

ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Программный проект на тему:

SEO-оптимизация карточки товара на маркетплейсе

Выполнили студенты:

группы МОВС 2022
Бузаева Софья Михайловна
Куликов Дмитрий Алексеевич

Принял руководитель ВКР:

Кантонистова Елена Олеговна
Доцент факультета компьютерных наук НИУ ВШЭ

Соруководитель:

Хажгериев Мурат Анзорович
Консультант стартапов по построению
инфраструктуры для Large Language models

Москва, 2024

Содержание

Аннотация	4
Введение	9
1 Цель и задачи работы	10
2 Постановка задачи	11
3 Обзор существующих решений	12
3.1 TurboText.Pro	12
3.2 Gerwin	13
3.3 CopyMonkey	14
3.4 Сравнительный анализ существующих решений	15
4 Данные	16
4.1 Этап 1	18
4.2 Этап 2	20
5 Теоретические основы	26
5.1 Задача многоклассовой классификации изображений	26
5.2 Обзор классификаторов изображения	27
5.2.1 ResNet	27
5.2.2 MobileNet	28
5.2.3 EfficientNet	30
5.2.4 Vision Transformers	32
5.3 Используемые метрики в задаче классификации	34
5.4 Задача создания подписей к изображениям	35
5.5 Обзор моделей для генерации текста	39
5.5.1 Рекуррентные нейронные сети	39
5.5.2 Модели генеративного предварительно обученного трансформера	43
5.6 Используемые метрики в задаче создания подписей к изображениям	45
5.7 Вывод	45
6 Проектирование архитектуры программного продукта (в процессе)	47
7 Направление дальнейшего развития	49

Аннотация

Выпускная квалификационная работа посвящена разработке эффективного и простого в использовании приложения, которое позволит пользователям легко и быстро подбирать категорию товара для маркетплейса по его фотографии, а также составлять к нему SEO-описание. В работе проведен анализ существующих решений, рассмотрены различные архитектуры нейронных сетей, описаны задачи классификации и создания подписей к изображениям, разработаны решения для реализации моделей нейронной сети для SEO-оптимизации карточки товара, учитывающие специфику данных. Особое внимание уделено использованным метрикам и методам сбора данных. Результатом работы является разработанный сервис, в основе которого реализована нейронная сеть, способная классифицировать товары на маркетплейсе на основе их фотографий и генерировать соответствующие к ним описания. Дальнейшие исследования в этой области могут включать улучшения качества результатов, а также расширение функциональности за счет генерации описания с помощью ключевых слов.

Abstract

The final qualification work is devoted to the development of an effective and easy-to-use application that will allow users to easily and quickly select a product category for a marketplace based on its photo, as well as create an SEO description for it. The paper analyzes existing solutions, examines various architectures of neural networks, describes the tasks of classifying and creating captions to images, and develops solutions for implementing neural network models for SEO optimization of product cards, taking into account the specifics of the data. Special attention is paid to the metrics and data collection methods used. The result of the work is a developed service based on a neural network capable of classifying products on the marketplace based on their photos and generating descriptions corresponding to them. Further research in this area may include improvements in the quality of results, as well as expanding functionality by generating a description using keywords.

Ключевые слова

Маркетплейс (от англ. *marketplace*; электронная торговая площадка) — платформа электронной коммерции, интернет-магазин электронной торговли, предоставляющий информацию о продукте или услуге третьих лиц.

SEO-оптимизация карточек товара — это процесс улучшения информации о товаре на маркетплейсе, чтобы она была более привлекательной и доступной для поисковых систем и пользователей. Цель состоит в том, чтобы повысить видимость товара в результатах поиска, привлечь больше потенциальных покупателей и увеличить конверсию.

Искусственный интеллект (ИИ) (от англ. *Artificial Intelligence*, AI) — свойство интеллектуальных компьютерных систем, обладающих возможностями выполнять творческие задачи, которые считаются прерогативой человека [ai].

Машинное обучение (от англ. *Machine Learning*, ML) — область искусственного интеллекта, изучающий различные способы построения обучающихся алгоритмов. Среди множества парадигм и подходов в машинном обучении выделяются нейронные сети [ml].

Компьютерное зрение (от англ. *Computer Vision*, CV) — область искусственного интеллекта, которое занимается задачами, связанными с анализом изображений и видео [cv].

Обработка естественного языка (от англ. *Natural Language Processing*, NLP) — это направление в машинном обучении, посвящённое распознаванию, генерации и обработке устной и письменной человеческой речи [nlp].

Нейронная сеть (от англ. *Neural Network*) — математическая модель, а также ее программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей, используемая для решения задач ИИ [neural].

Сокращения названий архитектур нейронных сетей:

- сверточная нейронная сеть (от англ. Convolutional Neural Network, CNN);
- рекуррентная нейронная сеть (от англ. Recurrent Neural Network, RNN);
- рекуррентная нейронная сеть с долгой краткосрочной памятью (от англ. Long Short-Term Memory, LSTM);
- управляемый рекуррентный блок (от англ. Gated Recurrent Unit, GRU);
- генеративный предварительно обученный трансформер, (от англ. Generative Pre-trained Transformer, GPT).

Введение

В современном мире маркетплейсы стали неотъемлемой частью электронной коммерции, предоставляя платформы для продажи товаров и услуг различным производителям и ритейлерам. В 2023 году маркетплейсы продолжили быть главной движущей силой российской онлайн-торговли. Рост объема трат на маркетплейсах в 1,5 раза по сравнению с предыдущим годом свидетельствует о том, что интерес потребителей к онлайн-покупкам только укрепляется. На это влияют общерыночные факторы: продолжают развиваться альтернативные каналы поставок продукции ушедших брендов, улучшаются условия доставки, повышается удобство пользования платформами, расширяется сеть пунктов выдачи.

Согласно исследованиям «Tinkoff Ecommerce» [[tinkoff-research](#)], количество транзакций на маркетплейсах за год выросло на 63% (см. рисунок 0.1). Лидерами по росту количества покупок стали «Мегамаркет» (число транзакций выросло в 4,3 раза), «Wildberries» (в 2 раза) и «Ozon» (в 1,6 раза).



Рис. 0.1: Динамика покупок на маркетплейсах в регионах [[tinkoff-research](#)].

Появился тренд на рост популярности маркетплейсов в российских регионах. В 2023 году жители российских городов стали значительно активнее совершать покупки на маркетплейсах: выросло как количество транзакций на онлайн-площадках, так и их сумма (см. рисунок 0.2). По количеству совершенных транзакций особенно заметен рост в таких городах, как Омск (+91%), Красноярск (+88%), Новосибирск (+79%), Челябинск (+79%) и Волгоград (+75%). В Москве зафиксирован наименьший прирост числа транзакций (+41%). Увеличение интереса жителей регионов к маркетплейсам объясняется рядом причин: расширением географии присутствия площадок, развитием сетей пунктов выдачи заказов и

логистических сервисов, улучшением условий доставки.



Рис. 0.2: Динамика покупок на маркетплейсах в регионах [tinkoff-research].

Вместе с тем выросло количество селлеров на 8%. Рынок становится более зрелым: место неопытных продавцов занимают более профессиональные. Они ведут бизнес более уверенно, укрепляют свои позиции на площадках и торгуют на нескольких платформах одновременно. Количество селлеров, ведущих торговлю на двух и более маркетплейсах, за год увеличилось на 17%. Самой привлекательной платформой для старта бизнеса является «Wildberries»: 63% продавцов в конце 2023 года выбирали ее в качестве первой площадки (см. рисунок 0.3).



Рис. 0.3: Популярность маркетплейсов за 2023 год [tinkoff-research].

С увеличением конкуренции продавцы все чаще обращаются к системам, которые могут помочь им в продажах, а также автоматизировать процесс работы. Одним из ключевых

факторов успешной продажи становится эффективная SEO-оптимизация карточек товаров. Подбор наиболее подходящей категории и создание продаваемого описания, содержащего ключевые слова, позволяет улучшить видимость товаров в результатах поиска как на самом маркетплейсе, так и в поисковых системах, что напрямую влияет на увеличение продаж.

На данный момент существуют несколько сервисов (например, TurboTextPro, Gerwin, CopyMonkey, подробнее можно ознакомиться в пункте 3), позволяющих сгенерировать описание товаров по характеристикам, ключевым словам или фотографии. Однако, качество сгенерированных описаний не всегда позволяют использовать их в системах автономного управления. В настоящее время российский рынок не предлагает специальных технологий и решений для подбора наиболее подходящей категории товара на маркетплейсе. Таким образом, задача создания качественного инструмента для эффективной SEO-оптимизации карточек товаров является актуальной.

