|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

***Мультимодальное обучение по тексту и изображениям. Создание описаний по изображениям***

Студент \_\_ИУ-5 84Б\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Р.Р. Сафин\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_О.О. Варламов\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Ю.Е. Гапанюк\_\_**

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Ю.Н.Кротов\_\_**

(Подпись, дата) (И.О.Фамилия)

2021 г.

# РЕФЕРАТ

Расчётно-пояснительная записка квалификационной работы бакалавра содержит 59 страниц. С приложениями объем составляет 84 страницы. Работа включает в себя 22 таблицы и 11 иллюстраций. В процессе выполнения было использовано 13 источников.

Объектом разработки является мультимодальная информационная система для создания текстовых описаний изображений.

Целью работы является разработка и реализация архитектуры мультимодальной системы. Данная система должна генерировать текстовое описание изображений, подаваемых в систему. В дальнейшем данные описания могут быть использованы для улучшения работы систем, основанных на работе с двумя модальностями.

Разработка мультимодальной информационной системы проводилась на основании задания на выполнение выпускной квалификационной работы бакалавра, утвержденного заведующим кафедрой ИУ5 МГТУ им. Н.Э. Баумана, и документа «Техническое задание», утверждённого руководителем выпускной квалификационной работы.

Расчетно-пояснительная записка состоит из нескольких частей.

Постановка задач разработки содержит описание предметной области, функциональные характеристики системы.

Исследовательская часть содержит описание и сравнение различных вариантов архитектурных решений, нацеленных на генерацию описаний к изображениям.

Конструкторско-технологическая часть содержит описание архитектуры разрабатываемой системы, описание используемых средств и библиотек.

Расчетно-пояснительная записка содержит 3 приложения.

# СОДЕРЖАНИЕ

[РЕФЕРАТ 2](#_Toc43235056)

[СОДЕРЖАНИЕ 3](#_Toc43235057)

[СПИСОК ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ 4](#_Toc43235058)

[ВВЕДЕНИЕ 5](#_Toc43235059)

[1 ПОСТАНОВКА ЗАДАЧ РАЗРАБОТКИ 6](#_Toc43235060)

[1.1 Постановка задач разработки 6](#_Toc43235061)

[1.2 Описание предметной области 7](#_Toc43235061)

[1.3 Функциональные задачи системы 7](#_Toc43235061)

[1.4 Обоснование критериев качества 8](#_Toc43235061)

[1.5 Сравнительный анализ аналогов 9](#_Toc43235061)

[2 ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ 14](#_Toc43235066)

[2.1 Исследование существующих архитектур 14](#_Toc43235067)

[2.1.1 Архитектура RCNN + BRNN 14](#_Toc43235068)

[2.1.2 Архитектура CNN + CNN 19](#_Toc43235069)

[2.1.3 Архитектура CNN-RNN 26](#_Toc43235070)

[2.2 Сравнение вариантов архитектур 32](#_Toc43235072)

[3 КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 37](#_Toc43235060)

[3.1 Разработка архитектуры системы 37](#_Toc43235073)

[3.2 Используемые при разработке технологии 38](#_Toc43235074)

[3.3 Обработка датасета 40](#_Toc43235079)

[3.4 Энкодер CNN 42](#_Toc43235080)

[3.5 Декодер RNN 45](#_Toc43235081)

[3.6 Обучение модели 47](#_Toc43235082)

[3.7 Модуль перевода описания текста 49](#_Toc43235083)

[3.8 Создание веб-приложения 53](#_Toc43235084)

[ЗАКЛЮЧЕНИЕ 57](#_Toc43235087)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 58](#_Toc43235088)

[ПРИЛОЖЕНИЕ А 60](#_Toc43235089)

[ПРИЛОЖЕНИЕ Б 70](#_Toc43235091)

[ПРИЛОЖЕНИЕ В 78](#_Toc43235148)

СПИСОК ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ

MIT — Multimodal Image-to-Text, название разрабатываемой мультимодальной информационной системы

NLP — Natural Language Processing, область машинного обучения посвященная обработке естественного языка

Flickr8K, Flickr30K — наборы данных для задач обнаружения объектов на изображении, сегментации изображений и создания описаний для изображений

Фреймворк — программная платформа, определяющая структуру программной системы, программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта

Эпоха — одна итерация в процессе обучения модели, включающая предъявление всех примеров из обучающего множества

# ВВЕДЕНИЕ

Современные технологии находятся в постоянном развитии. Если раньше большинство систем в области машинного обучения и искусственного интеллекта были одномодальными, то есть использовали, анализировали и работали только с одной модальностью, то сейчас, возросшие запросы, цели и стремления людей требуют от современных интеллектуальных информационных систем мультимодальности, то есть возможности работать одновременно с несколькими модальностями и преобразовывать данные из одного вида в другой, что напоминает принцип работы человеческого мозга и мышления.

Мы не обращаем внимания, насколько удивителен наш мозг, он автоматически может преобразовывать различные виды модальностей друг в друга. Например, читая какой-то рассказ, мы зачастую можем представить себе некую картину, ситуацию, которая описывается в произведении. Или даже просто смотря на некую картину, мы можем описать то, что видим на ней безо всякого труда, а ведь это по своей сути невероятно сложный когнитивный процесс.

Интеллектуальные системы машинного обучения и искусственного интеллекта лишены возможности делать это всё автоматически, как человеческий мозг, для этого их необходимо сначала обучить, задать явные связи между модальностями, чтобы система могла понимать, как изображение связано с текстом, и наоборот. Благодаря этому система сможет распознавать, анализировать и преобразовывать различную входную информацию одного типа в другой.

Разрабатываемая мультимодальная информационная система позволяет генерировать текстовые логические описания изображений, подаваемых, как входные данные. Эти логические описания в свою очередь могут использоваться в дальнейшем в различных системах, в которых присутствует компьютерное зрение или обработка изображений в том или ином виде.

# ПОСТАНОВКА ЗАДАЧ РАЗРАБОТКИ

## Постановка задач разработки

Уже долгое время во многих системах применяются модели машинного обучения и искусственного интеллекта. И со временем все эти системы развиваются и начинают использовать все больше и больше новых технологий, и методов. Однако не всегда технологии соответствуют должному уровню развития для постоянного использования во многих системах. В последнее время, в системах все чаще начинают появляться технологии, основанные на компьютерном зрении. Однако с этими технологиями не все так однозначно. Иногда в этих системах встречаются проблемы с обнаружением объектов, но намного чаще встречаются проблемы с логической интерпретацией изображений. То есть система может верно распознать объект, но не в состоянии верно определить логическую суть происходящего. А это является крайне важной и необходимой частью подобных систем, автоматизированная система должна иметь возможность логически интерпретировать ситуации из визуальных данных.

Для этого было решено спроектировать и реализовать мультимодальную информационную систему, которая могла бы по вводимым графическим данным верно интерпретировать суть изображенного и на основе этого создавать логическое текстовое описание изображения. Предполагается, что в дальнейшем выводимые данные, а именно логические описания, могут использоваться для принятия тех или иных решений в системе.

Таким образом, разрабатываемая система имеет широкий спектр применения. Может быть встроена в различные системы, использующие компьютерное зрение или графические данные в целом, для логической интерпретации ситуаций, и на основе этого, принятие каких-либо решений.

## Описание предметной области

Предметной областью данной работе, посвященной мультимодальной информационной системе, является совокупность объектов, выполняющих общую цель, а именно обработку изображения, которое посылает на вход системы пользователь, и генерацию логического текстового описания на его основе.

Данная предметная область (рисунок 1) состоит из определенных сущностей и связей между ними. Предметная область рассматривается с точки зрения пользователя, который имеет возможность загрузить изображение в систему и получить в ответ текстовое описание данного изображения.

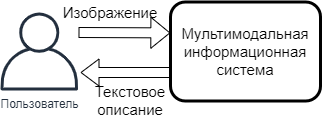


Рисунок 1 – Описание предметной области

Описание сущностей данной предметной области:

* Пользователь – человек, использующий данную информационную систему;
* Мультимодальная информационная система – разрабатываемая система, которая по заданному изображению генерирует логическое текстовое описание;

## Функциональные задачи системы

Разрабатываемая система должна обладать следующим функционалом:

* Разбиение изображения на признаки;
* Высокоточное определение расположения объектов на изображении;
* Хорошая точность генерации логических текстовых описаний изображения;
* Высокая точность перевода сгенерированных описаний;

Пользователь при работе с данной системой должен иметь следующие функции:

* Ввод любого изображения в мультимодальную информационную систему;
* Получение логического текстового описания введенного изображения;
* Возможность использовать ее в других системах;

## Обоснование критериев качества

Для разрабатываемой мультимодальной информационной системы основными критериями качества являются [1]:

* Простота использования;
* Точность определения ключевого объекта на изображении;
* Использование русского языка;
* Полнота описания композиции;
* Использование изображений любых размеров;
* Качество документации;

**Простота использования –** простой и интуитивно понятный интерфейс приложения должен упрощать использование системы. Так же важно не отягощать интерфейс, чтобы не было ничего лишнего.

**Точность определения ключевого объекта на изображении –** один из важнейших критериев, так как невозможно создать качественное логическое описание изображения, неверно определив ключевой объект на изображении. Так же к этому критерию относится неверное распознавание объекта, то есть ошибочное название ключевого объекта.

**Использование русского языка –** также один из важных критериев системы. Для простоты восприятия необходимо представлять итоговое логическое текстовое описание изображения в русском языке.

**Полнота описания композиции –** на изображении помимо ключевого объекта также присутствуют побочные, не менее важно верно определять и описывать объекты окружающие центральный, для полноты восприятия изображения.

**Использование изображений любых размеров –** возможность подавать на вход системы изображения любых размеров является достаточно важным критерием, так как предобработка изображения занимает время, а также не всегда есть возможность произвести изменения входного изображения. Следовательно, возможность системы работать с любыми изображениями – это большой плюс.

**Качество документации –** для простоты освоения системы важна документация, а точнее ее точность, понятность, полнота описания работы системы и наличие примеров эксплуатации.

## Сравнительный анализ аналогов

Определим для наших критериев весовые коэффициенты. Результаты приведены в таблице 1.

Таблица 1 – Критерии оценки систем

| **К** | **Название критерия** | **Весовой коэффициент** |
| --- | --- | --- |
| К1 | Простота использования | α |
| К2 | Точность определения ключевого объекта на изображении | 4α |
| К3 | Использование русского языка | 4α |
| К4 | Полнота описания композиции | 2α |
| К5 | Использование изображений любых размеров | 2α |
| К6 | Качество документации | α |

Определим величину α, рассчитав значения весовых коэффициентов по формуле 1.

(1)

где n – общее количество весовых коэффициентов;

– весовые коэффициенты;

Подставим значения весовых коэффициентов в формулу 1:

Теперь необходимо перевести качественные критерии в количественные по таблицам 2, 3, 4, 5, 6.

Таблица 2 – Вербально-числовая шкала для К1

| **Качественная оценка** | **Количественная оценка** |
| --- | --- |
| Просто использовать, интуитивно понятно | 1 |
| Средняя сложность использования, интуитивно понятно | 0,8 |
| Сложно использовать | 0,6 |

Таблица 3 – Вербально-числовая шкала для К3

| **Качественная оценка** | **Количественная оценка** |
| --- | --- |
| Русский язык используется | 1 |
| Русский язык не используется | 0 |

Таблица 4 – Вербально-числовая шкала для К4

| **Качественная оценка** | **Количественная оценка** |
| --- | --- |
| Полное описание побочных объектов окружения | 1 |
| Неполное описание побочных объектов | 0,8 |
| Побочные объекты на изображении почти не описываются | 0,6 |

Таблица 5 – Вербально-числовая шкала для К5

| **Качественная оценка** | **Количественная оценка** |
| --- | --- |
| Можно использовать изображения любых размеров | 1 |
| Нельзя использовать изображения любых размеров | 0 |

Таблица 6 – Вербально-числовая шкала для К6

| **Качественная оценка** | **Количественная оценка** |
| --- | --- |
| Качественная, полная, понятная документация с примерами | 1 |
| Качественная, полная, понятная документация без примеров | 0,9 |
| Хорошая, понятная документация, но неполная и без примеров | 0,8 |
| Плохая документация | 0,6 |

Определим варианты, с которыми будем сравнивать нашу систему. Выделим аналоги в таблице 7.

Таблица 7 – Аналоги для сравнения

| **В** | **Вариант** | **Описание** |
| --- | --- | --- |
| В1 | MIT | Разрабатываемая мультимодальная информационная система |
| В2 | STAIR Captions | Прямой аналог MIT |
| В3 | Enhanced image captioning via Azure Cognitive Services | Прямой аналог MIT |

Проведем сравнительный анализ аналогов по методу взвешенной суммы, используя формулу 2. Результат анализа представлен в таблице 8.

(2)

Таблица 8 – Сравнительный анализ аналогов по методу взвешенной суммы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Код критерия** | **Коэффициент важности локального критерия ()** | **Значения локальных критериев** | | |
| **В1** | **В2** | **В3** |
| K1 | 0,0714 | 1 | 1 | 0,6 |
| K2 | 0,2857 | 0,95 | 0,94 | 1 |
| K3 | 0,2857 | 1 | 0 | 0 |
| K4 | 0,1429 | 0,8 | 0,6 | 1 |
| K5 | 0,1429 | 1 | 0 | 1 |
| К6 | 0,0714 | 1 | 0,8 | 1 |
| **Итого** | **1** | **0,943235** | **0,48216** | **0,67146** |

Для нахождения лучшего вариант, воспользуемся формулой 3.

(3)

Вариант В1, то есть разрабатываемая мультимодальная информационная система, имеет наивысший коэффициент. На основании данного сравнительного анализа можно сделать вывод, что разрабатываемая система превосходит аналоги на данном наборе критериев, а значит, разработка данной системы оправдана.

# ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

## Исследование существующих архитектур

### Архитектура RCNN + BRNN

Архитектура, предназначенная для генерации описания изображений и их регионов на естественном языке. Этот подход использует наборы данных изображений и их описания, чтобы узнать об интермодальных соответствиях между языком и визуальными данными. Архитектура основана на новой комбинации сверточных нейронных сетей по областям изображения (RCNN), двунаправленных рекуррентных нейронных сетей (BRNN) по предложениям и структурированной цели, которая выравнивает две модальности через мультимодальное вложение. Архитектура мультимодальной рекуррентной нейронной сети использует предполагаемые выравнивания, чтобы научиться генерировать новые описания областей изображения. Данная архитектура демонстрирует современные результаты в экспериментах по поиску наборов данных Flickr8K, Flickr30K и MSCOCO. Сгенерированные описания значительно превосходят базовые показатели поиска как для полных изображений, так и для нового набора данных аннотаций на уровне региона [3].

Беглого взгляда на изображение достаточно, чтобы человек указал и описал огромное количество деталей визуальной сцены. Однако, эта замечательная способность оказалась труднодостижимой задачей для моделей визуального распознавания. Большая часть работ в области визуального распознавания была сосредоточена на маркировке изображений с помощью фиксированного набора визуальных категорий, и в этих усилиях был достигнут большой прогресс. Однако, хотя закрытые словари визуальных концепций представляют собой удобное допущение для моделирования, они значительно ограничивают по сравнению с огромным количеством подробных описаний, которые может составить человек. Были разработаны некоторые новаторские подходы, которые решают проблему создания описаний изображений. Однако эти модели часто полагаются на жестко запрограммированные визуальные концепции и шаблоны предложений, что накладывает ограничения на их разнообразие. Основная задача создания этой архитектуры заключается в разработке модели, достаточно богатой, чтобы одновременно рассуждать о содержании изображений и их представлении в области естественного языка. Кроме того, модель не должна содержать предположений о конкретных жестко запрограммированных шаблонах, правилах или категориях и вместо этого полагаться на обучение на основе данных обучения. Практическая проблема заключается в том, что наборы данных с подписями к изображениям доступны в большом количестве в интернете, но эти описания мультиплексируют упоминания нескольких объектов, расположение которых на изображениях неизвестно. Основная идея заключается в том, чтобы использовать эти большие наборы данных предложений изображений, рассматривая предложения как слабые метки, в которых смежные сегменты слов соответствуют некоторому конкретному, но неизвестному месту на изображении. Подход состоит в том, чтобы вывести эти совпадения и использовать их для изучения генеративной модели описаний.

Эта архитектура подразумевает модель глубокой нейронной сети, которая делает вывод о скрытом выравнивании между сегментами предложений и областью изображения, которую они описывают. А также, связывает две модальности через общее мультимодальное пространство вложения и структурированную цель.

Одной из основных целей рассматриваемой архитектуры является плотное аннотирование содержимого изображений. Основное внимание уделяется более полному и высокоуровневому описанию регионов.

Задача описания изображений с предложениями. Ряд подходов ставит задачу как проблему поиска, когда наиболее совместимая аннотация в обучающем наборе переносится на тестовое изображение, или, когда обучающие аннотации разбиваются и сшиваются вместе. Некоторые подходы генерируют подписи к изображениям на основе фиксированных шаблонов, которые заполняются на основе содержимого изображения или генеративных грамматик, но этот подход ограничивает разнообразие возможных выходов. Данная архитектура рекуррентной нейронной сети (RNN) обуславливает распределение вероятностей для следующего слова в предложении для всех ранее сгенерированных слов. Она выравнивает смежные сегменты предложений, которые более значимы, интерпретируемы и не имеют фиксированной длины. Что касается изображений, то сверточные нейронные сети (CNN) являются мощным классом моделей для классификации изображений и обнаружения объектов. Что касается предложений, эта архитектура задействует преимущества предварительно обученных векторов слов для получения низко размерных представлений слов.

Во время обучения входными данными нашей модели являются набор изображений и соответствующих им описаний предложений. Сначала идет модель, которая выравнивает фрагменты предложений с визуальными областями, которые они описывают, посредством мультимодального встраивания. Затем эти соответствия обрабатываются, как обучающие данные для второй мультимодальной модели рекуррентной нейронной сети, которая учится генерировать фрагменты.

Ключевой особенностью данной архитектуры является то, что предложения, написанные людьми, часто ссылаются на какое-то конкретное, но неизвестное место на изображении, аналогично и здесь, формируются скрытые соответствия, с конечной целью научиться позже генерировать эти фрагменты из областей изображений. Здесь применяется двунаправленная рекуррентная нейронная сеть для вычисления представлений слов в предложении, что обеспечивает избавление от необходимости вычислять деревья зависимостей и неограниченное взаимодействие слов и их контекста в предложении.

Для обнаружения объектов на каждом изображении используется региональная сверточная нейронная сеть (RCNN), которая предварительно обучена на ImageNet и точно настроена на 200 классах ImageNet Detection Challenge. Используется 19 лучших обнаруженных местоположений в дополнение ко всему изображению и вычисляется представления на основе пикселей внутри каждого ограничивающего прямоугольника. Параметры CNN содержат примерно 60 миллионов параметров. RCNN – вариант сверточной нейронной сети, который отличается тем, что разбивает изображение на регионы и создает региональные признаки.

Чтобы установить интермодальные отношения, слова в предложении представляются в том же трехмерном пространстве вложения, которое занимают области изображения. Для этого, самый простой подход - проецировать каждое отдельное слово прямо на это вложение. Однако этот подход не учитывает в предложении информацию о порядке и контексте слов. Расширением этой идеи является использование биграмм слов. Однако это по-прежнему требует произвольного максимального размера контекстного окна и требует использования анализаторов дерева зависимостей, которые могут быть обучены на несвязанных текстовых корпусах. Для решения этой проблемы, в архитектуре используется двунаправленная рекуррентная нейронная сеть (BRNN) для вычисления представлений слов. BRNN принимает последовательность из N слов (закодированных в представлении 1 из k) и преобразует каждое из них в h-мерный вектор. Также, представление каждого слова обогащается контекстом переменного размера вокруг этого слова. BRNN состоит из двух независимых потоков обработки, один из которых движется слева направо, а другой справа налево. По факту BRNN – объединение двух независимых RNN. Входная последовательность подается в обычном порядке времени для одной сети и в обратном порядке времени для другой. Выходы двух сетей обычно объединяются или суммируются на каждом временном шаге.

В архитектуре используются преобразования, которые отображают каждое изображение и предложение в набор векторов в общем трехмерном пространстве. Поскольку контроль осуществляется на уровне целых изображений и предложений, стратегия состоит в том, чтобы сформулировать оценку предложения изображения как функцию от оценок отдельных региональных слов. Интуитивно понятно, что пара предложение-изображение должна иметь высокий балл соответствия, если ее слова имеют уверенную поддержку в изображении.

В конце архитектуры располагается мультимодальная рекуррентная нейронная сеть для генерации описаний. В других разработанных языковых моделях, основанных на рекуррентных нейронных сетях (RNN), вывод описаний переменной длины заключается в определении распределения вероятностей следующего слова в последовательности с учетом текущего слова и контекста из предыдущих временных шагов. В архитектуре рассматривается достаточно эффективное расширение, которое дополнительно обуславливает процесс генерации содержимым входного изображения. Фактически, во время обучения Мультимодальная RNN принимает пиксели изображения и последовательность входных векторов. Затем вычисляется последовательность скрытых состояний и последовательность выходных данных.

BRNN обучается комбинировать слово с предыдущим контекстом для предсказания следующего слова. Прогнозы RNN основываются на информации об изображении через взаимодействия смещения на первом этапе. Условно обучение заключается в том, что каждый раз в качестве вектора слов в RNN подается слово, которое было сгенерировано предыдущим и ожидается предсказывания следующего слова описания. Первый вектор слов начинается с маркера START, а в качестве вывода последнего вектора слов выступает маркер END.

В данной архитектуре применяется SGD с мини-пакетами из 100 пар изображение-предложение, чтобы оптимизировать модель выравнивания. Происходит проверка скорости обучения и снижения веса. Присутствует регуляризация выпадения во всех слоях, кроме повторяющихся слоев, и отсекаем градиенты поэлементно на 5, что важно. Генеративную RNN труднее оптимизировать из-за несоответствия частоты слов между редкими и общими словами.

Итоговая архитектуру представляет из себя модель, которая генерирует описания областей изображений на естественном языке на основе слабых меток в форме набора данных изображений и предложений, с очень небольшим количеством жестко запрограммированных предположений. Также, архитектура включает новую модель ранжирования, которая согласовывает части визуальных и языковых модальностей посредством общего мультимодального встраивания. Было продемонстрировано, что эта модель обеспечивает высокую производительность на эксперименты по ранжированию изображений и предложений. Так же была описана архитектура мультимодальной рекуррентной нейронной сети, которая генерирует описания визуальных данных.

### Архитектура CNN + CNN

Создание описаний к изображениям - сложная задача, которая сочетает в себе область компьютерного зрения и обработку естественного языка. Было предложено множество подходов для достижения цели автоматического описания изображения, и в этой области доминируют модели на основе рекуррентной нейронной сети (RNN) или долгосрочной краткосрочной памяти (LSTM). Однако RNN или LSTM нельзя вычислять параллельно и игнорировать лежащую в основе иерархическую структуру предложения [2].

Эта архитектура предлагает структуру, которая использует только сверточные нейронные сети (CNN) для генерации описаний. Благодаря параллельным вычислениям, базовая модель примерно в 3 раза быстрее, чем NIC (модель на основе LSTM) во время обучения, а также обеспечивает лучшие результаты. По сравнению с моделями на основе LSTM, которые применяют аналогичные механизмы внимания, предложенные здесь модели, достигают сопоставимых баллов BLEU-1,2,3,4 и METEOR и более высоких баллов CIDEr. Эта архитектура также тестируется на наборе данных аннотаций абзацев и получает более высокий балл CIDEr по сравнению с иерархическими LSTM.

Для достижения цели, описания изображений, требуются по крайней мере три модели:

* модель зрения - для извлечения визуальных характеристик из изображений;
* языковая модель - для создания подписей;
* связь между зрением и языковыми моделями. ;

Подписи к изображениям объединяют две области - компьютерное зрение и обработку естественного языка (NLP), чтобы решить проблему понимания как изображений, так и их описаний. Почти все предлагаемые в настоящее время архитектуры находятся в рамках CNN + RNN, в которой CNN используется для модели зрения, а RNN используется для генерации предложений. Более того, существуют разные способы подключения CNN и RNN. Наивный способ - это прямая передача выходных данных CNN в RNN. Однако этот наивный подход обрабатывает объекты на изображении одинаково и игнорирует заметные объекты при генерации одного слова. Чтобы имитировать зрительный механизм внимания человека, в каркас CNN + RNN были введены модули внимания. Хотя структура CNN+RNN популярна и обеспечивает удовлетворительные результаты, у последовательных RNN есть недостатки:

* RNN должны вычисляться пошагово, что не поддается параллельным вычислениям во время обучения;
* Существует длинный путь между началом и концом предложения с использованием RNN, который легко забывает информацию дальнего действия. Древовидные структуры могут сократить путь между начальным и конечным словами, но деревья требуют специальной обработки, которую нелегко распараллелить. Также деревья определяются с помощью NLP (например, именная фраза, прилагательные, глагольная фраза и т. Д.), которая представляет собой созданную вручную структуру, которая может не быть оптимальной для задачи создания описаний.

Альтернативой языковой модели RNN и деревьев являются CNN, применяемые к предложению для слияния слов слой за слоем, таким образом изучая древовидную структуру предложения. CNN могут быть реализованы параллельно и иметь больший размер воспринимающего поля (может видеть больше слов) с использованием меньшего количества слоев. Например, для предложения, состоящего из 10 слов, RNN необходимо выполнить итерацию 10 раз, чтобы получить представление предложения (эквивалентно 10 слоям), в то время как CNN с размером ядра 3 требует только 5 слоев. Используя более широкие ядра, CNN могут обрабатывать более длинные предложения с меньшим количеством слоев. CNN широко применялись в области NLP, что привело к повышению производительности во многих задачах, таких как языковое моделирование, машинный перевод и классификация текстов. Основываясь на приложениях CNN в области NLP, разработана данная архитектура, в которой CNN используются только для субтитров к изображениям. Основные моменты этой архитектуры:

* В архитектуре используется структура CNN+CNN для описаний к изображениям, которая быстрее, чем модели на основе LSTM, и превосходит их по некоторым показателям;
* Архитектура включает в себя модуль иерархического внимания, чтобы связать видение CNN с языком CNN, что значительно улучшает производительность;
* В архитектуре рассматривается влияние гиперпараметров, включая количество слоев и ширину ядра языка CNN. Восприимчивое поле языка CNN может быть увеличено путем наложения большего количества слоев или увеличения ширины ядра, и, как показывают эксперименты, увеличение ширины ядра лучше.

Было предложено множество методов для автоматической генерации подписей к изображениям, и наиболее популярной структурой является CNN + RNN. Модель m-RNN использует обычную RNN в сочетании с различными CNN, скрытые состояния RNN и выходные данные CNN передаются в мультимодальный блок для объединения функций изображения и языка на каждом временном шаге, а слой softmax предсказывает следующее слово. Однако RNN страдают от проблемы исчезающего градиента. Следовательно, модель описания нейронных изображений (NIC) использует LSTM в качестве декодера. Вначале модель NIC принимает вектор признаков изображения в качестве входных данных, а затем визуальная информация передается по повторяющемуся пути. И в m-RNN, и в NIC изображение представлено одним вектором, который игнорирует различные области и объекты на изображении. Механизм пространственного внимания позволяет модели обращать внимание на разные области на каждом временном шаге. В модели семантического внимания применяется полностью сверточная сеть (FCN) для обнаружения семантики, а затем вычисляется вес для каждой семантики на каждом временном шаге. Хотя это семантическое внимание значительно улучшает производительность, в некоторой степени оно имеет проблему накопления ошибок предсказания по сгенерированным последовательностям. В качестве альтернативы использует семантику по-другому, генерируя параметры RNN или LSTM из выходных данных семантического детектора. Эту модель можно обучать сквозным образом. LSTM также использовались иерархически для моделирования неглубокой древовидной структуры. Во фразовой модели LSTM, имеется два уровня LSTM: один для моделирования предложения, состоящего из фраз, а другой для генерации слов во фразе. Иерархические LSTM также использовались для генерации описания абзаца, состоящего из нескольких предложений. В данной архитектур впервые применяется CNN для полной замены RNN в плане создания описаний изображений. В архитектуре предлагается модуль иерархического внимания, цель которого - изучить взаимосвязь между концепциями на каждом уровне и областями изображения. Кроме того, этот модуль внимания использует скалярное произведение, которое имеет меньше параметров и может быть вычислен достаточно быстро.

