAIAP)*. – 2023. – URL: https://iaeme.com (accessed: 06.09.2025).

2. Goh, H. A. Front-end Deep Learning Web Apps Development and Deployment // *International Journal of Computational Intelligence Systems*. – Springer, 2023.

3. Blagodelsky, A. S. Application of Artificial Intelligence in Front-End Development: From Theory to Practice // *Information Technologies and Programming*. – 2024. – URL: https://cyberleninka.ru (accessed: 06.09.2025).

4. Vedmanov, I. S. Intelligent Technologies for Organizing Component Interaction: Architecture with Front-End, Back-End, and ML Service // *Information Systems and Process Models*. – 2025. – URL: https://mathnet.ru (accessed: 06.09.2025).

5. Davletov, A. R. Main Challenges in Integrating Machine Learning into Commercial Operation // *Modern Technologies*. – 2023. – URL: https://cyberleninka.ru (accessed: 06.09.2025).

6. Machine Learning in Product Development Where It Is Least Expected // URL: https://habr.com (accessed: 06.09.2025).

7. AI for Front-End: Models for Interface Generation // URL: https://tproger.ru (accessed: 06.09.2025).