1 Цель и задачи работы

Цель данной работы — разработать сервис, в основе которого будет реализована модель нейронной сети, способная классифицировать товары на маркетплейсе на основе их фотографий и генерировать соответствующие к ним описания.

Для достижения поставленной цели необходимо решить следующие задачи:

- подготовить набор данных, содержащий изображения товаров и соответствующие им категории и текстовые описания;
- провести исследование текущих архитектур нейронных сетей, используемых для классификации изображений и генерации текста, и на основе этого исследования выбрать наиболее подходящую архитектуру или их комбинацию для решения поставленной задачи;
- обучить выбранную нейронную сеть на подготовленных данных, оптимизировать и настроить параметры модели для повышения её производительности и качества результатов, а затем оценить эффективность реализованной архитектуры нейронной сети;
- интегрировать модель в программное обеспечение.

2 Постановка задачи

В соответствии с заданием на выпускную квалификационную работу необходимо разработать программное обеспечение, на вход которого подаются фотографии товара. Выходом являются наиболее подходящая категория товара для маркетплейса и его SEO-описание.

Допущения:

- рассматривается только маркетплейс «Wildberries»;
- глубина подбора категории — второй уровень вложенности.

Постановка задачи в виде IDEF0-диаграммы представлена на рисунке 2.1.

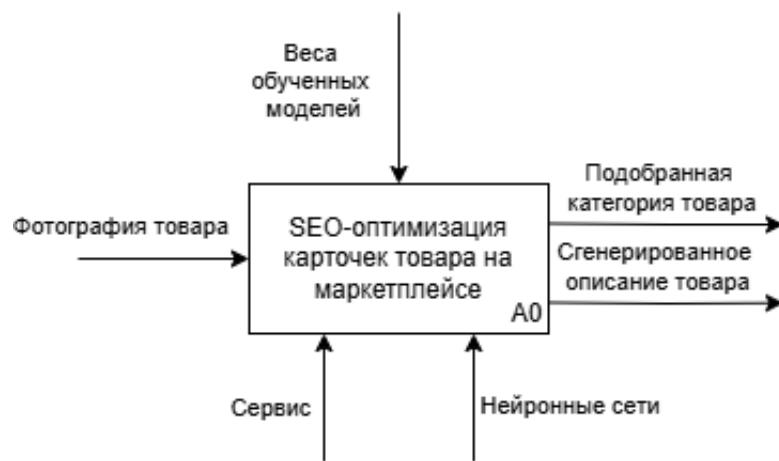


Рис. 2.1: Постановка задачи.

Для достижения целей и задач проекта требуется решить задачу многоклассовой классификации изображений (от англ. multiclass image classification) для подбора наиболее подходящей категории по фотографии товара, а также решить задачу создания подписей к изображениям (от англ. image captioning). Для успешной реализации проекта необходимо выбрать подходящие модели и определить метрики, которые будут использоваться для оценки их эффективности.

3 Обзор существующих решений

На данный момент существуют несколько сервисов, позволяющих сгенерировать описание товаров по характеристикам, ключевым словам или фотографии. Однако нет сервисов для подбора наиболее подходящей категории товара на маркетплейсе по фотографии. Для продавцов было бы крайне полезно иметь эти функциональности одновременно. Это значительно упростило бы процесс подготовки и публикации товаров, а также повысило эффективность и точность их работы.

3.1 TurboText.Pro

TurboText.Pro — создает описания по фотографии или характеристикам товара [turbotextpro.ru]. Во втором случае текст будет более высокого качества, так как ИИ напишет описание по заданным параметрам. Нейросеть создает три описания для одного товара. Можно выбрать самый лучший вариант и разместить его в товарной карточке, или использовать все на разных площадках. А еще можно заказать улучшение текста или SEO-оптимизацию для вывода товара в топ на «Wildberries» у опытных копирайтеров.

Плюсы данного сервисы:

- высокая скорость генерации (1-2 секунды);
- развернутое описание товара;
- все функции доступны бесплатно;
- можно сгенерировать описание товара по фотографии или характеристикам;
- есть возможность заказать SEO-оптимизацию описания товара.

Минусы:

- генерация только на русском языке;
- качество описания по фото ниже, чем описания по характеристикам товара;
- есть ограничение при генерации описания по характеристикам — от 30 символов.

На рисунке 3.1 представлен интерфейс данного сервиса.

VESVEN 400 г/кв.м, 104 см, Молния

Вариант 1 17 октября, 13:31

Зимнее пальто прямого силуэта. Выполнено из плотной плащевой ткани с водоотталкивающей пропиткой, из подкладке. Модель имеет два кармана на молнии и один нагрудный карман с клапаном на кнопках. Рукава втачные длинные на манжетах. Воротник-стойка. Наполнитель: полизифирное волокно, плотностью 400 г/кв.м. рассчитано на температурный режим до -20 градусов. Ткань верха: 100% полиэстер. Утеплитель: 400 г/км. Рекомендованный температурный режим от 0 до -30 градусов.

[Скопировать описание](#)

Вариант 2 17 октября, 13:31

Теплая куртка-пуховик выполнена из водоотталкивающего материала. Модель застегивается на молнию с двумя бегунками, также имеется один карман в боковом шве. Воротник стойка, два прорезных кармана на молнии. Рукава втачные со шлицами, с трикотажными манжетами и регулировкой объема по низу изделия. Низ куртки по линии талии регулируется кулиской. Подкладка капюшона, полочек и спинки выполнена из хлопковой ткани. Ткань верха — плащевая ткань с водоотталкивающим покрытием.

[Скопировать описание](#)

Вариант 3 17 октября, 13:31

Зимняя верхняя одежда VESVEN изготовлена из высококачественной водоотталкивающей ткани, которая обеспечивает 100% защиту от влаги и ветра. Утеплитель нового поколения (400 г/кв.м) сохраняет тепло до -30С. Удобный крой позволяет надеть куртку на любой тип фигуры. Подкладка изделия выполнена из мягкого трикотажа. На куртке имеются боковые карманы, а также один внутренний карман. В рукавах предусмотрена резинка для плотного прилегания к телу.

[Скопировать описание](#)

Описание для товара

Бренд VESVEN

Категории Верхняя одежда, Пуховики

Характеристики

Плотность синтетического утеплителя	400 г/кв.м
Длина изделия по спинке	104 см
Вид застежки	Молния
Сезон	зима

[Изменить продукт](#)

Улучшите описание товара

Закажите SEO-оптимизацию описания товара или его доработку на бирже копирайтинга TurboText. Десятки опытных копирайтеров готовы приступить к заказу прямо сейчас.

[Улучшить описание](#)

Рис. 3.1: Сервис TurboText.Pro.

3.2 Gerwin

Gerwin — создает описания для товара по ключевым словам [gerwin].

Плюсы данного сервиса:

- высокая скорость генерации (1-2 секунды);
- есть варианты генерации на разных языках;
- разные описания к одному товару на выбор;
- можно протестировать сервис бесплатно по промокоду.

Минусы:

- ограничение по количеству ключевых слов — не больше 3-х;
- из-за ограничений исходных данных нельзя сделать более развернутое описание товара;
- нет возможности сгенерировать описание товара по фотографии;
- чтобы протестировать сервис бесплатно, нужно «добыть» промокод через Telegram-бота. Время ожидания — в течение 8 часов.

На рисунке 3.2 представлен интерфейс данного сервиса.

The screenshot shows a user interface for generating product descriptions. On the left, there's a logo of a blue book and a title 'Описание товара для Wildberries'. Below it, a text input field contains 'VESVEN Пуховик женский зимний, куртка зимняя. нейлон 100%'. A section for 'Ключевые слова' has a single entry: 'Плотность синтетического утеплителя 400 г/кв.м'. Under 'Язык', 'Русский' is selected. On the right, three examples of generated descriptions are shown in cards:

- Несколько секунд назад**: 'Этот женский зимний пуховик изготовлен из 100% нейлона и имеет плотность синтетического утеплителя 400 г/кв.м. Он застегивается на молнию, а длина спины составляет 104 см.'
- 2 минуты назад**: 'Женский зимний пуховик vesven изготовлен из 100% нейлона и тканей утеплителем плотностью 400 г/кв.м. Изделие имеет высоту 104 см по спине и застегивается на молнию.'
- 3 минуты назад**: 'Это женская зимняя пуховая куртка, изготовленная из 100% нейлона. Это отличный выбор для встречи в холодные зимние месяцы.'

Рис. 3.2: Сервис Gerwin.

3.3 CopyMonkey

CopyMonkey — создает описания для товара по его характеристикам и ключевым словам [copymonkey].

Плюсы данного сервисы:

- высокая скорость генерации (1-2 секунды);
- есть бесплатные попытки ежедневно;
- разные описания к одному товару на выбор;
- можно протестировать сервис бесплатно по промокоду.