Методы глубокого обучения доминируют в области NLP, и CNN являются важным инструментом для решения проблем в этой области. В некоторых архитектурах CNN применяются к нескольким задачам NLP, таким как разбиение на фрагменты, тегирование части речи, распознавание именованных сущностей и маркировка семантических ролей. Эта основанная на CNN модель обеспечивает точные результаты с высокой скоростью и способна изучать представления вместо использования созданных вручную функций. Некоторые модели используют очень глубокие CNN для классификации текста, и результаты показывают, что глубина и максимальное объединение улучшают производительность. В рассматриваемой архитектуре предлагается использование языка CNN без объединения слоев. Кроме того, используется каузальную свертку, чтобы наша модель могла генерировать подписи пословно.

CNN демонстрируют относительно сильную способность работать с очень длинными последовательностями. В модели предлагается фрейм CNN+CNN для описаний к изображениям. В этой архитектуре есть четыре модуля:

* Модуль зрения, который приспособлен для «просмотра» изображений;
* Языковой модуль, предназначенный для моделирования предложений;
* Модуль внимания, который соединяет модуль зрения с языковым модулем;
* Модуль предсказания, который берет визуальные характеристики из модуля внимания и концепции из языкового модуля в качестве входных данных и предсказывает следующее слово;

Модуль технического зрения представляет собой CNN без полностью подключенного слоя, вывод которого представляет собой карту характеристик d × d × Dc. В каждой позиции карты признаков размерный вектор признаков Dc представляет часть изображения. В этой архитектуре используется VGG-16 в качестве CNN для визуального модуля.

Языковая модель основана на CNN без объединения, что сильно отличается от типичной структуры на основе RNN. RNN используют повторяющийся путь для запоминания контекста, в то время как CNN использует ядра и складывает несколько слоев для моделирования контекста. Сначала используется справочная таблица для проецирования каждого слова в пространство вложения с размерами De и вычисляются вложения E = [e1, …, eL], где ej ∈ R De. На этом этапе предложение представлено матрицей L × De. Пакет сверточных слоев со стробированными линейными блоками (GLU) следует за слоем внедрения. Фильтры свертки являются причинными фильтрами, зависящими только от текущих и прошлых входов. Во время вывода эта структура позволяет сгенерировать предложение с использованием процесса прямой связи, в котором предсказанное слово на выходном слое используется в качестве следующего входного слова.

Для моделирования предложений длина вывода должна быть такой же, как и длина предложения ввода. Поскольку нет объединяющих слоев и полностью связанных слоев, в начало входного и скрытого слоя добавлено нулевое заполнение. Если ширина сверточного ядра равна k, что указывает на то, что на каждом этапе учитывается k концептов, перед сверткой матрицы слов/концепций должны быть дополнены нулями с помощью k-1 нулевых векторов. Выходом CNN является набор концептов c = [c1, c2, …, cL], где cj ∈ R De.

Чтобы предсказывать разные слова, нужно следить за разными объектами на изображениях и вводить их в модуль предсказания. Модуль внимания принимает визуальные признаки V и концепты C на верхнем уровне в качестве входных данных. Для каждого концепта cj и вектора визуальных признаков vi вычисляется оценка. Следовательно, каждому понятию cj соответствует вектор оценок sj = [s1, j, s2, j, · · ·, sN, j], указывающий совпадения с характеристиками изображения. Затем sj подается в слой softmax, который присваивает вес wi, j вектору визуальных признаков vi. Далее используется операция взвешенной суммы для вычисления окончательного вектора внимания следующим образом. Результатом работы модуля внимания являются признаки внимания a = [a1, a2, …, aL], где ai ∈ R Dc, соответствующие каждому слову в предложении.

Модуль прогнозирования принимает на вход функции внимания A и концепции C. Модуль прогнозирования представляет собой нейронную сеть с одним скрытым слоем. В позиции j предложения в сеть подаются aj и cj, на выходе - вероятность предсказания следующего слова Pj + 1. Во время обучения дается пара входных изображений и предложений, поэтому модуль прогнозирования может быть реализован как сверточное ядро ​​1 × 1, что быстрее, чем модели на основе RNN.

Базовая модель CNN+CNN в предыдущем подразделе извлекает признаки внимания из верхнего уровня языковой модели. Однако разные уровни языка CNN представляют разные уровни концепций, которые могут выиграть от визуального ввода. Следовательно, в архитектуре также присутствует модуль иерархического внимания (hier-att), где векторы внимания вычисляются на каждом уровне языковой модели и передаются на следующий уровень.

В отличие от моделей на основе RNN, которые вычисляют карты внимания слева направо (слово за словом), в данной архитектуре карты внимания рассчитываются снизу-вверх (слой за слоем), чтобы предотвратить боковые связи в том же слое. Это позволяет обучать модель параллельно всем словам в предложении, а не дословно. Стоит обратить внимание, что модель по-прежнему видит сопутствующие элементы предыдущих слов через слой каузальной свертки.

Во время обучения, когда задаются пары изображение-заголовок, структуры свертки применяются обычным образом, а функция потерь для каждого предложения представляет собой кросс-энтропию. Во время вывода заголовок создается для изображения с использованием процесса прямой связи. Заголовок инициализируется как нулевое заполнение и начальный токен <S> и подается в качестве входного предложения в модель для прогнозирования вероятности следующего слова. Прогнозируемое слово добавляется к заголовку, и процесс повторяется до тех пор, пока не будет предсказан конечный токен </S> или не будет достигнута максимальная длина. Предсказанные слова выбираются с использованием наивного жадного метода (наиболее вероятное слово), который эквивалентен алгоритму лучевого поиска с шириной луча 1.

В данной архитектуре была реализована структура CNN+CNN для создания подписей к изображениям. Также были исследованы влияние ширины ядра и глубины слоя языка CNN. Эта архитектура продемонстрировала, что модели на основе CNN конкурентоспособны по сравнению с моделями на основе LSTM, но их можно обучить быстрее. В архитектуре реализована визуализация выученных карт внимания, чтобы показать, что модель способна усваивать концепции и осмысленно обращать внимание на соответствующие области на изображениях.

### Архитектура CNN-RNN

Архитектура CNN-RNN все шире применяется в различных задачах описания изображений, включая классификацию с несколькими метками и создание субтитров. Существующие модели используют слабо семантический скрытый слой CNN или его преобразование в качестве встраивания изображения, которое обеспечивает интерфейс между CNN и RNN. Это оставляет RNN чрезмерно загруженной двумя задачами: прогнозированием визуальных концепций и моделированием их корреляций для создания структурированных выходных аннотаций. Важно отметить, что это делает сквозное обучение CNN и RNN медленным и неэффективным из-за сложности обратного распространения градиентов через RNN для обучения CNN. В этой архитектуре реализована простая модификация шаблона проектирования, которая делает обучение более эффективным и действенным. В частности, использование семантически регуляризованного уровня внедрения в качестве интерфейса между CNN и RNN. Упорядочение интерфейса может частично или полностью разделить проблемы обучения, позволяя каждому обучаться более эффективно, а совместное обучение - гораздо эффективнее. Обширные эксперименты показывают, что максимальная производительность достигается при классификации нескольких ярлыков, а также при добавлении подписей к изображениям [4].

В данной архитектуре применяется структура проектирования кодер-декодер сверточная нейронная сеть - рекуррентная нейронная сеть (CNN-RNN) для решения задачи предсказания структурированной метки как при классификации нескольких меток, так и в описаниях изображений. CNN используется для кодирования изображения в вектор фиксированной длины, который затем передается в RNN, который либо декодирует его в список тегов (с несколькими метками), либо в последовательность слов, составляющих предложение (описания). С этой архитектурой кодера-декодера CNN и RNN могут быть обучены от начала до конца, вводя изображение и выводя упорядоченный список меток. У всех архитектур, основанных на этой структуре, есть общая ключевая характеристика: встраивание изображений, обеспечивающее интерфейс CNN-RNN, является последним функциональным слоем CNN.

Использование таких слоев в качестве входных данных для RNN имеет ряд неблагоприятных последствий для обучения сквозной модели аннотаций повторяющихся изображений. Во-первых, поскольку функция вывода CNN не имеет явного семантического смысла, задачи прогнозирования меток и моделирования корреляции, а также грамматики меток теперь должны выполняться только моделью RNN. Это усугубляет и без того сложную задачу обучения RNN, поскольку количество визуальных концепций слов часто велико и их корреляция обширна. Во-вторых, модель подключенной CNN-RNN является достаточно глубокой, учитывая развертывание RNN. Существующие модели CNN-RNN применяют контроль только на конечном выходе RNN и распространяют контроль обратно на более ранние уровни, CNN. Это приводит к трудностям обучения в виде «исчезающих» градиентов. Кроме того, совместное обучение CNN и RNN должно проводиться очень осторожно, чтобы предотвратить искажение модели CNN зашумленными градиентами, распространяющимися обратно от RNN. В результате сходимость модели часто происходит очень медленно.

В этой архитектуре предлагается изменить уровень встраивания изображений и ввести семантическую регуляризацию в модель CNN-RNN, чтобы получить значительно более точные результаты и сделать обучение модели более стабильным и быстрым. В частности, выполняется многозадачное обучение, в котором вспомогательной задачей является регуляризация уровня встраивания изображения, интерфейса для кодирования семантически значимых визуальных концепций, которые напрямую связаны с задачей прогнозирования меток. Это можно понять с нескольких точек зрения:

* Как разделение системы на модель для генерации унарных потенциалов (CNN) путем индивидуального прогнозирования метки и моделирования их отношений (RNN) для структурированного прогнозирования. Поскольку унарная CNN берет на себя ответственность за предсказание концепций, реляционная модель RNN может лучше сосредоточиться на изучении корреляций концепций и генерации предложений. В случае классификации с несколькими метками, где пространство меток семантической регуляризации и выходное пространство RNN одинаковы, это можно рассматривать как аналог CRF-декодирования совместного распределения;
* Как сеть с глубоким надзором, обеспечивающая вспомогательный надзор за серединой того, что по сути является очень глубокой сетью. Такой глубокий контроль улучшает точность и скорость сходимости. В данном случае, в частности, это в значительной степени устраняет проблему зашумленных градиентов RNN, распространяющихся в обратном направлении, чтобы повредить кодировщик CNN. Таким образом, это обеспечивает лучшую и более эффективную настройку модуля CNN, а также быструю сходимость в сквозном обучении полной модели CNN-RNN;
* В соответствии с моделью кодер-декодер с априорной предвзятостью предпочтения семантически значимых кодов;

Эта архитектура имеет следующие особенности:

* Предлагается новая модель аннотации изображений CNN-RNN, которая отличается от существующих моделей выбором слоя встраивания изображения и введением глубоко контролируемой семантической регуляризации в слой встраивания;
* Предлагаемая семантическая регуляризация обеспечивает надежную тонкую настройку кодировщика изображений CNN, а также быструю сходимость сквозного обучения CNN-RNN;
* С помощью обширных экспериментов демонстрируется, что как при классификации с несколькими ярлыками, так и при использовании аннотаций к изображениям модель достигает высочайшего уровня производительности;

Подпись к изображениям на основе нейронной сети. В ряде недавних исследований с использованием аннотаций используется восходящий подход, когда сначала обнаруживаются слова или фразы, а затем составляются предложения с использованием языковой модели. По сравнению с этой моделью данная архитектура представляет собой сквозную модель CNN-RNN, которая совместно изучает модули кодирования изображений и языкового декодирования.

Модели аннотаций к изображениям на основе CNN-RNN стали популярными. В этой архитектуре используются семантические концепции для регуляризации представления уровня интерфейса CNN-RNN, что приводит к значительному повышению производительности и гораздо более простому обучению модели.

Модель CNN-RNN состоит из двух частей:

* Визуальный кодировщик воспринимает визуальное содержание изображения и кодирует его для встраивания изображения;
* Декодер принимает вложение в качестве входных данных и генерирует последовательности меток (слов).

Для данного изображения I визуальный кодировщик закодирует его в вектор фиксированной длины Ie ∈ R d × 1, называемый признаками изображения. В данной архитектуре это представлено семантическим представлением для лучшего взаимодействия с RNN.

Затем декодер RNN примет Ie в качестве условия и сгенерирует прогнозируемый путь π = (a1, a2, ..., an). Путь представляет собой упорядоченную последовательность, поэтому при классификации с несколькими метками необходимо определить приоритет меток, чтобы преобразовать метки в последовательность. Используется редкий первый порядок, чтобы придать редким классам большее значение во время прогнозирования, тем самым противодействуя проблеме дисбаланса меток.

Было рассмотрено множество различных CNN для кодировщика, но для RNN-декодера модель долговременной краткосрочной памяти (LSTM) была выбрана почти всеми существующими моделями. Это связано с тем, что он управляет передачей сообщений между временными шагами с помощью ворот, чтобы облегчить проблему исчезающего, увеличивающегося градиента, которая мешала обучению предыдущих моделей RNN.

Чтобы сгенерировать обусловленные изображением последовательности, декодер должен использовать преимущества признаков изображения и существующие модели достигают этого несколькими способами.

Но несмотря на это, существующие модели CNN-RNN имеют ключевую общую характеристику, а именно признаки изображения, которые действуют как интерфейс между моделями CNN и RNN, считается слоем слабой и неявной семантики. Это означает, что RNN должна одновременно учиться предсказывать семантические концепции на основе предоставленных функций, а также моделировать корреляцию этих концепций. Научиться предсказывать концепции сложнее для RNN, потому что градиенты распространяются обратно от относительно «далекого» надзора, RNN выводит на будущих временных шагах. Более того, точная настройка CNN становится сложной, потому что шумные градиенты, распространяющиеся от RNN, могут легко ухудшить, а не улучшить производительность.

