**2** СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ

В этом будет произведено разделение программного средства, реализуемого в данном дипломном проекте, на структурные блоки. Для каждого блока ниже будут описаны назначение блока, краткое описание и связь с остальными блоками.

Для данного дипломного проекта была выбрана архитектура «клиент- сервер» (рисунок 2.1). Серверная часть является основной частью проекта, так как именно она будет содержать свёрточную нейронную сеть, позволяющую колоризировать полутоновые изображения. Предполагается, что серверная часть будет содержать следующие блоки:

* блок колоризации изображений;
* блок обучения нейронной сети;
* блок хранения обработанных изображений;
* блок внешнего интерфейса серверной части.

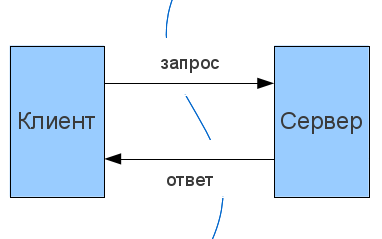


Рисунок 2.1 – Архитектура «клиент-сервер»

Блок колоризации изображений является основным структурным блоком всего дипломного проекта. Этот блок предназначен для колоризации полутоновых изображений. На вход данного блока поступают чёрно-белые изображения из блока внешнего интерфейса серверной части. Колоризованные изображения поступают в блок хранения обработанных изображений. Блок колоризации содержит нейронную сеть, которая в свою очередь не будет функционировать, если не произвести процесс обучения сети. Потому данный блок так же будет взаимодействовать с блоком обучения нейронной сети.

Блок обучения нейронной сети – структурный блок, который является необходимым для обучения нейронной сети. Свёрточные нейронные сети на практике обычно обучаются с использованием «учителя». Обучение с учителем – один из способов [машинного обучения](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), в ходе которого испытуемая система принудительно обучается с помощью примеров «стимул-реакция». Данный блок будет предоставлять блоку, содержащему нейронную сеть, полутоновые изображения, а также идеальный вариант того, как сеть должна обработать изображение. Для реализации подобного блока необходима база изображений, которые являются цветными. Эти изображения будут использоваться для предоставления нейронной сети идеального варианта её работы. В качестве же исходного изображения для сети будут использованы преобразованные в полутон изображения из подготовленной базы – информация о яркости каждого пикселя исходного изображения. Из этого ясно, что блок обучения будет содержать в себе логику, которая позволит попиксельно преобразовать цветное изображение в полутоновое. Алгоритм данного преобразования будет описан ниже.

Также важно понимать, что блок обучения необходим только на стадии проектирования сети и в процессе переобучения нейронной сети. В остальных случаях блок не задействуется и может быть исключён из готового сервиса

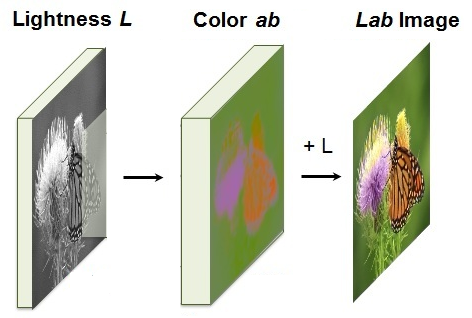


Рисунок 2.2 – Пример входного и выходного интерфейсов свёрточной нейронной сети, выполняющей задачу колоризации

Блок хранения обработанных изображений будет разработан для хранения колоризированных нейронной сетью изображений, которые готовы для передачи на клиентскую часть. Данный блок необходимо реализовать, потому что работа нейронной сети, особенно специализированной на обработку изображений, занимает относительно большое время. Если клиент будет пересылать изображение по сети, используя стандартный HTTP протокол, то вероятнее всего из-за длительной обработки полутонового изображения сервером произойдёт истечение времени для ожидания ответа на запрос. Поэтому было решено, что сервер после принятия сообщения будет сообщать клиенту о начале обработки переданного изображения свёрточной сетью либо же сообщать о невозможности обработать изображение на данный момент. После колоризации данный блок будет принимать для временного хранения обработанные изображения, а оповещать блок внешнего интерфейса о необходимости оповестить клиентское приложение об окончании обработки полутонового изображения серверной частью. Средства необходимые для реализации такой логики будут подробнее описаны в следующих разделах пояснительной записки.