Минусы:

- низкое качество текста, нейросеть просто сгруппировала характеристики;
- есть логические несостыковки в тексте;
- один вариант описания товара без возможности выбора.

На рисунке 3.3 представлен интерфейс данного сервиса.

The screenshot shows the CopyMonkey service interface. At the top, there's a logo of a person writing on a clipboard and the title "Описание товара" (Product Description). Below it, a subtitle says "Генерация описаний товаров для интернет-магазина и маркетплейсов".

Название товара * (Product Name) field contains "Пуховик женский зимний" with a character count of 22/350.

Результат (Result) tab is selected, showing the generated text: "Пуховик женский зимний – это плотный синтетический утеплитель, изделие длиной по спинке 104 см. Застежка – молния. Длина одежды – ниже колена. Комплектация – пояс. Пол – Женский. Сезон – зима." There is also an "История" (History) tab.

Детали товара (Product Details) section lists: Плотность синтетического утеплителя 400 г/кв.м, Длина изделия по спинке 104 см, Вид застежки молния, Длина одежды ниже колена, Комплектация пояс, Пол Женский, Сезон зима. Character count is 163/2000.

Ключевые слова (Keywords) section has a field with "Например, свитер в подарок" and a button to add more. It also shows "Пуховик" and "Зимняя куртка" as selected keywords. A note says "У вас есть 3 бесплатных генераций в день. Осталось 2".

At the bottom, there are "Очистить форму" (Clear form) and "Сгенерировать" (Generate) buttons.

Рис. 3.3: Сервис CopyMonkey.

3.4 Сравнительный анализ существующих решений

TurboText.Pro генерирует более развернутое описание товара, но есть пара неточностей, которые легко поправить, потратив на это не больше минуты. После описание можно размещать в карточке товара интернет-магазина или на маркетплейсе. CopyMonkey и Gerwin справляются со своей задачей хуже. Все три сервиса можно протестировать бесплатно и выбрать тот, который подходит для работы лучше всего.

Сравнительный анализ существующих решений может быть представлен в таблице 3.1.

Сервис	Стоимость	Генерация по фотографии	Качество генерации
TurboText.Pro	Бесплатно	+	Высокое
Gerwin	Бесплатно	—	Среднее
Copymonkey	Бесплатно 3 попытки в день	—	Низкое

Таблица 3.1: Сравнение существующих решений по генерации SEO-описания товара.

4 Данные

Для того чтобы датасет был полезным и эффективным для достижения поставленной цели и задач проекта, он должен удовлетворять следующим требованиям:

- данные должны быть достоверными и правильно отражать характеристики товаров;
- данные должны иметь четкую классификацию товаров по категориям и подкатегориям, соответствующих структуре категорий маркетплейсов в РФ (например, «Ozon» или «Wildberries»);
- данные должны быть структурированы в удобном для обработки формате, таком как CSV, JSON или в виде базы данных;
- данные должны включать фотографии товаров с разных ракурсов и в различных вариациях, а также иметь их текстовые описания;
- необходимо иметь достаточное количество данных для каждой категории товаров, чтобы обеспечить репрезентативность для анализа и обучения моделей;
- данные должны быть актуальными и регулярно обновляемыми для отражения текущих трендов и состояния рынка.

На данный момент существуют несколько датасетов для работы с товарами, взятых с интернет-магазинов электронной торговли:

- Fashion-MNIST — подходит для продуктовой категоризации. MNIST содержит почти 60 000 обучающих изображений и 10 000 тестовых изображений одежды и аксессуаров, разделенные на 10 категорий (Футболка, Брюки, Свитер, Платье, Пальто, Сандалии, Рубашка, Кроссовки, Сумка, Ботинки). Каждое изображение имеет размер 28x28 пикселей и представлено в оттенках серого. [**fashion_mnist**].
- Innerwear Data from Victoria's Secret and Others — содержит данные с 600 000+ товаров нижнего белья, извлеченного из популярных торговых объектов. Включает в себя описание продукта, цену, категорию и рейтинг [**victoria_secret_data**].
- eCommerce Item Data — набор данных, который содержит артикулы и связанные с ними описания продуктов из каталога продукции бренда верхней одежды. Подходит для рекомендательных систем [**eCommerce_item_data**].

- Fashion Products on Amazon — набор данных, созданный путем извлечения данных из Amazon. Он состоит примерно из 22 000 товаров на Amazon и включает в себя описание продукта, цену, категорию и рейтинг [amazon_data].

Данные для поставленной задачи собирались самостоятельно, поскольку в открытых источниках не имеется удовлетворяющего всем требованиям датасета. Был проведен анализ на возможность парсинга наиболее популярных маркетплейсов в РФ «Ozon» и «Wildberries».

Анализ парсинга «Ozon»:

- Сильная защита, частая блокировка пользователей, смена userAgent не всегда помогает.
- На странице много динамического контента и ленивой подгрузки, что создает трудности при парсинге динамически подгружаемых категорий.
- Огромное количество подкатегорий товара.
- Большая вариативность описания, нет единного шаблона для парсинга. Так например, когда-то вместо текстов могут быть просто картинки или описание товара сопряжено с картинкой.
- Динамическая подгрузка отзывов, которая усложняет скачивание фотографий из них.

Анализ парсинга «Wildberries»:

- Слабая защита, не требуется смена userAgent.
- Статический контент для парсинга категорий и подкатегорий.
- При парсинге товаров существует их ленивая подгрузка: для одной подкатегории подгружается 15 товаров, чтобы подгрузить следующие нужно «прокрутить» экран вниз.
- Единное оформление карточки товара, упрощает парсинг товара: его описание, характеристики, фотографии из сео и из отзывов.

Принято решения писать ВКР для продавцов, использующий «Wildberries», так как им чаще всего пользуются и он легче подается парсингу. Данные собирались согласно особенностям структуры этого маркетплейса. Для удобства введем некоторые термины, которыми будем оперировать далее:

- **Карточка товара** – это страница продукта на маркетплейсе, где размещена информация о товаре, фотографии, описание цены и кнопка «Купить»

- **Конечная категория** – это категория, на которых располагаются карточки товаров
- **Материнская категория** – это категория, которая содержит конечные и материнские категории и на которой не располагаются карточки товаров

Каталог «Wildberries» разделен на категории, в которых размещены карточки товаров одного типа. Категории выстроены по принципу дерева. Есть основные «широкие» категории, такие как «Женщинам», «Дом», «Продукты», которые объединяют внутри себя более мелкие подкатегории. Например, в разделе «Дом» имеются подкатегории «Ванная», «Кухня», «Спальня» и тд, которые в свою очередь могут подразделяться на еще более маленькие подкатегории.

Требуемые данные располагались на товарных карточках, в которые можно попасть только зная конечную категорию товара. Поэтому было принято решение разделить сбор данных на 2 этапа. На первом этапе был произведен сбор всех имеющихся на «Wildberries» конечных категорий. На втором – сбор необходимой информации с карточек товаров.

4.1 Этап 1

«Wildberries» предлагает 22 основные категории (см. рисунок 4.1), из которых одна является конечной категорией. Данные категории в дальнейшем будем называть категориями первой вложенности. Их подкатегории, соответственно, будут называться категориями второй вложенности. И так далее, спускаясь все ниже по дереву категорий. Экспериментальным путем была выявлена максимальная глубина вложенности – 5.

Для правильного формирования таргета для классификации при сохранении ссылки на конечную категорию нужно было учитывать весь путь по дереву категорий, начиная с первой вложенности. Решением данной задачи стало создание таблицы, где отражалось какие категории было предшествовавшими конкретной конечной категории (см. таблицу 4.1). При отсутствии более глубокой вложенности на месте данных категорий ставились «NaN». Таким образом, было собрано 1668 конечных категорий.