Для снижения нагрузки на RNN, в данной архитектуре предлагается разделить эти две задачи, а именно: изучение семантических концепций и реляционное моделирование. В частности, изучение семантических концепций здесь выполняется унарной моделью CNN, которая принимает в качестве входных изображений и связанную с ними дополнительную информацию, если таковая имеется, и производит вероятностную оценку семантических концепций. Реляционное моделирование обрабатывается моделью RNN, которая принимает оценки вероятности концепции и моделирует их корреляции для генерации последовательностей меток, слов. Поскольку выбранное встраивание обучается под непосредственным контролем достоверных меток и визуальных концепций, оно имеет четкое семантическое значение: каждая единица соответствует семантическому понятию.

В семантически регуляризованной CNN-RNN (S-CNN-RNN) часть CNN принимает изображение I в качестве входных данных и прогнозирует вероятность семантических концепций sˆ ∈ R k × 1, где k - это количество семантических понятий. Модель RNN принимает sˆ в качестве входных данных и генерирует последовательности π. Ключевым моментом является то, что теперь надзор можно добавить, как на выходной уровень RNN, так и на уровень внедрения sˆ. Это приводит к двум потерям: потере предсказания концепции и потере реляционного моделирования.

Внедрение семантической регуляризации в середине CNN-RNN позволяет более эффективно обучать модели. Это упрощает двухэтапную стратегию обучения. На первом этапе предварительно обучается модель CNN и модель RNN параллельно, а на втором этапе они настраиваются вместе.

Для предварительного обучения модели CNN основные семантические концепции si используются в качестве цели обучения в стандартной кросс-энтропийной потере для k визуальных концепций.

Для предварительного обучения LSTM вход концепции sˆi сначала подключается к полносвязному (FC) слою, а затем используется для установки начального скрытого состояния LSTM. Модель LSTM учится максимизировать вероятность генерации целевых последовательностей, обусловленных семантическим входом, а потеря Lr (π, π ∗ | sˆ) представляет собой просто сумму отрицательного логарифмического правдоподобия по всем временным шагам. Подавая s, а не sˆ, LSTM можно предварительно обучить независимо от CNN.

После того, как модели CNN и RNN предварительно обучены, вся модель может быть совместно обучена путем одновременной оптимизации тщательно контролируемых совместных потерь L. Для вывода обуславливается изображение, устанавливая начальное состояние, затем подается сигнал запуска и периодически выполняется выборка предсказаний модели предыдущего шага в качестве входных данных до тех пор, пока не будет сгенерирован сигнал завершения. Для классификации с несколькими метками просто берется максимальный выход модели, в то время как лучевой поиск с шириной три используется для подписи изображений.

Чтобы применить архитектуру CNN-RNN к классификации с несколькими метками, сначала ранжируются обучающие метки в соответствии с их частотой в обучающем наборе и генерируется упорядоченный список меток, в первую очередь с редкими метками. Затем комбинируется модель изображения с предварительно обученной моделью тега, суммируя их прогнозы, и обучаются вместе с перекрестной потерей энтропии.

Архитектура представляет из себя семантически регуляризованную модель CNN-RNN для аннотации изображений. Семантическая регуляризация делает интерфейс CNN-RNN семантически значимым, распределяет задачи прогнозирования меток и корреляции между моделями CNN и RNN, и, что важно, глубокий контроль делает обучение всей модели более стабильным и эффективным. Обширные оценки NUS-WIDE и MSCOCO демонстрируют эффективность данной архитектуры, как в отношении классификации нескольких меток, так и в отношении описания изображений.

## Сравнение вариантов архитектур

Для разработки мультимодальной информационной системы требуется выбрать подходящую основу будущей архитектуры.

Для этого рассмотрим варианты описанные раннее. В таблице 9 представлены варианты архитектур, участвующие в сравнении.

Таблица 9 – Сравниваемые архитектуры

| **В** | **Вариант архитектуры** |
| --- | --- |
| В1 | RCNN-BRNN |
| В2 | CNN-CNN |
| В3 | CNN-RNN |

Сравнивать эти варианты мы будем по набору критериев, представленных в таблице 10.

Таблица 10 – Критерии сравнения архитектур

| **К** | **Название критерия** | **Описание** |
| --- | --- | --- |
| К1 | Meteor | Метрика для оценки результатов машинного перевода. Показатель основан на гармоническом среднем значении точности и отзыва униграмм, причем отзыв имеет больший вес, чем точность. |
| К2 | CIDEr | Критерий оценки описания изображений основанный на человеческом консенсусе. |
| К3 | BLEU-1 | 1-граммовый балл BLEU, оценка для сравнения перевода текста или для оценки сгенерированного текста. |
| К4 | BLEU-2 | 2-граммовый балл BLEU, оценка для сравнения перевода текста или для оценки сгенерированного текста. |
| К5 | BLEU-3 | 3-граммовый балл BLEU, оценка для сравнения перевода текста или для оценки сгенерированного текста. |
| К6 | BLEU-4 | 4-граммовый балл BLEU, оценка для сравнения перевода текста или для оценки сгенерированного текста. |

Определим весовые коэффициенты критериев методом базового критерия.

Группы важности [1]:

* Первая группа (наименее значимые) – К3, К4
* Вторая группа (более значимые) – К1, К5
* Третья группа (самые значимые) – К2, К6

Количество групп = 3

k1=1, k2=2, k3=4 – коэффициенты групп важности.

Вычислим веса критериев, подставив значения в формулу 1.

Итоговые весовые коэффициенты критериев представлены в таблице 11.

Таблица 11 – Весовые коэффициенты критериев

| **К** | **Название критерия** | **Весовые коэффициенты критерия** |
| --- | --- | --- |
| К1 | Meteor | 0,143 |
| К2 | CIDEr | 0,286 |
| К3 | BLEU-1 | 0,071 |
| К4 | BLEU-2 | 0,071 |
| К5 | BLEU-3 | 0,143 |
| К6 | BLEU-4 | 0,286 |

Итоговые нормированные значения критериев представлены в таблице 12.

Таблица 12 – Итоговые нормированные значения критериев

| **Вариант** | **Весовой коэффициент** | **В1** | **В2** | **В3** |
| --- | --- | --- | --- | --- |
| **К1** | 0,143 | 0,76 | 0,92 | 1 |
| **К2** | 0,286 | 0,67 | 0,85 | 1 |
| **К3** | 0,071 | 0,84 | 0,93 | 1 |
| **К4** | 0,071 | 0,78 | 0,89 | 1 |
| **К5** | 0,143 | 0,74 | 0,85 | 1 |
| **К6** | 0,286 | 0,71 | 0,82 | 1 |

Проведем сравнение методом близости к идеалу в таблице 13 [1].

Таблица 13 – Значения локальных критериев методом близости к идеалу

| **Вариант** | **Весовой коэффициент** | **В1** | **В2** | **В3** |
| --- | --- | --- | --- | --- |
| **К1** | 0,143 | 0,0576 | 0,0064 | 0 |
| **К2** | 0,286 | 0,1089 | 0,0225 | 0 |
| **К3** | 0,071 | 0,0256 | 0,0049 | 0 |
| **К4** | 0,071 | 0,0484 | 0,0121 | 0 |
| **К5** | 0,143 | 0,0676 | 0,0225 | 0 |
| **К6** | 0,286 | 0,0841 | 0,0324 | 0 |
| **Итого** | 1 | 0,28 | 0,145 | 0 |

Основываясь на сравнении методом близости к идеалу можем сделать вывод, что вариант архитектуры В3 является наилучшим.

Для того чтобы убедиться в том, что этот вариант архитектуры является наиболее предпочтительным, проведем еще одно сравнение архитектур, методом ранжирования Борда [1].

Для нахождения лучшего вариант по методу ранжирования Борда необходимо проставить каждому варианту ранг по сравнению с другими вариантами в рамках одного критерия по формуле 4:

(4)

Анализ представлен в таблице 14.

Таблица 14 – Ранжирование вариантов методом Борда

| **Вариант** | **В1** | **В2** | **В3** |
| --- | --- | --- | --- |
| **К1** | 3 | 2 | 1 |
| **К2** | 3 | 2 | 1 |
| **К3** | 3 | 1,5 | 1,5 |
| **К4** | 3 | 2 | 1 |
| **К5** | 3 | 2 | 1 |
| **К6** | 3 | 2 | 1 |
| **Итого** | 18 | 11,5 | 6,5 |

Для нахождения наилучшего вариант используем формулу 5:

(5)

По методу ранжирования Борда наилучшим вариантом получился В3.

Таким образом, наиболее предпочтительным, на основании 2 методов, оказался вариант архитектуры В3, то есть CNN-RNN. Следовательно, эта архитектура и послужит основанием для разработки.

# КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## Разработка архитектуры системы

В качестве основы архитектуры мультимодальной информационной системы был выбран концепт CNN – RNN [4].

CNN–RNN – архитектура состоящая из двух основных модулей нейронных сетей, для обработки различных типов данных. CNN для обработки графических данных, а RNN для работы с текстовыми данными, а точнее для генерации текстового логического описания.

Суть архитектуры заключается в том, что она разбивается на две составляющие декодер и энкодер. Энкодер – блок, который кодирует входящую информацию, в данном случае это блок CNN [7]. Декодер – блок, который декодирует информацию для её дальнейшего использования и генерации текста, в данном случае это блок RNN [6].

Работа архитектуры заключается в том, что во входной поток данных подаётся изображение, к которому необходимо сгенерировать логическое описание, для начала изображение преобразуется до необходимого размера, после чего это изображение подаётся на вход обученной модели CNN, которая обрабатывает изображение и формирует на его основе матрицу свойств, которая в свою очередь ужимается до вектора свойств полносвязного слоя. Финальный вектор свойств должен по своему размеру равняться входному вектору первого слоя декодера LSTM [5]. Вектор свойств проходит токенизацию, то есть присвоение тем или свойствам идентификаторов, после чего направляется на вход первого слоя LSTM. В декодере вектор свойств преобразуется в слова, на основе присвоенных идентификаторов свойств, сопоставляющихся со словарем. Таким образом генерируется описание изображения. В дальнейшем это описание из блока декодера поступает на модуль перевода на основе Google Translate API.

На рисунке 2 представлена архитектура мультимодальной информационной системы.

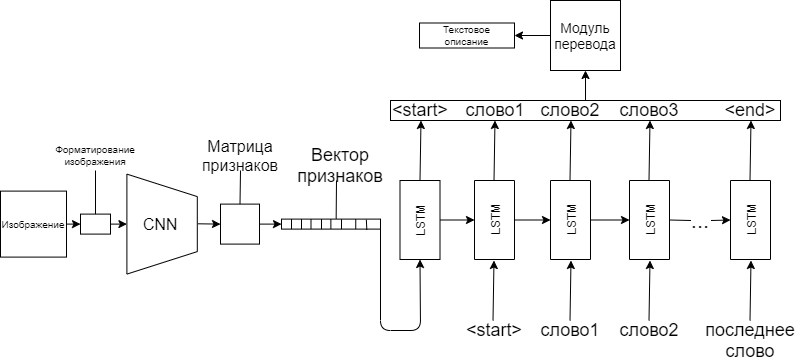


Рисунок 2 – Архитектура мультимодальной информационной системы

## Используемые при разработке технологии

В разработке модели мультимодального обучения использовались стандартные технологии. Модель была написана на языке Python 3.

Python – это высокоуровневый объектно-ориентированный язык программирования, широко использующийся во многих областях, в том числе в машинном обучении. Для разработки нейронных сетей на Python имеется огромное количество различных библиотек и фреймворков. К наиболее распространенным фреймворкам относятся Tensorflow, Keras и PyTorch. При разработке системы использовался PyTorch.

PyTorch – фреймворк машинного обучения для языка Python с открытым исходным кодом, написанный на базе Torch. Используется он в различных задачах: компьютерное зрение, обработка естественного языка и прочих. Этот фреймворк способствует увеличению скорости и эффективности обучения моделей, что в данном случае имеет огромное значение. Он обладает рядом неоспоримых преимуществ:

* Благодаря архитектуре фреймворка, процесс создания модели достаточно прост и прозрачен;
* Поддержка декларативного параллелизма данных;
* Имеет много предварительно обученных моделей и готовых модульных частей, которые легко комбинировать;

В качестве среды исполнения кода и обучения модели был использован Google Colaboratory [13]. В его основе лежит блокнот Jupyter для работы на Python, только с базой на облачном средстве хранения данных Google Диске, а не на диске компьютера. Использование подобной среды упрощает работу с моделями машинного обучения, их обучение и тестирование. В Google Colaboratory предустановлено огромное количество библиотек Python для работы с машинным обучением и нейронными сетями.

К особенностям этой среды можно отнести то, что помимо использования CPU, она предоставляет возможность использования бесплатного удалённого доступа к машине с GPU и TPU, что может сильно сказаться на скорости обучения и тестирования некоторых моделей.

CPU — центральный процессор — мозг компьютера, который выполняет операции с данными. Настолько универсален, что может использоваться почти для всех задач: от записи фотографий на накопители до моделирования физических процессов.

GPU — графический процессор. Обрабатывает данные быстрее, так как задачи выполняет параллельно, а не последовательно, как CPU. Он заточен исключительно под графику, поэтому на нем удобнее работать с изображением и видео, например, заниматься 3D-моделированием или монтажом.

TPU — тензорный процессор, разработка Google. Он предназначен для тренировки нейросетей. У этого процессора в разы выше производительность при больших объемах вычислительных задач.