Блок интерфейса серверной части является одним из звеньев связывающих серверную часть с клиентским приложением. Данный блок выполняет следующие функции:

* получение полутоновых изображений от клиента;
* оповещение клиента о приёме нейронной сетью изображения на обработку либо о невозможности обработки изображения в данный момент времени;
* оповещение о готовности передачи окрашенного нейронной сетью изображения обратно на клиентское приложение;
* передача обработанного изображения;
* проверка получаемых изображений;
* обработка ошибок, связанных со сбоем сети.

Разработка внешних интерфейсов серверной и клиентской частей представляет построение серверного API для дальнейшего его предоставления клиентской части и проектирование протокола прикладного уровня.

Клиентское мобильное приложение будет включать в себя следующие структурные блоки:

* блок получения изображения;
* блок предобработки изображения;
* блок внешнего интерфейса клиентской части;
* блок пользовательского интерфейса.

Блок получения изображения является источником для изображений, обработанные версии которых будут использоваться в качестве исходных изображений для свёрточной нейронной сети. Как было изложено ранее, если рассматривать данный дипломный проект в качестве средства для реставрации старых чёрно-белых фотографий или киноплёнок, то выбор платформы для клиента, должен зависеть от того, будет ли поддерживать данная платформа оборудование и программное обеспечение для оцифровки аналоговых источников. Так как доступ к подобному обеспечению и оборудованию отсутствует в рамках данного дипломного проекта, было дано предпочтение реализации клиентской части в качестве мобильного, а не прикладного приложения. Такой выбор оправдан, если позиционировать проект в качестве демонстрации работы свёрточной нейронной сети, выполняющей задачу колоризации изображения, так как мобильное приложение имеет возможность получить фотографию с помощью камеры, которой на текущий момент оборудовано практически любое мобильное устройство.

Реализация структурного блока получения изображения будет разниться от платформы, к платформе. Однако, так как предполагается, что именно операционная система Android будет являться платформой для клиентской части, исходя из опыта работы, можно сказать, что данный структурный блок будет представлять собой часть встроенного фреймворка для работы с камерой мобильного устройства, а также получением изображения из хранилища данных смартфона. Кроме того, этот блок будет обрабатывать исключительные ситуации, в случае если на устройство не будет оборудовано встроенной камерой или же доступ к ней будет блокирован системой.

Блок предобработки изображения необходим для преждевременной обработки полученного из предыдущего блока изображения. К подобной обработке можно отнести уменьшение размера изображения, его сжатие для передачи по сети. Кроме того, планируется реализация возможности получить полутоновое изображение из цветного, до его отправления на серверную часть. Данный структурный блок напрямую связан с внешним интерфейсом клиентской части. Более подробное описание реализации будет предоставлено в следующих разделах пояснительной записки.

Блок внешнего интерфейса клиентской части будет выполнять следующие функции:

* передача подготовленных изображений на сервер;
* ожидание оповещений от сервера о конце операции обработки изображения нейронной сетью;
* получение колоризированных нейронной сетью изображений от серверной части;
* обработка исключительных ситуаций, связанных со сбоем сети, или отказом сервера в обслуживании.

Сам по себе блок внешнего интерфейса клиентского приложения будет представлять собой надстройку над сторонней библиотекой Retrofit 2, краткое описание которой было приведено в обзоре литературы. Более подробное описание этой библиотеки, а также пример использования будет приведён в дальнейшем.

Блок пользовательского интерфейса является основным блоком для клиентской части. Данный блок в разной степени связан со всеми остальными структурными блоками мобильного приложения. Основными функциями данного блока являются реагирование на действие пользователя, а также отображение информации в том числе – обработанных изображений, полученных от сервера.

В целом же любой блок клиентской части проекта можно разделить при помощи MVC – схема разделения данных приложения, [пользовательского интерфейса](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) и управляющей логики на три отдельных компонента: модель, представление и контроллер (рисунок 2.3).

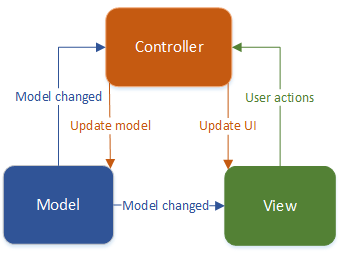


Рисунок 2.3 – Паттерн MVC