Таблица 4.1: Фрагмент таблицы, полученной после первого этапа сбора данных.

	category1	category2	category3	category4	category5
437	Дом	Предметы интерьера	Фоторамки и фотоальбомы	Фотоальбомы	NaN
438	Дом	Предметы интерьера	Картинами и постерами	Рамы для постеров	NaN
439	Дом	Предметы интерьера	Картинами и постерами	Постеры	Детская тематика
440	Дом	Предметы интерьера	Картинами и постерами	Картинами	Арт и абстракция
441	Дом	Предметы интерьера	Картинами и постерами	Постеры	Фэнтези

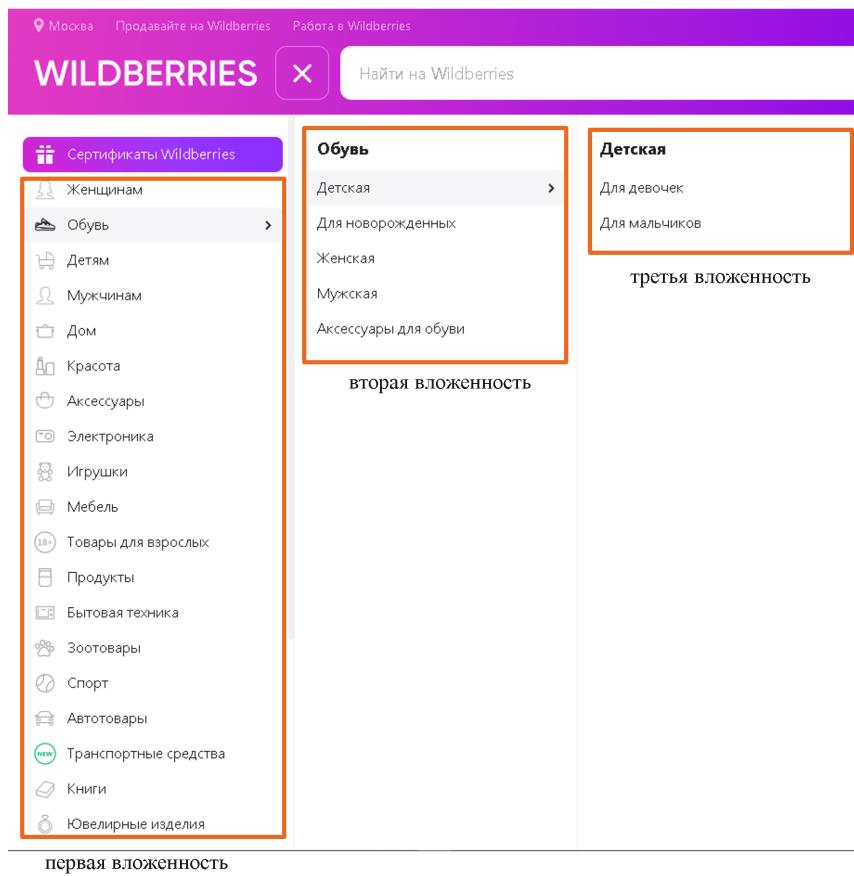


Рис. 4.1: Структура каталога «Wildberries» на примере категории «Обувь».

Составление данной таблицы производилось посредством парсинга данных с сайта «Wildberries» через Python с использованием библиотек selenium и BeautifulSoup. Блокировок со стороны маркетплейса замечено не было. Особенность и неудобством парсинга была динамическая подгрузка страниц, которая вынуждала выдерживать паузы в несколько секунд для удовлетворяющей загрузке страницы. Данное обстоятельство привело к значительному увеличению времени парсинга данных.

При анализе собранной таблицы были выявлены некоторые особенности категориальной политики «Wildberries». Во-первых, категории у данного маркетплейса не фиксированы. Например, было отмечено, что часть категорий активно перемещается из раздела в раздел, какие-то категории могут пропадать, также могут появляться новые категории. Данные, собранные в текущем датасете, актуальны на конец января 2024 года. Однако для поддержания списка категорий в актуальном состоянии необходимы механизмы регулярного обновления данных. Во-вторых, на маркетплейсе имеются конечные категории, ссылающиеся на одни и те же url страницы. Подобные категории будут называться дублирующими. Подобные дублики могли иметь разное происхождение: особенности маркетинга и неудачное время парсинга, выпавшее на перемещение категорий. С точки зрения мар-

кетинга подобные дублирования оправданы, поскольку потенциальный покупатели могут по своим соображениям относить одни и те же товары к разным категориям. Для примера, категория «Коврики» находилась в разделе «Автотовары_Коврики» и «Электроника_Автоэлектроника&и&навигация_Коврики». Для корректной работы модели была написана отдельная процедура удаления подобных дублирующих категорий. Выбор, какой из дубликатов оставлять, производился вручную. Всего было найдено 69 дублирующих ссылок, которые могли встречаться 2 и более раза. Таким образом, после удаления в таблице осталось 1580 категорий.

Далее можно было переходить ко 2му этапу.

4.2 Этап 2

Второй этап сбора данных заключался в прохождении по собранному ранее списку конечных категорий и сбора из каждой из них информации с карточек товаров. Было принято решение брать по 20 товаров из каждой конечной категории. Из каждой карточки товара сохранялось первое фотография от продавца, первая фотография из отзыва и описание товара (см. рисунок 4.2). Первая фотография от продавца бралась по причине ее обязательного присутствия в карточке товара, а также гарантированного качественного изображения товара на ней. Однако поскольку разрабатываемый сервис рассчитан на работу в большинстве случаев с фотографиями от пользователей, все дефекты, присущие любительским фотографиям могут иметь место быть. Поэтому для стабильности предсказаний классификационной модели, было решено подавать в нее также фотографии из отзывов, которые максимально близко будут похожи на фотографии, с которыми будет работать в дальнейшем сервис. Описания товара были нужны для задачи генерации текстовых описаний к изображениям. В итоге при полном наборе для каждой конечной категории имелось 40 фотографий (20 фотографий от продавцов и 20 фотографий из отзывов) и 20 описаний.

При выполнении данного этапа было несколько трудностей. Во-первых, стандартная на «Wildberries» динамическая загрузка страниц, увеличивающая время парсинга более чем в 3 раза. Приходилось делать паузы при открытии страницы с карточкой товара, при открытии описания, лежащее в отдельной вкладке, и пролистывание страницы вниз для подгрузки информации об отзывах. Примерное время парсинга данных, затраченное на второй этап, равнялось 2м неделям. Во-вторых, имели товары с меткой «18+», для которых требовалось дополнительное нажатие кнопки, подтверждающей достижение указанного возраста. В-третьих, некоторые поля в карточке товара заключали в себе картинки, которые вынуж-

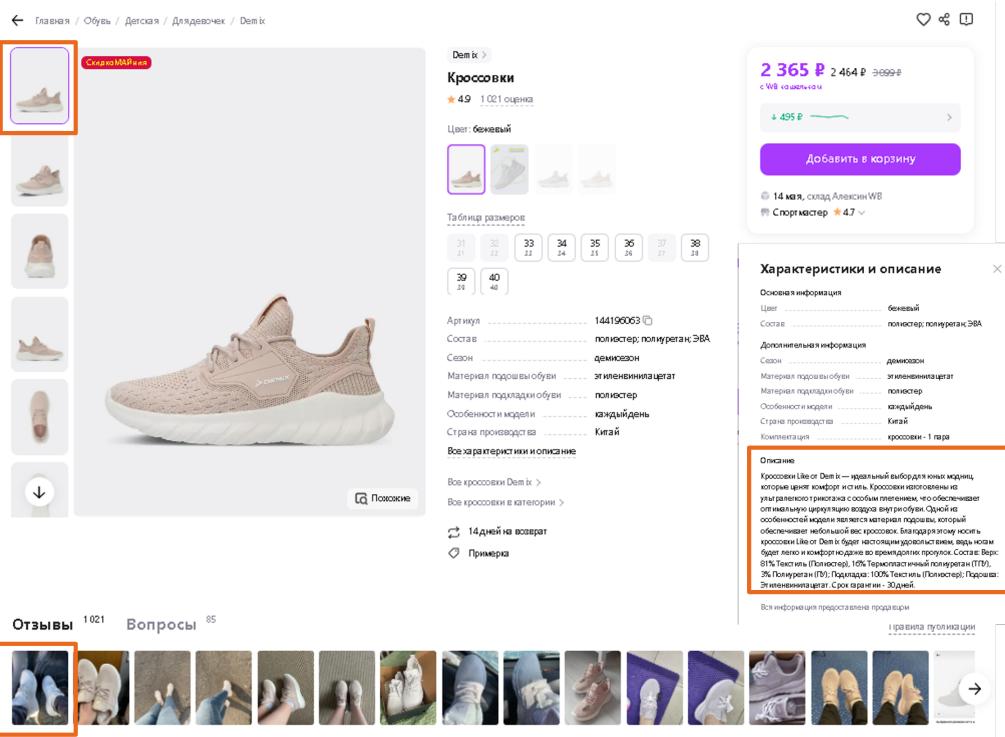


Рис. 4.2: Пример карточки товара на «Wildberries». Верхний левый прямоугольник – первая фотография от продавца. Нижний левый прямоугольник – первая фотография из отзыва. Правый прямоугольник – текстовое описание товара.

дали в определенных случаях дополнительно пролистывать страницу вниз. В-четвертых, для нажатия кнопки с целью получения описание к товару, выдвигалось требование расположение кнопки в зоне видимости экрана. Это приводило к еще более тонкой настройке пролистывания страницы, подобранной под конкретный размер экрана компьютера.