При разработке было использовано большое количество библиотек Python для работы с машинным обучением и нейронными сетями. Некоторые из них:

* Matplotlib – библиотека на языке программирования Python для визуализации данных двумерной (2D) графикой;
* Nltk (Natural Language Toolkit) - пакет библиотек и программ для символьной и статистической обработки естественного языка, написанных на языке программирования Python. Содержит графические представления и примеры данных;
* Numpy – математическая библиотека для языка Python, которая предоставляет реализацию вычислительных алгоритмов, оптимизированных для работы с многомерными массивами;
* Pillow – библиотека изображений Python, нужна для обработки графики в Python;
* Argparse – библиотека предоставляющая возможность работы с параметрами командной строки;
* Pycocotools – библиотека от Microsoft, для импорта информации;
* Google\_trans\_new – библиотека необходимая для выполнения перевода логического текстового описания [10];
* Pandas – библиотека для анализа данных на языке Python;
* Torchvision – библиотека с инструментами для компьютерного зрения на PyTorch;

## Обработка датасета

В разрабатываемой системе присутствуют две нейронных сети, каждая из которых выполняет свою определённую роль в архитектуре системы. Для того, чтобы система могла функционировать правильно необходимо обучить обе нейронные сети на определённых наборах данных.

Для этой цели был взят открытый датасет Microsoft COCO 2014 года [8]. Он состоит из изображений и текстовых аннотаций к этим изображениям. Это набор данных большого объема, изначально собранный для задач обнаружения, сегментации и локализации. Датасет сразу разделён на несколько выборок данных, а именно на тренировочную выборку, в которую входят 83 тысячи изображений (13ГБ) и текстовые аннотации к ним (160МБ), проверочную выборку, которая включает в себя 41 тысячу изображений (6ГБ) и текстовые аннотации к ним (81МБ), а также тестовую выборку, состоящую из 41 тысячи изображений (6ГБ). Плюсом этого набора данных можно выделить его большой объём. Также его особенностью является то, что он содержит изображения сложных повседневных сцен, содержащих объекты в их естественном контексте.

Данные в данном датасете подразделяются на 91 категорию объектов, которые в свою очередь сгруппированы в 11 групп.

На рисунке 3 представлена классификация объектов набора данных Microsoft COCO.

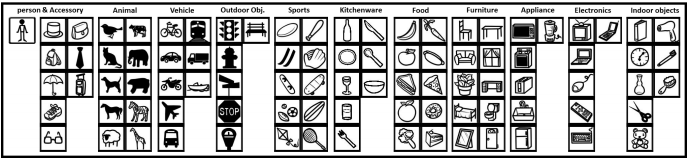


Рисунок 3 – Классификация объектов набора данных MS COCO

Однако, у подобного датасета есть и существенные недостатки, которые выявились походу его обработки и обучения сетей:

* У набора данных нет определённой предметной области, то есть он содержит абсолютно различные объекты;
* Малый размер текстовых аннотаций к изображениям;
* Формат аннотаций, они содержат простое текстовое описание объектов на изображении, не имеют сегментации, для указания конкретного объекта на изображении;
* Недостаточное количество изображений, в связи с отсутствием конкретной предметной области набора данных;

Для хранения датасета использовалось облачное дисковое хранилище Google Диск. Причина выбора именно этого хранилища заключается в том, что его удобно связывать и работать со средой Google Colaboratory.

Все изображения в наборе данных имеют разные размеры, что очень сильно затрудняет их дальнейшее использование при обучении. Для решения данной проблемы необходимо обработать изображения из данного набора данных. В процессе обработки, все изображения будут приведены к одному одинаковому размеру, для возможности дальнейшего, беспроблемного обучения нейронных сетей.

Помимо графических данных, также необходимо обработать и текстовые аннотации датасета. Обработка текстовых описаний изображений заключается в создании словаря, содержащего все возможные слова из всех аннотаций, в сочетании с уникальными индексами этих слов, для возможности дальнейшего декодирования идентификаторов признаков изображений в слова.

## Энкодер CNN

После своей обработки изображения готовые к использованию в процессе обучения моделей. В качестве первого входного блока обработки изображения используется CNN [7].

CNN (Convolutional Neural Network), или сверточная нейронная сеть – класс глубоких нейронных сетей, часто применяемый в анализе визуальных образов. Сверточные нейронные сети являются разновидностью многослойного перспептрона с использованием операций свёртки. Они нашли применение в распознавании изображений и видео, рекомендательных системах, классификации изображений, NLP (natural language processing) и анализе временных рядов.

В сверточной нейронной сети в операции свертки используется лишь ограниченная матрица весов небольшого размера, которая перемещается по всему обрабатываемому слою, изначально по самому изображению, формируя после каждого сдвига сигнал активации для нейрона следующего слоя с аналогичной позицией. То есть для различных нейронов выходного слоя используются одна и та же матрица весов, которую также называют ядром свертки. Её интерпретируют как графическое кодирование какого-либо признака, например, наличие наклонной линии под определённым углом. Тогда следующий слой, получившийся в результате операции свёртки такой матрицей весов, показывает наличие данного признака в обрабатываемом слое и ее координаты, формируя так называемую карту признаков. Естественно, в сверточной нейронной сети набор весов не один, а целая гамма, кодирующая элементы изображения. При этом такие ядра свертки не закладываются исследователем заранее, а формируются самостоятельно путем обучения сети. Проход каждым набором весов формирует свой собственный экземпляр карты признаков, делая нейронную сеть многоканальной, много независимых карт признаков на одном слое. Также следует отметить, что при переборе слоя матрицей весов ее передвигают обычно не на полный шаг, то есть размер этой матрицы, а на небольшое расстояние. Так, например, при размерности матрицы весов 5×5 ее сдвигают на один или два нейрона вместо пяти, чтобы не пропустить искомый признак.

Далее следует операция субдискретизации, которая выполняет уменьшение размерности сформированных карт признаков. В данной архитектуре сети считается, что информация о факте наличия искомого признака важнее точного знания его координат, поэтому из нескольких соседних нейронов карты признаков выбирается максимальный и принимается за один нейрон уплотненной карты признаков меньшей размерности. За счёт данной операции, помимо ускорения дальнейших вычислений, сеть становится более инвариантной к масштабу входного, изначального изображения.

Слой свертки (convolution layer) — это основной блок сверточной нейронной сети. Слой свертки включает в себя для каждого канала свой фильтр, ядро свертки которого обрабатывает предыдущий слой по фрагментам, суммируя результаты поэлементного произведения для каждого фрагмента. Весовые коэффициенты ядра свертки неизвестны и устанавливаются в процессе обучения.

Особенностью сверточного слоя является сравнительно небольшое количество параметров, устанавливаемое при обучении. Так, например, если исходное изображение имеет размерность 100×100 пикселей по трем каналам, что порядка 30000 входных нейронов, а сверточный слой использует фильтры c ядром 3x3 пикселя с выходом на 6 каналов, тогда в процессе обучения определяется только 9 весов ядра, однако по всем сочетаниям каналов, то есть 9×3×6 =162, в таком случае данный слой требует нахождения только 162 параметров, что существенно меньше количества искомых параметров полносвязной нейронной сети.

Рассмотрим типовую структуру сверточной нейронной сети более подробно. Сеть состоит из большого количества слоев. После начального слоя, а именно входного изображения, сигнал проходит серию сверточных слоев, в которых чередуется собственно свертка и субдискретизация. Чередование слоев позволяет составлять карты признаков из карт признаков, на каждом следующем слое карта уменьшается в размере, но увеличивается количество каналов. На практике это означает способность распознавания сложных иерархий признаков. Обычно после прохождения нескольких слоев карта признаков вырождается в вектор или даже скаляр, но таких карт признаков становятся сотни. На выходе сверточных слоев сети дополнительно устанавливают несколько слоев полносвязной нейронной сети (персептрон), на вход которому подаются оконечные карты признаков. На рисунке 4 представлена архитектура CNN.

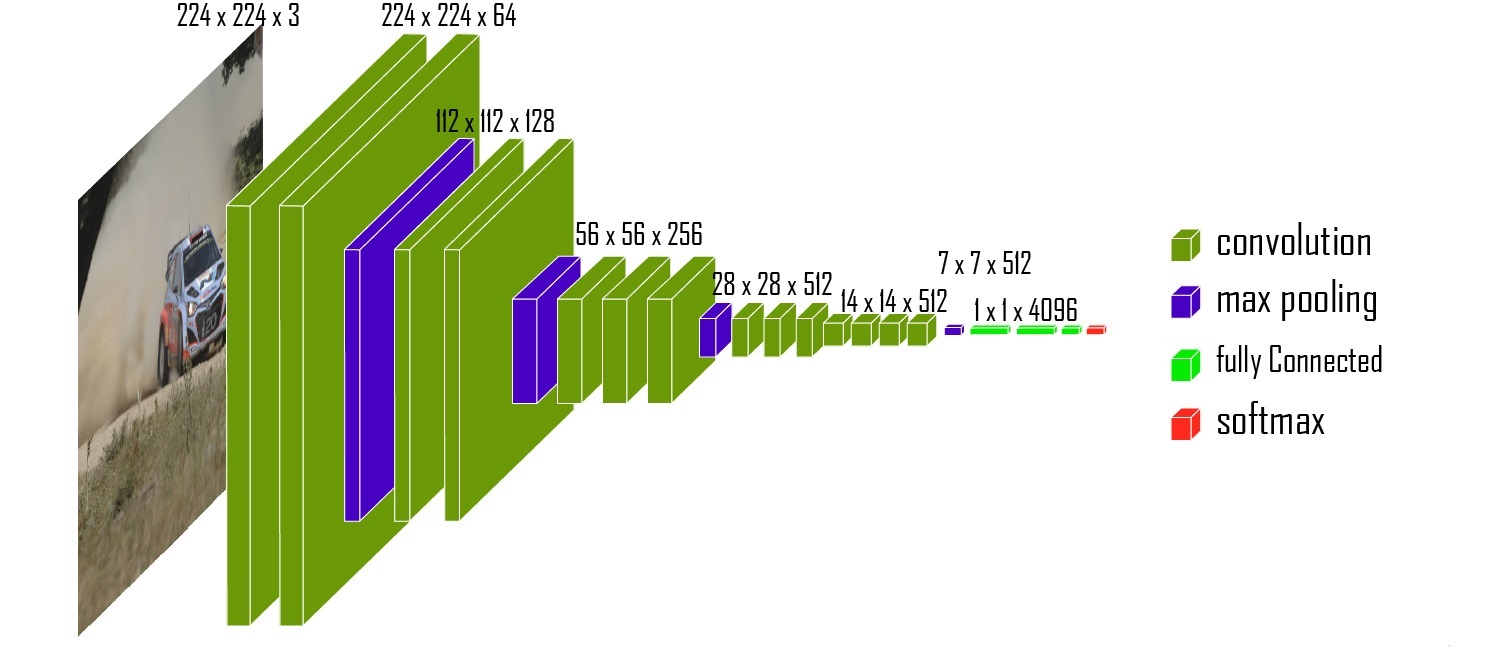


Рисунок 4 – Архитектура CNN

После прохождения CNN вектор признаков проходит процесс токенизации, то есть присвоения признакам идентификаторов. Выходной вектор имеет размерность необходимую, для входного потока блока декодера.

## Декодер RNN

Далее, финальный вектор признаков поступает во второй блок, декодер. Этот блок должен декодировать признаки из этого вектора и сгенерировать на их основе логическое текстовое описание изображения. В системы для этого блока была использована RNN [6].

RNN (Recurrent Neural Network), или рекуррентные нейронные сети – вид нейронных сетей, где связи между элементами образуют направленную последовательность. Благодаря этому появляется возможность обрабатывать серии событий во времени или последовательные пространственные цепочки. В отличие от многослойных персептронов, рекуррентные сети могут использовать свою внутреннюю память для обработки последовательностей произвольной длины. Поэтому сети RNN применимы в различных задачах, где нечто целостное разбито на части. Было предложено много различных архитектурных решений для рекуррентных сетей от простых до сложных. Наиболее популярные на данный момент LSTM и GRU.

При разработке информационной мультимодальной системы, в архитектуре была использована сеть LSTM (Long short-term memory) [5]. LSTM-сеть является универсальной в том смысле, что при достаточном числе элементов сети она может выполнить любое вычисление, на которое способен обычный компьютер, для чего необходима соответствующая матрица весов, которая может рассматриваться как программа. В отличие от традиционных рекуррентных нейронных сетей, LSTM-сеть хорошо приспособлена к обучению на задачах классификации, обработки и прогнозирования временных рядов в случаях, когда важные события разделены временными лагами с неопределенной продолжительностью и границами. Относительная невосприимчивость к длительности временных разрывов дает LSTM преимущество по отношению к альтернативным рекуррентным нейронным сетям. Основная особенность этой сети – это возможность обучения долгосрочным зависимостям. Благодаря чему сеть не требует объемного обучения.

LSTM прекрасно справляется с задачей генерации текста, когда нужно учитывать контекст. В данной системе LSTM состоит из нескольких слоев, каждый из которых генерирует текст, а точнее всего лишь одно слово. На вход подается вектор признаков, который несет в себе токенизированное описание изображения, но помимо идентификаторов текста, там так же присутствуют идентификаторы начала, разделения и конца предложения. Только на вход первого слоя LSTM подается вектор признаков, на все последующие слои подается слово, сгенерированное на предыдущем слое. Вектор признаков, содержащий идентификаторы слов, декодируется при помощи словаря, содержащего в себе все слова, которые встречались в аннотациях, а также их уникальные идентификаторы. Таким образом сеть понимает, какие слова, а точнее строка слов содержится в векторе признаков. Первый слой всегда подает распознает, а затем генерирует признак начала предложения «start», который затем подается на второй слой, где генерируется слово, и по такой аналогии продолжается до последнего символа вектора признаков, когда последний слой генерирует слово окончания предложения «end». На рисунке 5 представлена архитектура LSTM [5].

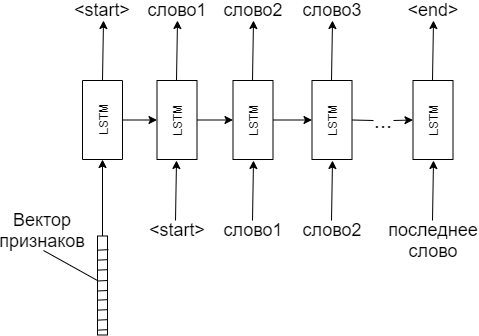


Рисунок 5 – Архитектура LSTM

## Обучение модели

Для правильного функционирования мультимодальной информационной системы необходимо обучить ее в достаточном объеме. Для этого использовался набор данных MS COCO, а точнее его тренировочная и проверочная выборки.