Для сохранения данных из карточек товара была придумана специальная структура с целью дальнейшего удобства использования в задаче классификации и генерации текста. Все товары, собранные из одной конечной категории, сохранялись в отдельную папку, содержащую следующие элементы:

- папку «card», куда складывались фотографии от продавцов
- папку «feedbacks», куда складывались фотографии из отзывов
- файл «descriptions.csv», где сохранялись описания к товарам

Название данной папки определялось посредство таблицы 1 и складывалось из всех материнских категорий, участвовавших в пути к конечной категории. Например, для конечной категории «Фотоальбомы» (см. таблицу 4.1) название папки было следующее: «Дом_Предметы» а для категории «Фэнтези» - «Дом_Предметы&интерьера_Картины&и&постеры_Постеры_Фэнтези»

Более подробно об использовании подобной структуры ранения данных будет описание в главе ___ в разделе ___.

Первичный анализ собранных данных выявил, что не у всех товаров имелись отзывы с фотографиями и описания. Описания имелись в 99.8% проценте случаев. В таблице 4.2 приведены некоторые статистические данные о собранных фотография от продавца и из отзыва. Можно заметить, что некоторые конечные категории были полностью без фотографий в отзывах. Однако, опираясь на перцентили, можно сделать вывод, что таких категорий было довольно мало. Касательно фотографий от продавцов можно сделать 2 вывода. Во-первых, есть категории, представленные менее чем 20ю товарами. Во-вторых, есть как минимум одна категория, в которой имеется только 1 товар. Подобные категории нас не устраивают, потому что далее будет производиться деление каждой категории на 2 части, и категории с одним товаром невозможно будет разделить.

Таблица 4.2: Описательная статистика по фотографиям от продавца (столбец «card») и фотографиям из отзыва (столбец «feedbacks»).

	card	feedbacks
count	1580	1580
mean	19.98	17.72
std	0.63	4.23
min	1	0
25%	20	18
50%	20	19
75%	20	20
max	20	20

Всего категорий, представленных менее 20 товарами, было выявлено 5 штук (см. таблицу 4.3). Из них представляли наибольший интерес __ и __, из-за чересчур малого количества товаров. Категорию с одним товаром было решено удалить. Таким образом, осталось 1579 конечных категорий, с которыми шла вся дальнейшая работа.

Таблица 4.3: Таблица с категориями, имеющими менее 20 товаров.

кол-во товаров	категория
18	Дом_Кухня_Кухонный&текстиль_Чехлы&для&ручек&холодильников
4	Дом_Освещение_Лифты&для&люстр
19	Мебель_Гардеробная&мебель_Ящики
1	Мебель_Офисная&мебель_Перегородки&офисные
19	Мебель_Офисная&мебель_Шкафы

При более детально рассмотрении собранных данных было замечено, что фотографии из отзывов довольно шумные (см. рисунок 4.3). Очень много одинаковых фотографий, фотографий, где не очень понятно, что изображено. Поэтому для дальнейшей работы ис-

пользовались только фотографии товара от продавца.



Рис. 4.3: Фотографии из отзывов в категории «Игрушки_Антистресс».

Далее интересно было рассмотреть количество собранных данных в разрезе категорий первой вложенности (см. рисунок 4.4). Из круговой диаграммы можно заметить, что категории довольно несбалансированный. Например, категория «Дом» вмещает в себя порядка 6000 пример. В то время как в категории «Ювелирные&изделия» только 320 примеров.

Подобную картину можно наблюдать в категориях всех вложений (см. приложение А).

После появление базового понимания данных надо было приступить к их детальному изучению. Как правило, парсинг большого количества данных, тем более с постоянно меняющихся маркетплейсов, не проходит идеально. В сохраненных данных могут быть оплошности, которые помешают грамотно решать задачу классификации и генерации текста. Приведем некоторые примеры, выявленных особенностей, требуемых принятие решений с нашей стороны:

- Встречаются категории, которые с точки зрения категорийного менеджмента, имеют место быть. Одна для классификационной модели машинного обучения такие категории будут не осиливаемыми. Например, в разделе «Женщинам» есть подкатегории «Женщинам_Белье» и «Женщинам_Большие&размеры_Белье». Если детально изучить фотографии, которые в этих категориях присутствуют, можно сделать вывод, что особых различий между ними нет. Единственное, что было подмечено, что на очень немногих фотографиях стоит надпись «4XL» или что-то подобное, указывающее, что у

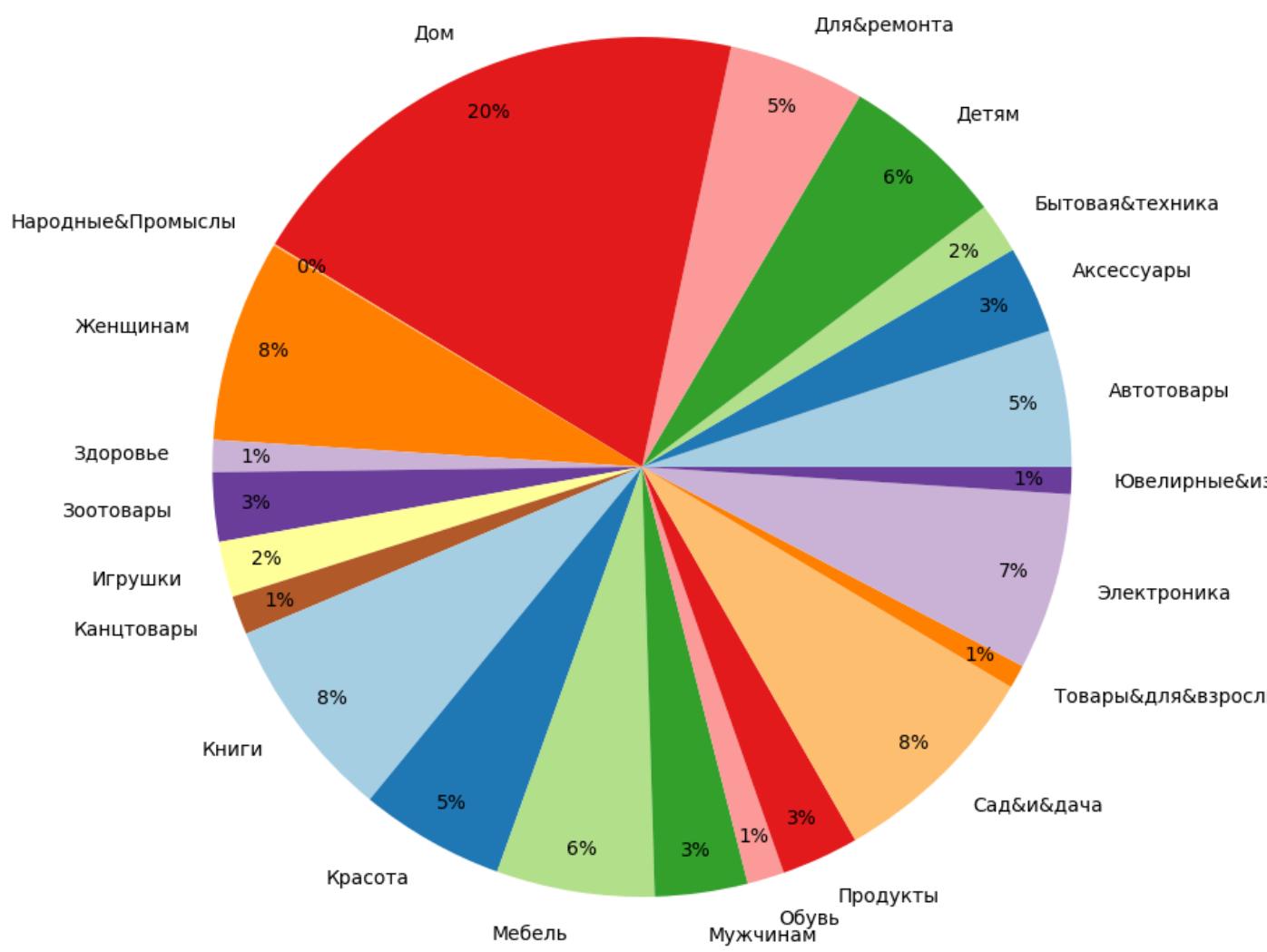


Рис. 4.4: Распределение данных по категориям первой вложенности.

данного товара имеются большие размера. Более того, в обеих категориях присутствуют одинаковые товары.