Процесс обучения системы продвигается последовательно, сначала данные поступают на блок энкодер, где обрабатываются и формируется вектор признаков. После чего вектор признаков продолжает свое движение по системе и попадает в блок декодера, где он преобразуется, также в блок LSTM поступает и текстовая аннотация к данному изображению совместно с доступом к словарю, содержащему все слова и их уникальные идентификаторы, что в свою очередь и даёт возможность обучить сеть, распознавать конкретные признаки из вектора, и на этой основе генерировать слова в дальнейшем.

В качестве модели CNN использовалась предобученная модель ResNet-152, которая, в процессе выполнения работы, была дополнительно обучена на наборе данных MS COCO 2014 [8]. Обучение проводилось в среде Google Colaboratory, так как она обеспечивает доступ к высокопроизводительным машинам, что сильно ускоряет процесс обучения.

Обучение состояло из 5 эпох, где каждая содержала порядка 3000 ступеней. На рисунке 6 можно увидеть процесс обучения моделей.

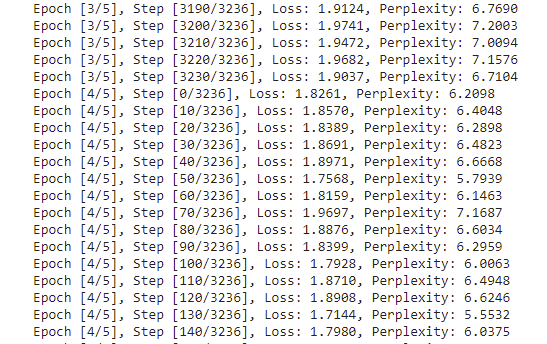


Рисунок 6 – Процесс обучения моделей

Промежуточные этапы обучения модели сохранялись, чтобы в случае непредвиденных обстоятельств была возможность загрузиться с определенного момента и продолжить обучение, а не начинать его с начала, пример сохранений можно увидеть на рисунке 7. На этапе сохранения создавалось два файла, один для энкодера и один для декодера.

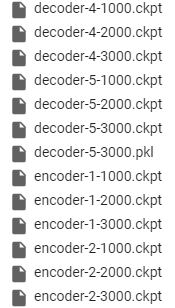


Рисунок 7 – Пример сохраненных моделей

## Модуль перевода описания текста

Текстовые логические описания генерируются на английском языке, что не совсем удобно с точки зрения восприятия. В связи с этим было принято решение использовать модуль перевода текста с английского на русский язык.

Модуль должен принимать строку из блока декодера (LSTM), затем связывается при помощи сети интернет с сервером перевода, после чего переводить эту строку на русский язык с достаточной точностью. Архитектура модуля перевода представлена на рисунке 8.

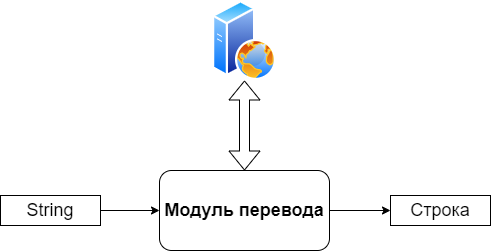


Рисунок 8 – Архитектура модуля перевода

Было рассмотрено несколько вариантов переводчиков, которые было возможно встроить в данную архитектуру.

Варианты выбранных переводчиков представлены в таблице 15.

Таблица 15 – Варианты переводчиков

| **Вариант** | **Обозначение** | **Описание** |
| --- | --- | --- |
| Google Translate API | В1 | Переводчик, созданный компанией Google. Имеет возможность быстро переводить тексты на более чем сто языков, просто в применении в приложениях и сайтах [10]. |
| Yandex Translate API | В2 | Переводчик, созданный компанией Yandex. С помощью данного API можно получить доступ к онлайн-сервису машинного перевода Яндекса. Он поддерживает более 90 языков и умеет переводить отдельные слова и целые тексты. Этот API позволяет встроить переводчик в мобильное приложение или веб-сервис для конечных пользователей [11]. |
| Microsoft text translation API | В3 | Переводчик, созданный компанией Microsoft. Это переводчик, входящий в состав Azure Cognitive Services, является облачной службой машинного перевода, поддерживает 90 языков и диалектов. Можно использовать для создания приложений, веб-сайтов, инструментов или любого решения, требующего многоязыковой поддержки [12]. |

Критерии для сравнения выбранных API для перевода представлены в таблице 16.

Таблица 16 – Критерии для выбора переводчика

| **Критерий** | **Код критерия** | **Описание** | **Весовой коэффициент** |
| --- | --- | --- | --- |
| Документация | К1 | Документация, ее доступность, понятность, наличие примеров использования | 2α |
| Функциональные возможности | К2 | Наличие минимального необходимого функционала | α |
| Доступность | К3 | Бесплатный или платный сервис | 2α |
| Качество перевода | К4 | Точность, грамотность и качество перевода | 4α |
| Простота использования | К5 | Насколько просто можно освоить и использовать сервис, для включения его в работу системы | α |
| Количество поддерживаемых языков и диалектов | К6 | Количество поддерживаемых языков и диалектов | 2α |
| Быстродействие | К7 | Скорость перевода при использовании из приложения | 4α |

Определим величину α, рассчитав значения весовых коэффициентов по формуле 1:

Для проведения сравнения выбранных вариантов переводчиков по данным критериям необходимо значения качественных критериев перевести в количественные по шкале, представленной в таблицах 17, 18, 19, 20, 21 [1].

Таблица 17 – Вербально-числовая шкала для К1 (Документация)

| **Качественная оценка** | **Оценка** |
| --- | --- |
| Качественная документация на обоих языках с примерами использования в приложениях | 1 |
| Качественная документация на английском языке с примерами использования в приложениях | 0,9 |
| Документация без примеров, но с описанием использования | 0,8 |
| Плохая документация | 0,6 |

Таблица 18 – Вербально-числовая шкала для К2 (Функциональные возможности)

|  |  |
| --- | --- |
| **Качественная оценка** | **Оценка** |
| Большой базовый функционал | 1 |
| Средний базовый функционал | 0,8 |
| Малый базовый функционал | 0,6 |

Таблица 19 – Вербально-числовая шкала для К3 (Доступность)

|  |  |
| --- | --- |
| **Качественная оценка** | **Оценка** |
| Бесплатная | 1 |
| С бесплатным периодом | 0,7 |
| Платная | 0 |

Таблица 20 – Вербально-числовая шкала для К4 (Качество перевода)

|  |  |
| --- | --- |
| **Качественная оценка** | **Оценка** |
| Высокое качество перевода | 1 |
| Хорошее качество перевода, незначительные ошибки в формах слов | 0,9 |
| Среднее качество перевода, ошибки в переводе слов, но с сохранением смысла контекста | 0,8 |
| Низкое качество перевода, неверный перевод слов в контексте | 0,6 |

Таблица 21 – Вербально-числовая шкала для К5 (Простота использования)

| **Качественная оценка** | **Оценка** |
| --- | --- |
| Легко использовать | 1 |
| Нелегко использовать, но интуитивно понятно | 0,9 |
| Достаточно сложно использовать, но интуитивно понятно | 0,8 |
| Сложно использовать | 0,6 |

Проведем сравнительный анализ по методу взвешенных локальных критериев вариантов переводчиков, выбранных ранее [1].

Результаты анализа приведены ниже в таблице 22.

Таблица 22 – Результаты сравнения

| **Вариант** | **Весовой коэффициент** | **В1** | **В2** | **В3** |
| --- | --- | --- | --- | --- |
| **К1** | 2α | 1 | 1 | 0,9 |
| **К2** | α | 1 | 1 | 1 |
| **К3** | 2α | 1 | 0 | 0,7 |
| **К4** | 4α | 1 | 1 | 0,9 |
| **К5** | α | 1 | 0,8 | 1 |
| **К6** | 2α | 1 | 0,9 | 0,9 |
| **К7** | 4α | 0,85 | 1 | 0,8 |
| **Итоговый вес** | | 15,4α | 13,6α | 13,8α |

Максимальную оценку имеет вариант В1 – Google Translate API. Таким образом именно этот переводчик будет использоваться в мультимодальной системе в модуле перевода текстового описания.

В системе будет использоваться библиотека, основанная на библиотеке googletrans, а точнее google\_trans\_new. Для его использования в системе необходимо установить необходимые библиотеки и модули googletrans и google\_trans\_new.

После этого достаточно просто импортировать google\_trans\_new в коде программы, затем создать объект класса google\_translator. И библиотека перевода готова к работе.

## Создание веб-приложения

Для более наглядной демонстрации работы программы и для упрощения ее запуска, было принято решение создать веб приложение под эту мультимодальную информационную систему. Так как, система для функционирования нуждается в интернете не было необходимости создавать обычное автономное приложение, которое бы могло работать без интернета.

Веб приложение должно обладать необходимым функционалом, который требует система, а также иметь возможность совмещаться с кодом на языке Python и хорошо подходить для работы с машинным обучением. Исходя из всех ранее названных требований, было принято решение об использовании в системе библиотеки Streamlit.

Streamlit – фреймворк с открытым кодом, написанный, как библиотека для языка программирования Python [9]. Streamlit предназначен для разработки пользовательских интерфейсов, в которых применяются технологии машинного обучения. У данного фреймворка есть ряд неоспоримых достоинств:

* Позволяет без глубокого изучения фронтенда создавать пользовательские интерфейсы для отображения данных из программ;
* Является библиотекой языка Python, а значит прекрасно с ним функционирует;
* Имеет высокий уровень стабильности и почти не имеет каких-либо критических ошибок;
* Приложение запускается локально в веб-браузере на устройстве пользователя;
* Хорошо работает с моделями машинного обучения, невероятно удобен, для вывод данных;
* Интуитивно понятный язык и синтаксис;

Для начала необходимо в исполняемый файл, написанный на языке Python, импортировать библиотеку Streamlit. Затем, создадим оглавление интерфейса нашего приложения при помощи команды title. После этого на форме интерфейса мы выделяем две области под необходимые задачи. Первая область будет отвечать за ввод данных, а точнее ввод изображения. Для этого создадим форму загрузки данных, которая будет состоять из элемента file\_uploader, у которого мы ограничим перечень доступных тип файлов для ввода, и кнопки «Отправить» (рисунок 9).

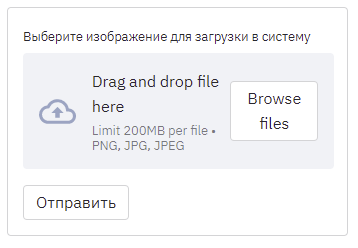


Рисунок 9 – Форма загрузки изображения в систему

Второе поле мы выделим для отображения текстового описания введенного изображения. До момента ввода изображения это поле будет пустым. После загрузки изображения в первую форму необходимо нажать на кнопку отправить, после чего введенное изображение будет отображено в приложении. Затем после небольшой задержки, во втором поле появится текстовое описание введенного изображения. Задержки генерации описания могут быть связаны с различными вещами:

* Низкая производительность аппаратной составляющей пользователя, в следствие чего обработка изображения и генерация логического текстового описания может протекать дольше;
* Низка скорость подключения к сети интернет, так как последний модуль системы, а именно модуль перевода, напрямую связан с интернет соединением;

Для запуска приложения локально необходимо просто в командной строке прописать команду (рисунок 10). После чего открыть выведенную локальную ссылку в браузере.

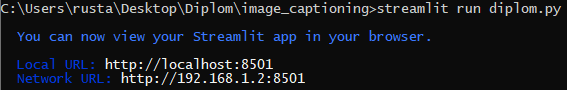


Рисунок 10 – Запуск приложения Streamlit через командную строку

Итоговый интерфейс приложения, после загрузки в него изображения, вывода в приложении этого изображения и вывода текстового логического описания, представлен на рисунке 11.

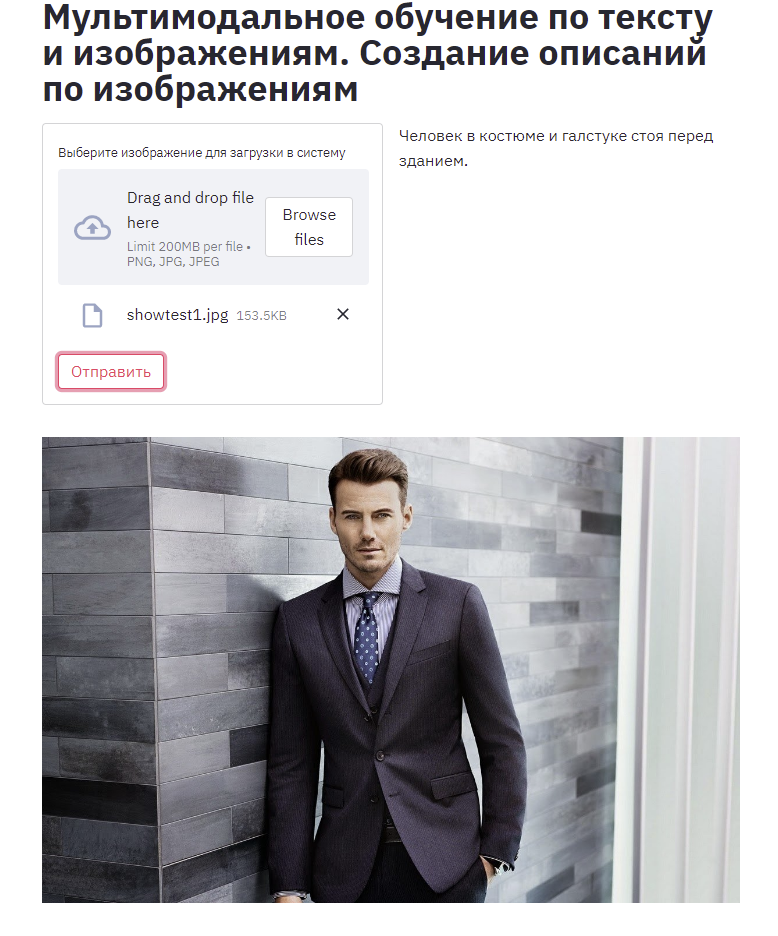


Рисунок 11 – Интерфейс веб-приложения

# ЗАКЛЮЧЕНИЕ

При разработке мультимодальной информационной системы создания описаний к изображениям были получены следующие результаты:

* Была изучена предметная область, связанная с мультимодальным обучением, обнаружением объектов на изображении и генерация описаний к изображениям.
* Были исследованы различные типы архитектур для задач генерации текстовых описаний и проведено сравнение этих вариантов, для выявления лучшей архитектуры на практике.
* Были сформулированы критерии качества разрабатываемой мультимодальной информационной системы и доказана актуальность разработки системы.
* Выбраны современные технологии для реализации системы.
* Спроектирована и описана архитектура системы.
* Спроектированы и описаны входные и выходные данные в системе.
* Разработан и описан модуль перевода текстовых описаний.
* Разработана предобработка входных данных.
* Реализованы и успешно обучены блоки энкодера и декодера.
* Разработано и описано веб-приложение для системы.