- В данных попадались «мусорные» категории. Предположительно, подобное возникало из-за изменения url ссылок на категории со стороны «Wildberries». В подобных категориях находились товары, собранные случайным образом из всевозможных категорий с маркетплейса.
- Распределение товаров по категориям не очень четкая задача. В связи с этим встречались одинаковые товары, находящиеся в разных категориях. Например, одна и та же продукт мог находиться в категориях «Зоотовары_Груминг&и&уход», «Зоотовары_Для&кошек_Груминг&и&уход» и «Зоотовары_Для&собак_Груминг&и&уход».

- В части материнских категорий встречались разделы «Подарки» (например, материнские категории «Мужчинам_Подарки&мужчинам» и «Женщинам_Подарки&женщинам»), куда были собраны товары из совершенно разных категорий, таких как «Аксессуары», «Дом», «Продукты» и тд.
- Поскольку при парсинге из каждой категории брались первые 20 товаров, появляется неконтролируемый фактор того, какие товары стоят вначале. Как правило, пользователи смотрят только на первые товары в выдаче. Поэтому на маркетплейсах существует множество механизмов и правил отбора товаров, которые будут показаны пользователю в начале. В нашем случае было замечено, что некоторые категории стали более шумными из-за сезонных товаров. Например, в категории «Зоотовары_Фермерство» были найдены пасхальные яйца.
- Бывали категории, которые по смыслу имели место быть как отдельные категории, однако в них были собраны не совсем подходящие товары. Например, в категории «Мужчинам_Религиозная_Православие» находились обычные рубашки и штаны, часть из которых присутствовала также в категории «Мужчинам_Рубашки» и «Мужчинам_Брюки».

Приведенные особенности сохраненных данных требовали ручной очистки датасета. Необходимо было применять следующие действия: полное удаление категории, удаление конкретной фотографии из отзыва, удаление товара полностью (фотографию от продавца, из отзыва и описание к нему) и произведение полного переноса товара (фотографию от продавца, из отзыва и описание к нему) из одной категории в другую. Для удобства и быстроты данной процедуры были написаны функции, позволяющие механизмами Python вносить изменения в собранные данные.

По окончании данной процедуры было подмечено, что категории требовали очистки в разной степени. Какие-то категории, как «Зоотовары», требовали практически полного переформирования. Другие категории обходились легкой очисткой, например «Продукты». Некоторые категории совсем не требовали вмешательства. Таким образом, финальный датасет стал состоять из 1459 конечных категорий (см. таблицу 4.3).

Таблица 4.4: Описательная статистика по фотографиям от продавца после очистки собранных данных. Примечание: статистика по фотографиям из отзывов не приведена, поскольку, как упоминалось ранее, решено было в дальнейшем работать только с фотографиями от продавца.

	card
count	1459
mean	20.09
std	2.32
min	4
25%	20
50%	20
75%	20
max	54

5 Теоретические основы

5.1 Задача многоклассовой классификации изображений

Задача многоклассовой классификации изображений заключается в том, чтобы отнести каждое изображение из набора данных к одной из заранее определенных категорий или классов. Для решения задачи необходимо использовать разнообразный набор изображений, каждый из которых должен быть помечен (аннотирован) соответствующим классом.

Для многоклассовой классификации часто используют сверточные нейронные сети, такие как ResNet, MobileNet или EfficientNet (подробнее см. в разделе 5.2). Для оценки качества модели обычно применяются следующие метрики: Accuracy, Precision, Recall и F1-score (подробнее см. в разделе 5.3).

Для повышения производительности модели можно использовать различные методы, например:

- тонкая настройка гиперпараметров — корректировка параметров обучения, таких как скорость обучения, размер батча и т.д.
- дополнительная аугментация данных — применение различных преобразований к изображениям, таких как повороты, сдвиги, масштабирование, для увеличения разнообразия обучающих данных.
- использование предобученных моделей — начинать обучение с модели, предварительно обученной на большом наборе данных, таком как ImageNet.

5.2 Обзор классификаторов изображения

В этом разделе будет дан обзор существующих классификаторов изображений, их архитектурные особенности, а также преимущества и недостатки.

5.2.1 ResNet

ResNet (Residual Neural Network) — это семейство моделей нейронных сетей, разработанные для решения проблемы затухания градиента [resnet]. Модели получили широкое признание и стали основой для многих современных архитектур благодаря своей способности эффективно тренировать очень глубокие нейронные сети.

Ключевой инновацией ResNet является использование остаточных блоков (от англ. residual blocks), которые помогают бороться с проблемой затухания градиентов в очень глубоких сетях. Основная идея заключается в том, чтобы добавить прямые связи (от англ. skip connections) через несколько слоев, что позволяет градиентам легче проходить через сеть.

Остаточный блок — это компонент архитектуры ResNet, который содержит «обходную связь идентичности», обходящую один или большее количество слоев. Остаточный блок состоит из двух или трех сверточных слоев с прямой связью, которая пропускает входы блока напрямую к его выходу. Это помогает сохранять информацию от предыдущих слоев и упрощает обучение. На рисунке 5.1 представлен пример остаточного блока.

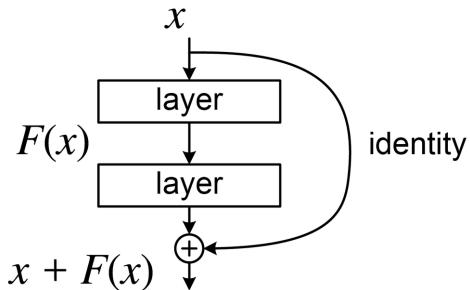


Рис. 5.1: Остаточный блок [resnet].

ResNet предлагает несколько версий моделей с различной глубиной: ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152. Эти версии различаются количеством остаточных блоков и слоев, что позволяет выбрать модель, соответствующую конкретной задаче и доступным вычислительным ресурсам.

5.2.2 MobileNet

MobileNet — это семейство моделей нейронных сетей, специально разработанных для работы на мобильных и встраиваемых устройствах [mobilenet]. Они хорошо оптимизированы и обеспечивают высокую точность при низких вычислительных затратах. MobileNet широко используется в задачах компьютерного зрения, таких как классификация изображений, детекция объектов и сегментация.

Основной инновацией MobileNet является использование глубоких разделяемых сверточных слоев вместо стандартных. Глубокие разделяемые сверточные слои состоят из двух отдельных операций: глубинного свертывания (от англ. depthwise convolution) и точечного свертывания (от англ. pointwise convolution). Глубинное свертывание применяется отдельно к каждому каналу входного изображения, а точечное свертывание используется для объединения выходов глубинного свертывания.

MobileNet использует параметры разложения для управления шириной сети (от англ. width multiplier) и разрешением входного изображения (от англ. resolution multiplier). Width multiplier уменьшает количество каналов в каждой сверточной операции, что снижает количество вычислений и параметров, а resolution multiplier уменьшает размер входного изображения, что дополнительно снижает вычислительные затраты.

Благодаря глубинным разделяемым сверточным слоям и параметрам разложения, MobileNet достигает хорошего баланса между производительностью и вычислительными затратами. На рисунке 5.2 представлена архитектура MobileNet слоя.

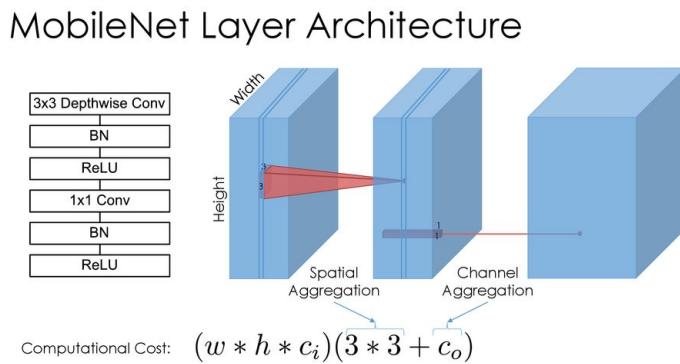


Рис. 5.2: Архитектура MobileNet слоя [mobilenet].

На данный момент существует несколько версий MobileNet, каждая из которых приносит свои улучшения и оптимизации.

MobileNetV2 — включает улучшения, такие как инвертированные остаточные блоки (от англ. inverted residuals) и линейные узкие слои (от англ. linear bottlenecks) [mobilenetv2].

Эти улучшения помогают сохранять более полезные характеристики признаков и обеспечивают лучшую производительность при тех же вычислительных затратах. Разница между остаточным и инвертированным остаточным блоком представлена на рисунке 5.3.

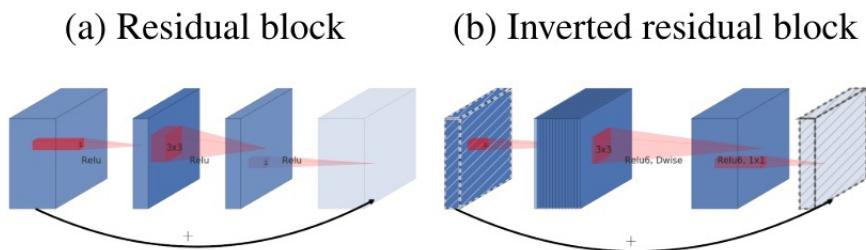


Рис. 5.3: Разница между остаточным и инвертированным остаточным блоком [mobilenetv2].