Полученное информационно-программное изделие имеет возможности расширения за счет подключения, изменения или дообучения новых блоков и модулей в системе, а также за счет развития интерфейсов приложения. Это позволит системе привлечь новых пользователей и оставаться конкурентно способной среди аналогов.

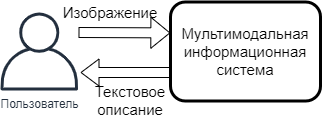
# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

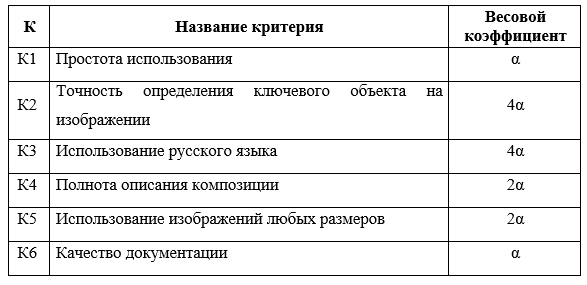
1. Постников В. Основы эксплуатации автоматизированных систем обработки информации и управления. Краткий курс: учеб. пособие. — 177 с.
2. CNN+CNN: Convolutional Decoders for Image Captioning [Электронный ресурс] // arxiv.org URL: https://arxiv.org/pdf/1805.09019.pdf / (дата обращения: 29.05.2021)
3. Deep Visual-Semantic Alignments for Generating Image Descriptions [Электронный ресурс] // arxiv.org URL: https://arxiv.org/pdf/1412.2306.pdf / (дата обращения: 29.05.2021)
4. Semantic Regularisation for Recurrent Image Annotation [Электронный ресурс] // arxiv.org URL: https://arxiv.org/pdf/1611.05490.pdf / (дата обращения: 30.05.2021)
5. Долгая краткосрочная память [Электронный ресурс] // ru.wikipedia.org URL: https://ru.wikipedia.org/wiki/Долгая\_краткосрочная\_память / (дата обращения: 29.05.2021)
6. Рекуррентная нейронная сеть [Электронный ресурс] // ru.wikipedia.org URL: https://ru.wikipedia.org/wiki/Рекуррентная\_нейронная\_сеть / (дата обращения: 29.05.2021)
7. Сверточная нейронная сеть [Электронный ресурс] // ru.wikipedia.org URL: https://ru.wikipedia.org/wiki/Сверточная\_нейронная\_сеть / (дата обращения: 29.05.2021)
8. Microsoft COCO: Common Objects in Context [Электронный ресурс] // arxiv.org URL: https://arxiv.org/pdf/1405.0312.pdf / (дата обращения: 29.05.2021)
9. Streamlit 0.82.0 documentation [Электронный ресурс] // docs.streamlit.io URL: https://docs.streamlit.io/en/stable / (дата обращения: 29.05.2021)
10. Google Translate API [Электронный ресурс] // translate.google.com URL: https://cloud.google.com/translate/?hl=ru / (дата обращения: 29.05.2021)
11. Yandex Translate API [Электронный ресурс] // yandex.ru URL: https://yandex.ru/dev/translate/doc/dg/concepts/about.html / (дата обращения: 29.05.2021)
12. Translator Text API [Электронный ресурс] // microsoft.com URL: https://microsoft.com/en-us/translator/business/translator-api / (дата обращения: 29.05.2021)
13. Google Colaboratory [Электронный ресурс] // colab.research.google.com URL: https://colab.research.google.com/notebooks/basic\_features\_overview.ipynb / (дата обращения: 29.05.2021)

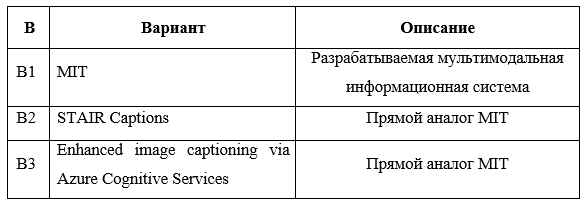
# ПРИЛОЖЕНИЕ А

# ГРАФИЧЕСКИЕ МАТЕРИАЛЫ

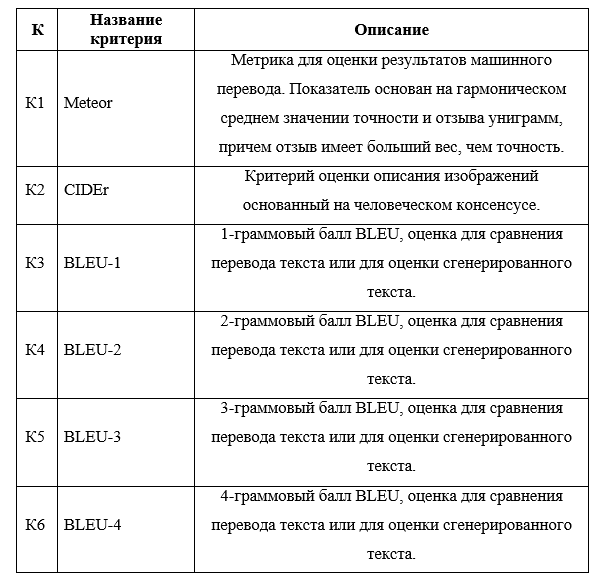
1. Описание предметной области
2. Критерии сравнения с аналогами
3. Варианты аналогов для сравнения
4. Сравнение с аналогами
5. Критерии сравнение архитектур
6. Варианты архитектур
7. Нормированные критерии архитектур
8. Сравнение архитектур методом близости к идеалу
9. Сравнение архитектур методом ранжирования Борда
10. Архитектура мультимодальной информационной системы
11. Архитектура CNN
12. Архитектура LSTM
13. Архитектура модуля перевода
14. Варианты переводчиков
15. Критерии выбора переводчиков
16. Сравнение переводчиков
17. Интерфейс веб-приложения