MobileNetV3 — включает дополнительные улучшения, такие как блоки сжатия и возбуждения (от англ. squeeze-and-excitation, SE) и расширенные функции активации [mobilenetv3], см. рисунок 5.4. Блоки SE позволяют сети лучше фиксировать канальные зависимости в данных, помогает повысить способность модели извлекать значимые функции из входных данных, что приводит к повышению производительности задач распознавания изображений. MobileNetV3 использует расширенные функции активации, такие как h-swish и h-sigmoid. Эти функции обеспечивают более плавные градиенты во время обучения, что может привести к более быстрой сходимости и повышению общей производительности. MobileNetV3 предоставляет две версии: MobileNetV3-Large и MobileNetV3-Small, предназначенные для различных требований производительности и эффективности.

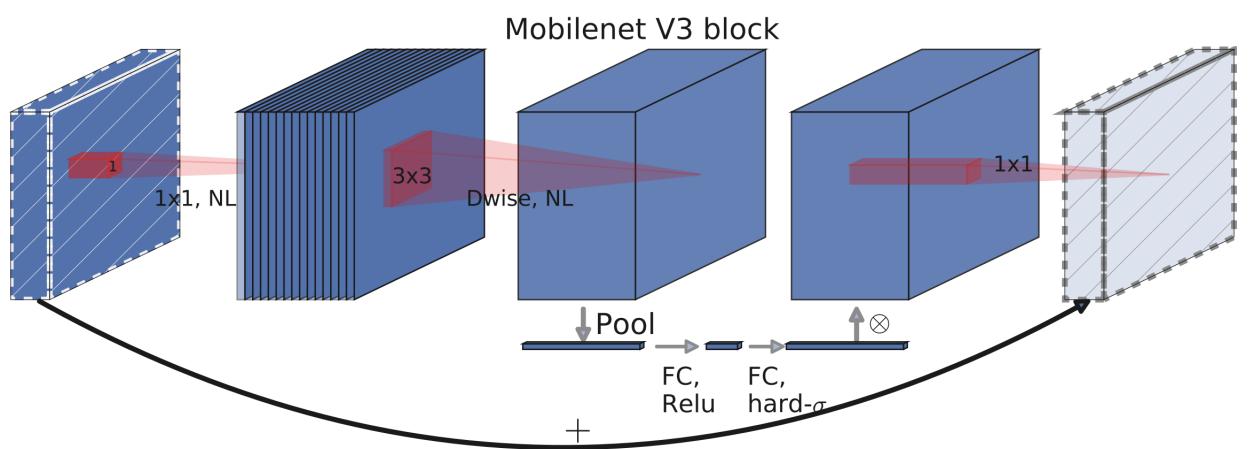


Рис. 5.4: MobileNetV3 блок [mobilenetv3].

Однако, несмотря на свою эффективность и компактность, MobileNet в сравнении с более крупными и сложными моделями, такими как EfficientNet, может не достигать такой же высокой точности на некоторых задачах компьютерного зрения.

5.2.3 EfficientNet

EfficientNet — архитектура нейронной сети, которая основывается на идеи масштабирования моделей и балансирования между собой глубины и ширины (количества каналов) сети, а также разрешения изображений[efficientnet]. Предлагается новый метод составного масштабирования (англ., compound scaling method), который равномерно изменяет глубину, ширину и разрешение с фиксированными пропорциями между ними. Метод составного масштабирования показан на рисунке 5.5.

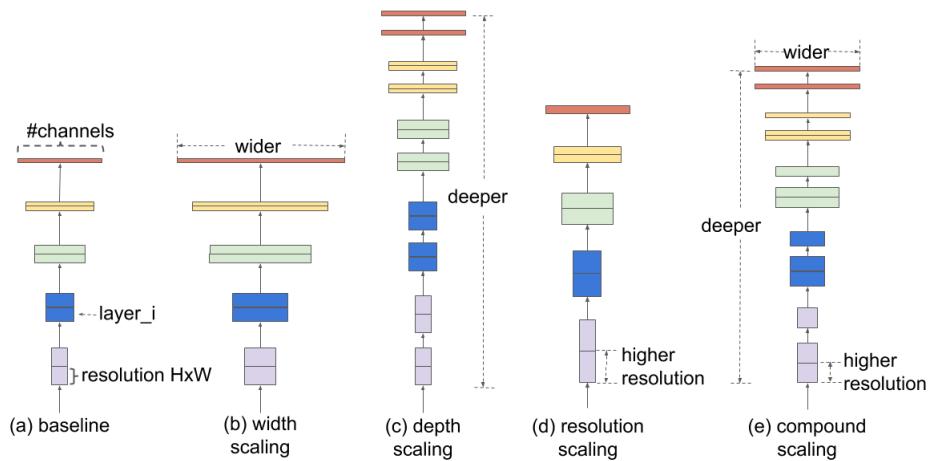


Рис. 5.5: Различные методы масштабирования по сравнению со составным[efficientnet].

Архитектура EfficientNet состоит из следующих ключевых компонентов:

1. MBConv (англ., Mobile Inverted Bottleneck Convolution) — основной строительный блок EfficientNet, унаследованный из архитектуры MobileNetV2. MBConv включает в себя следующие элементы:
 - Pointwise Convolution (1×1) — уменьшает размерность каналов, улучшая вычислительную эффективность.
 - Depthwise Convolution (3×3 или 5×5) — выполняет свертку по каждому каналу независимо, что значительно снижает количество параметров.
 - Squeeze-and-Excitation Block — улучшает представление важных характеристик путем адаптивного перенастройки каналов.
2. Использование функции активации Swish, которая демонстрирует лучшие результаты в глубоких сетях благодаря своей непрерывности и плавности.
3. Compound Scaling — ключевая концепция EfficientNet, включающая одновременное масштабирование трех аспектов модели:

- Глубина — увеличение количества слоев в сети.
- Ширина — увеличение количества каналов в каждом слое.
- Разрешение — увеличение разрешения входных изображений.

Подход compound scaling обеспечивает сбалансированное увеличение модели, что позволяет достигать высокой точности и производительности без чрезмерного увеличения вычислительных затрат, что делает EfficientNet особенно привлекательной для внедрения в мобильные устройства.

Существует несколько версий EfficientNet, каждая из которых разработана для различных применений и требований к вычислительным ресурсам. Оригинальные модели используют простой метод compound scaling, который включает масштабирование глубины, ширины и разрешения изображений с помощью фиксированных коэффициентов. В то время как EfficientNetV2 улучшает этот метод, вводя динамическое масштабирование, которое позволяет более гибко адаптировать модель к различным задачам и условиям. Также EfficientNetV2 использует улучшенные блоки, такие как Fused-MBConv, которые объединяют традиционные MBConv и элементы из ResNet, что позволяет увеличить эффективность обучения и улучшить качество представления данных.

Оригинальные модели EfficientNet (B0 - B7):

- EfficientNet-B0 — базовая модель, на основе которой построены все остальные версии, использует минимальное количество параметров и вычислительных ресурсов. Архитектура сети представлена на рисунке 5.6
- EfficientNet-B1 - B7 — модели, которые последовательно масштабируются с использованием compound scaling, увеличивая глубину, ширину и разрешение. Каждая следующая версия обладает большей вычислительной мощностью и точностью по сравнению с предыдущей.

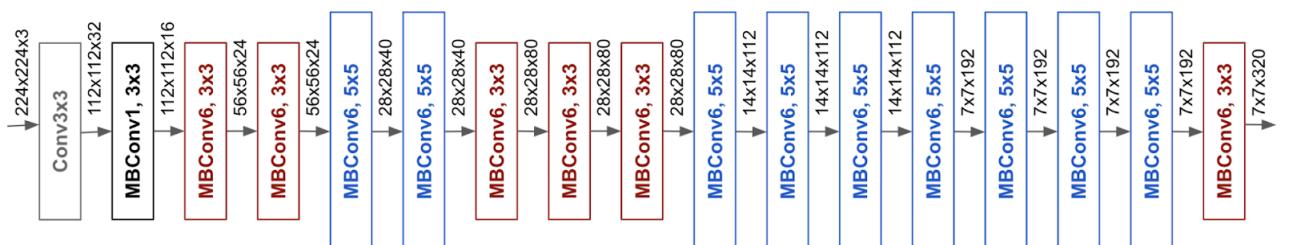


Рис. 5.6: Архитектура базовой сети EfficientNet-B0[efficient-b0].