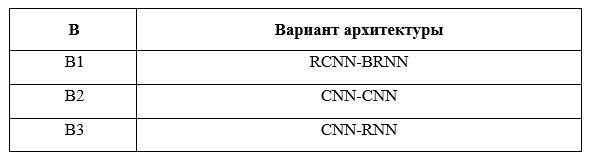


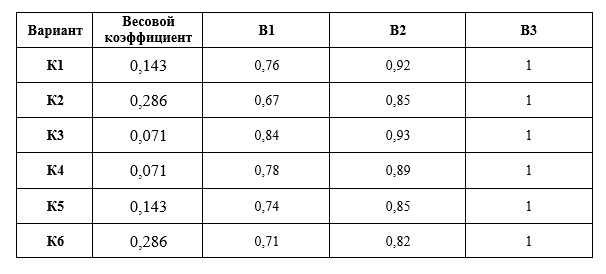


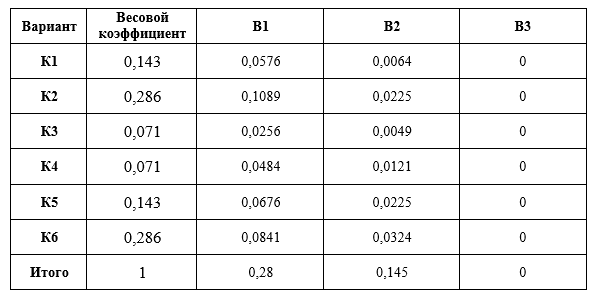


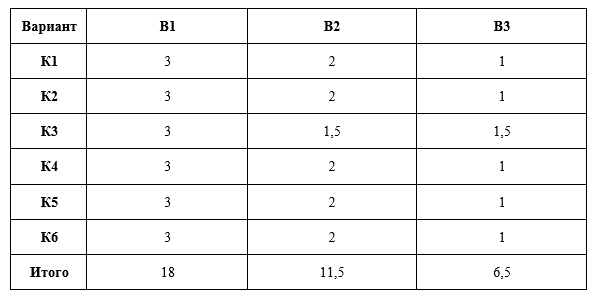


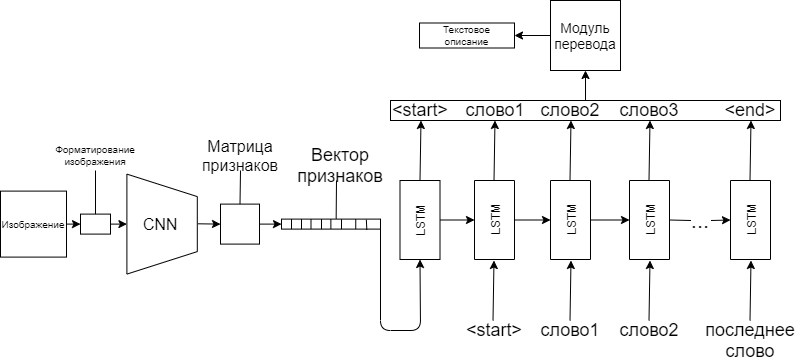


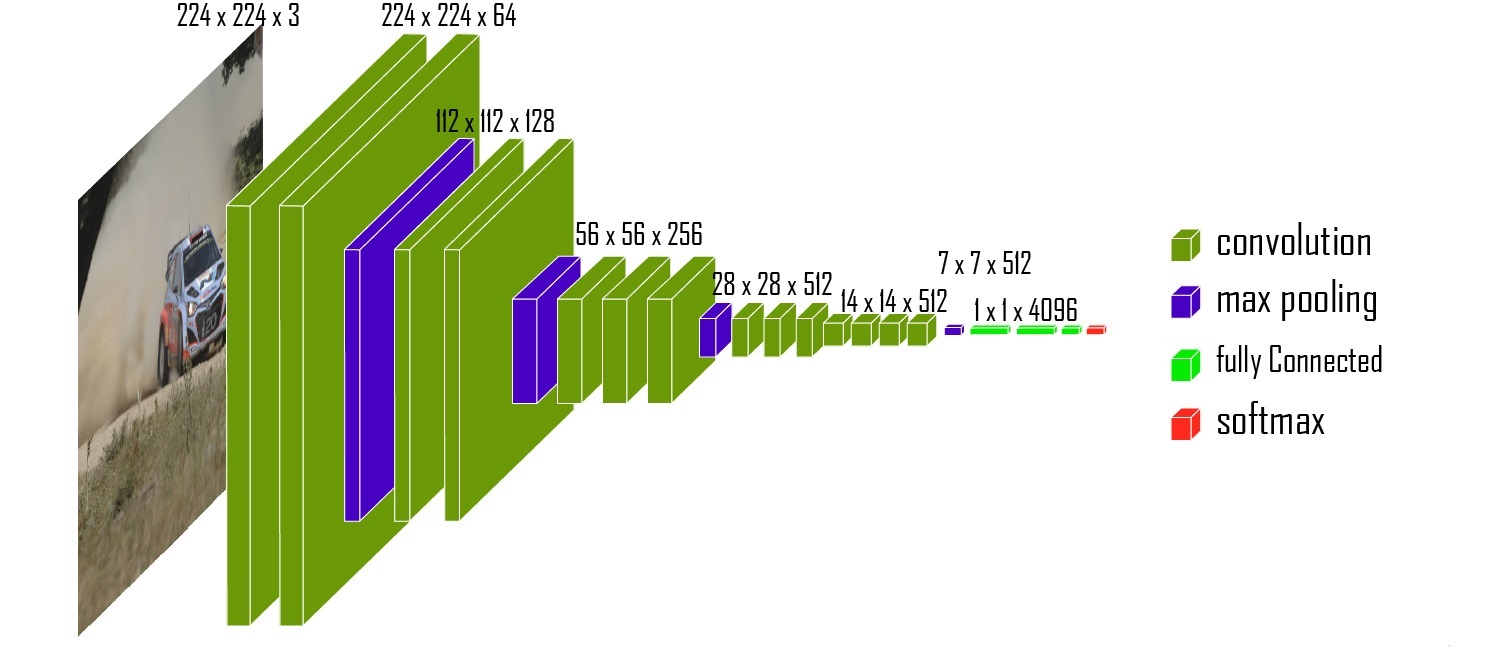


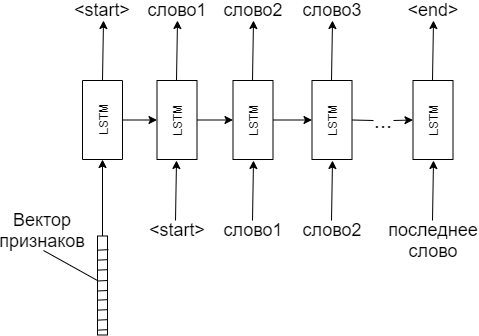


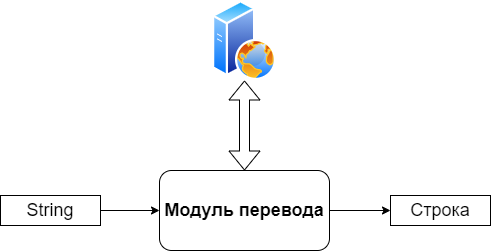


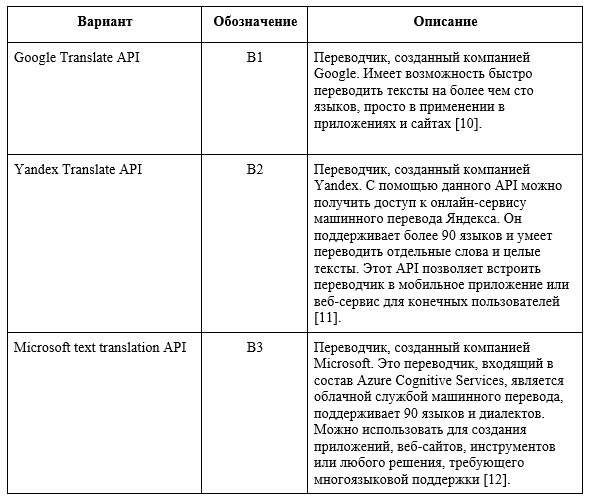




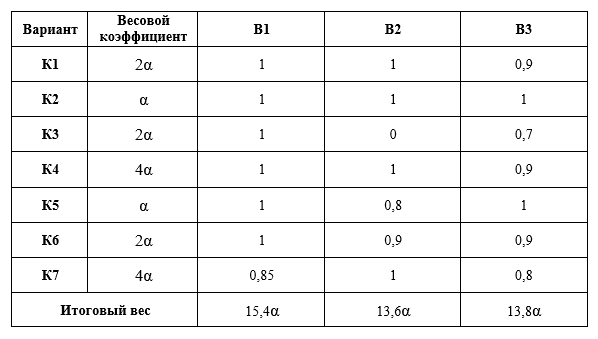


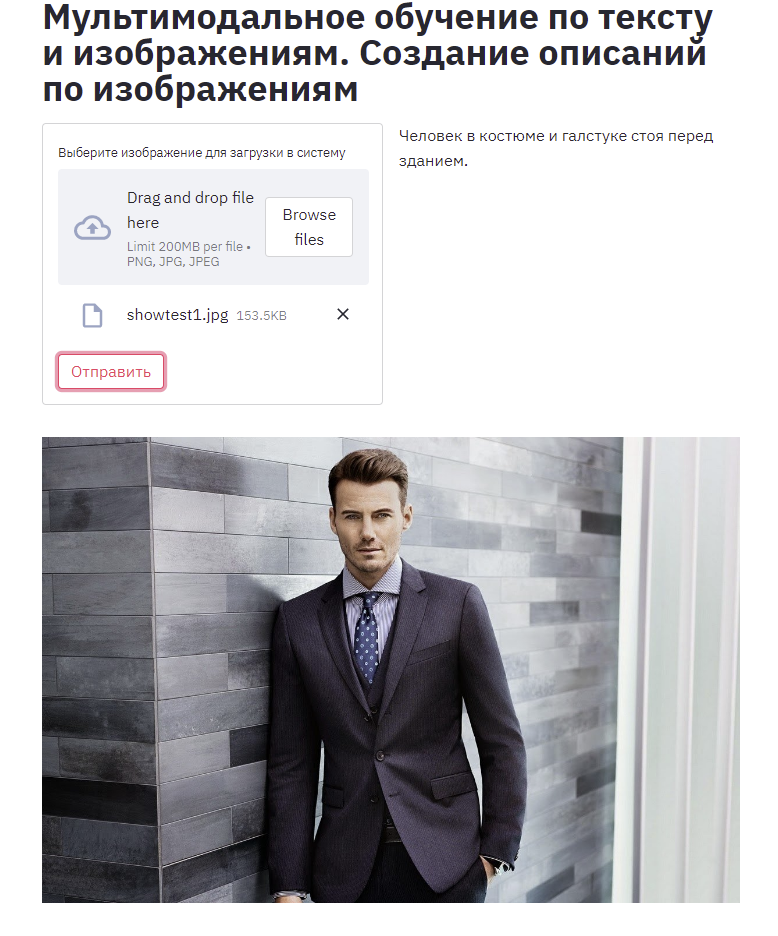












# ПРИЛОЖЕНИЕ Б

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего профессионального образования**

**«Московский государственный технический университет имени Н.Э. Баумана»**

**(МГТУ им. Н.Э.Баумана)**

Утверждаю Согласовано

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_»\_\_\_\_\_\_\_\_\_\_2021 г. «\_\_\_»\_\_\_\_\_\_\_\_\_\_2021 г.

**«Мультимодальное обучение по тексту и изображениям. Создание описаний по изображениям»**

техническое задание

(вид документа)

писчая бумага

(вид носителя)

7

(количество листов)

Исполнитель:

студент группы ИУ5-84Б

Сафин Р. Р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_2021 г.

Москва, 2021 г.

**СОДЕРЖАНИЕ**

[1 Наименование 73](#_Toc199783522)

[2 Основание для разработки 73](#_Toc199783523)

[3 Исполнитель 73](#_Toc199783524)

[4 Назначение и цель разработки 73](#_Toc199783525)

[5 Задачи, подлежащие решению 74](#_Toc199783526)

[6 Требования к разработке 74](#_Toc199783526)

[6.1 Требования к функциональным характеристикам 74](#_Toc199783528)

[6.2 Требования к архитектуре системы 74](#_Toc199783529)

[6.3 Требования к составу программных средств 74](#_Toc199783530)

[6.4 Требования к программным компонентам 75](#_Toc199783531)

[6.5 Требования к входным и выходным данным 75](#_Toc199783532)

[6.6 Требования к составу технических средств 75](#_Toc199783535)

[7 Этапы разработки 76](#_Toc199783541)

[8 Техническая документация, предъявляемая по окончании работы 7](#_Toc199783542)7

[9 Порядок приемки работы 7](#_Toc199783543)7

10 [Дополнительные условия 77](#_Toc199783544)

# Наименование

Мультимодальное обучение по тексту и изображениям. Создание описаний по изображениям. Краткое обозначение объекта испытаний: «MIT» (Multimodal Image-to-Text)

# Основание для разработки программного продукта

Основанием для разработки является задание на ВКР, подписанное руководителем ВКР и утвержденное заведующим кафедрой ИУ5 «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана.

# Исполнитель

Исполнителем является студент МГТУ им. Н. Э. Баумана группы ИУ5-84Б Сафин Рустам Равильевич.

# Назначение и цель разработки

Разрабатываемая система предназначена для создания текстовых логических описаний изображений.

Целью данной разработки является создание системы, позволяющей генерировать логическое текстовое описание на основе изображения, для дальнейшего возможного использования полезной информации и упрощения восприятия изображений в различных областях.

# Задачи, подлежащие решению

В ходе разработки автоматизированной системы должны быть решены следующие задачи:

1. Исследование предметной области;
2. Разработка архитектуры системы;
3. Разработка программного изделия;
4. Тестирование программного изделия;
5. Отладка программного изделия;

# Требования к разработке

## Требования к функциональным характеристикам

1. Доступность запущенного приложения локально;
2. Загрузка нового изображения в приложение;
3. Сохранение и отправка изображения в мультимодальную модель;
4. Вывод на экран загруженного в систему изображения;
5. Вывод текстового описания загруженного изображения;

## Требования к архитектуре системы

Архитектура системы состоит из нескольких блоков обработки различных типов данных: текстовых и графических. А также блока генерации текста на основе изображения.

## Требования к составу программных средств

Для работы системы на устройстве пользователя требуется:

1. Операционная система не регламентируется.
2. Любой браузер

## Требования к программным компонентам

Основным требованием к программным компонентам является корректное исполнение функций, заявленных в пункте 6.1.

## Требования к входным и выходным данным

### **Требования к входным данным**

Входными данными являются изображения форматов jpg, png, jpeg, загружаемые в систему для дальнейшего анализа. Объём изображения не должен превышать 200МБ.

### **Требования к выходным данным**

### Выходными данными для системы являются текстовые описания изображений.

## Требования к составу технических средств

Для пользователей необходимо устройство с установленным браузером и доступом в интернет.

# Этапы разработки

Разработка программного изделия выполняется по этапам, представленным в следующей таблице:

| **Номер этапа** | **Наименование этапа и содержание** | **Сроки исполнения** |
| --- | --- | --- |
| 1 | Разработка и утверждение ТЗ | Март 2021 |
| 2 | Исследование предметной области | Март 2021 |
| 3 | Разработка структуры программы | Март-Апрель 2021 |
| 4 | Создание системы | Апрель 2021 |
| 5 | Тестирование системы | Апрель-Май 2021 |
| 6 | Демонстрация работы программы | Май 2021 |
| 7 | Оформление документации | Май-Июнь 2021 |
| 8 | Защита работы | Июнь 2021 |

# Техническая документация, предъявляемая по окончании работы

1. Техническое задание;
2. Расчетно-пояснительная записка;
3. Программа и методика испытаний;
4. Текст программы;
5. Графические материалы по продукту;

# Порядок приёма работы

Приём и контроль программного изделия осуществляется в соответствии с методикой испытаний (см. документ «Программа и методика испытаний»).

# Дополнительные условия

Данное техническое задание может уточняться и дополняться в установленном порядке.

# ПРИЛОЖЕНИЕ В

# ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего профессионального образования**

**«Московский государственный технический университет имени Н.Э. Баумана»**

**(МГТУ им. Н.Э.Баумана)**

Утверждаю Согласовано

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_»\_\_\_\_\_\_\_\_\_\_2021 г. «\_\_\_»\_\_\_\_\_\_\_\_\_\_2021 г.

**«Мультимодальное обучение по тексту и изображениям. Создание описаний по изображениям»**

Программа и методика испытаний

(вид документа)

писчая бумага

(вид носителя)

6

(количество листов)

Исполнитель:

студент группы ИУ5-84Б

Сафин Р. Р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_2021 г.

Москва, 2021 г.

**СОДЕРЖАНИЕ**

[1. Наименование 81](#_Toc72176965)

[2. Цель испытаний 81](#_Toc72176966)

[3. Состав предъявляемой документации 81](#_Toc72176967)

[4. Технические требования 81](#_Toc72176968)

[4.1. Требования к составу программной документации 81](#_Toc72176969)

[4.2. Требования к составу технического обеспечения 81](#_Toc72176970)

[4.3. Требования к составу программного обеспечения 82](#_Toc72176971)

[5. Последовательность проведения испытаний 82](#_Toc72176972)

[6. Результат испытаний 84](#_Toc72176973)

# 

# Наименование

Объектом испытаний является модель мультимодального обучения по тексту и изображениям, а именно создание описаний по изображениям. Краткое обозначение объекта испытаний: «MIT» (Multimodal Image-to-Text)

# Цель испытаний

Цель испытаний состоит в проверке работоспособности модели мультимодального обучения по тексту и изображениям, а именно создания описаний по изображениям и её соответствия требованиям, описанным в техническом задании.

# Состав предъявляемой документации

Перед проведением испытаний предъявляются следующие документы:

1. Техническое задание;
2. Программа и методика испытаний.

# Технические требования

## 4.1. Требования к составу программной документации

Состав программной документации должен удовлетворять требованиям документа «Техническое задание».

Программная документация должна быть оформлена в соответствии с ГОСТ и ЕСПД по составлению и оформлению документов на программное изделие.

## 4.2. Требования к составу технического обеспечения

Для работы данного программного продукта системы требуется устройство с установленным браузером и выходом в интернет.

Для корректной работы данного программного продукта требуется следующее минимальное аппаратное обеспечение для серверной части:

* 64 битный процессор, ГГц – 2.3;
* оперативная память объемом 4 Гб;
* жесткий диск объемом 50 Гб;
* сетевая карта с выходом в сеть Интернет.

## 4.3. Требования к составу программного обеспечения

Для работы системы на компьютере, ноутбуке пользователя требуется:

4.3.1. Операционная система не регламентируется.

4.3.2. Один из перечисленных веб-браузеров:

* Google Chrome 40.0 и выше.
* Mozilla Firefox 40.0 и выше.
* Safari 9.0 и выше.
* Яндекс.Браузер 20.0 и выше

4.3.4. Для работы серверной части требуется:

* Интерпретатор языка Python версии 3.7.
* Установленные библиотеки PyTorch
* Установленная библиотека Streamlit

# Последовательность проведения испытаний

Последовательность испытаний приведена в следующей таблице.

Таблица 1. Последовательность испытаний модели «MIT».

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№** | **Пункт ТЗ** | **Проверяемая функция** | **Действие пользователя** | **Ожидаемый результат** |
| 1 | 6.1.1 | Доступность запущенного приложения локально. | Открыть в браузере выведенную ссылку с портом 8051. | Страница успешно откроется. |
| 2 | 6.1.2 | Загрузка нового изображения в приложение. | В модуле загрузки изображения нажать на кнопку «Browse files» и выбрать необходимое изображение. | Изображение успешно загружено в систему, выводится имя загруженного файла. |
| 3 | 6.1.2 | Загрузка нового изображения в приложение. | В модуль загрузки изображения перетащить необходимое изображение. | Изображение успешно загружено в систему, выводится имя загруженного файла. |
| 3 | 6.1.3 | Сохранение и отправка изображения в мультимодальную модель. | Нажать на кнопку отправить. | Изображение будет сохранено и направлено в модель, для создания текстового описания. |
| 4 | 6.1.4 | Вывод на экран загруженного в систему изображения. | Отправить изображение в систему. | Вывод загруженного в систему изображения на экран. |
| 5 | 6.1.5 | Вывод текстового описания загруженного изображения. | Загрузить изображение «showtest1.jpg» и нажать на кнопку «Отправить» | Вывод текстового описания изображения: «Человек в костюме и галстуке стоя перед зданием.» |
| 6 | 6.1.5 | Вывод текстового описания загруженного изображения. | Загрузить изображение «showtest2.jpg» и нажать на кнопку «Отправить» | Вывод текстового описания изображения: «Группа жирафов, стоящих рядом друг с другом.» |
| 7 | 6.1.5 | Вывод текстового описания загруженного изображения. | Загрузить изображение «showtest3.jpg» и нажать на кнопку «Отправить» | Вывод текстового описания изображения: «Жираф, стоящий в поле рядом с деревом.» |

# Результат испытаний

Основой испытаний является демонстрация работы компонентов системы и их работы в совокупности.

Испытание считается пройденным успешно, если в процессе демонстрации все действия прошли успешно и результат соответствовал ожидаемому с учетом используемых данных.