Модели EfficientNetV2 (S, M, L):

- EfficientNetV2-S — оптимизированная версия для мобильных устройств, характеризующаяся компактностью и высокой эффективностью. Подходит для приложений, где критичны вычислительные ресурсы и время отклика.
- EfficientNetV2-M — средняя версия, предлагающая баланс между производительностью и вычислительными затратами. Часто используется в серверных и облачных средах.
- EfficientNetV2-L — наиболее мощная версия, предназначенная для задач, требующих максимальной точности и производительности. Используется в высокопроизводительных системах и приложениях с большими объемами данных.

Благодаря продуманному подходу к масштабированию, все версии EfficientNet демонстрируют высокую точность на стандартных наборах данных, таких как ImageNet. Стоит отметить, что модели EfficientNetV2 значительно превосходят оригинальные модели EfficientNet. Так например, EfficientNetV2-M снижает параметры на 17 %, а количество операций ввода-вывода — на 37 %, но при этом работает в 4.1 раза быстрее при обучении и в 3.1 раза быстрее при выводе, чем EfficientNet-B7. Также модели EfficientNetV2 значительно быстрее и обеспечивают лучшую точность и эффективность параметров, чем предыдущие модели ConvNet и Vision Transformers на ImageNet [[efficientnetv2](#)]. Несмотря на высокую производительность, EfficientNetV2 не использует механизм внимания, как ViTs, что может быть недостатком в некоторых случаях.

5.2.4 Vision Transformers

Визуальные трансформеры (от англ. Vision Transformers, ViTs) — класс моделей глубокого обучения, которые представляют собой архитектуру в области компьютерного зрения, адаптированную из трансформеров, широко используемых в обработке естественного языка [[vit](#)]. Основная идея заключается в разделении изображения на небольшие патчи (части), которые затем обрабатываются как токены последовательности в модели трансформера. ViTs используют механизм самовнимания для анализа изображений, что позволяет моделям эффективно захватывать глобальные и локальные зависимости в данных. Vision Transformers достигли выдающихся результатов в задачах классификации изображений.

Архитектура Vision Transformers состоит из следующих этапов (см. рисунок [5.7](#)):

1. Разделение на патчи. Изображение делится на небольшие патчи фиксированного размера (например, 16x16 пикселей), которые затем выравниваются в одномерные векторы. Каждый патч рассматривается как токен в последовательности.

2. Линейное преобразование патчей. Каждому патчу сопоставляется эмбеддинг фиксированной размерности с помощью линейного слоя. Этот шаг аналогичен созданию эмбеддингов для слов в задачах NLP.
3. Добавление позиционных эмбеддингов. Поскольку трансформеры изначально не содержат информации о позиции токенов, то к эмбеддингам патчей добавляются позиционные эмбеддинги, которые кодируют пространственное положение каждого патча в исходном изображении.
4. Последовательность патчей передается через многослойную трансформерную модель, состоящую из чередующихся слоев самовнимания (self-attention) и полносвязных слоев (feed-forward), с целью учесть взаимосвязи между всеми патчами одновременно, эффективно захватывая глобальные и локальные зависимости в изображении.
5. К началу последовательности патчей добавляется специальный классификационный токен CLS, который предназначен для агрегирования информации от всех патчей и используется для финальной классификации. Выходной эмбеддинг CLS токена передается через несколько полносвязных слоев с функцией активации (например, ReLU), завершающихся слоем Softmax для получения вероятностей классов. Финальная классификационная голова преобразует выходной эмбеддинг в предсказание класса для всего изображения.

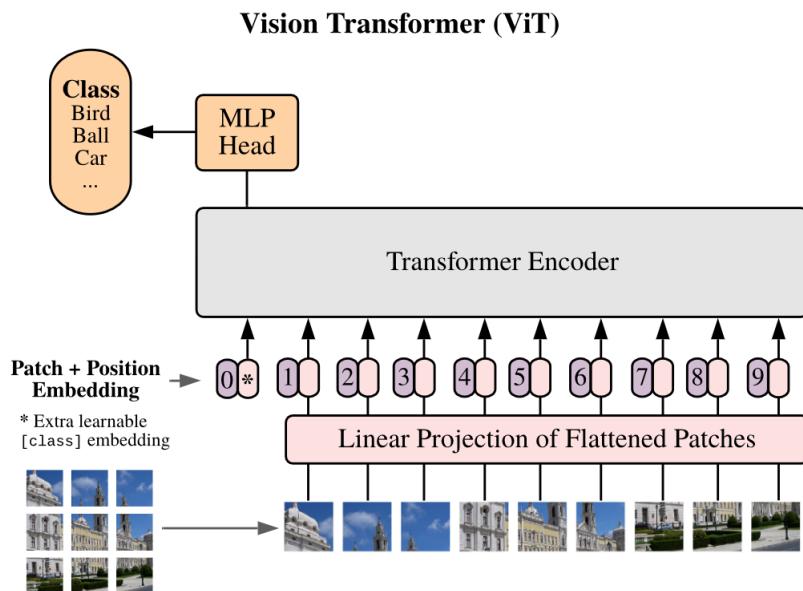


Рис. 5.7: Архитектура Vision Transformers[vit].

Архитектура Vision Transformers достигла высоких показателей производительности в

задаче классификации изображений на датасете ImageNet после предварительного обучения на JFT-300M [vit]. Однако важно отметить, что эффективность моделей резко снижается при использовании небольших объемов данных для предварительного обучения. Более крупные модели чаще всего показывают лучшие результаты на больших наборах данных. Это проблема, вероятно, связана с тем, что трансформерные сети не имеют предварительных предположений о порядке данных, что может привести к их недостаточной адаптации к конкретным требованиям задачи или к обучающему набору данных.

5.3 Используемые метрики в задаче классификации

В задаче классификации для оценки качества моделей машинного обучения используются специальные показатели — метрики.

Для описания метрик используется матрица ошибок классификаций (англ. confusion matrix). Пусть есть два класса и метод, предсказывающий принадлежность каждого объекта одному из классов. В таблице 5.1 представлена матрица ошибок классификации.

	$y = 1$	$y = 0$
$\hat{y} = 1$	True Positive (TP)	False Positive (FP)
$\hat{y} = 0$	True Negative (TN)	False Negative (FN)

Таблица 5.1: Матрица ошибок классификации. y — истинный класс, \hat{y} — результат модели.

Простейшей метрикой является accuracy — доля правильных ответов:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Стоит отметить, что эта метрика может быть применима только на сбалансированном датасете.

Для оценки качества работы алгоритма на каждом из классов по отдельности вводятся метрики точность (англ. precision) и полнота (англ. recall):

$$precision = \frac{TP}{TP + FP} \quad (2)$$

$$recall = \frac{TP}{TP + FN} \quad (3)$$

Precision — доля объектов, которые классификатор определил как положительные и при этом они действительно являются положительными, а recall — метрика, которая показывает, какую долю положительных объектов из всех объектов положительного класса

обнаружила модель. Введение precision не позволяет определять все объекты в один класс, так как в этом случае получается рост FP. Recall показывает способность метода вообще обнаруживать данный класс, а precision — способность отличать этот класс от других. В отличие от accuracy, recall и precision не зависят от соотношения классов и поэтому могут быть применимы на несбалансированном датасете.

Также существует несколько способов объединить recall и precision в объединенный критерий оценки. Например, F-мера (в общем случае F_β) — среднее гармоническое recall и precision:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \quad (4)$$

β в данном случае определяет вес точности в метрике.

F-мера достигает максимума при точности и полноте, равными 1, и близка к 0, если один из аргументов близок к 0.

5.4 Задача создания подписей к изображениям

Задача создания подписей к изображениям (от англ. image captioning) предполагает использование методов глубокого обучения для генерации текстовых описаний на основе визуального содержимого изображений. Для решения этой задачи используется архитектура кодера-декодера (от англ. encoder-decoder). Основные подходы для ее реализации:

1. Использование комбинации сверточных нейронных сетей для обработки изображения и рекуррентных нейронных сетей для генерации подписи.
2. Использование трансформеров.

Архитектура с использованием комбинации CNN и RNN представлена на рисунке 5.8. В качестве кодировщика используется CNN, такие как ResNet, MobileNet или EfficientNet. Эти модели предварительно обучены на больших наборах данных, таких как ImageNet, для извлечения визуальных признаков. Кодировщик отвечает за преобразование изображения в компактное представление (вектор признаков), которое сохраняет важную информацию об изображении. В качестве вектора признаков изображения может использоваться выходной слой CNN, но чаще всего это слой перед последним полносвязанным. Декодировщик принимает вектор признаков изображения и генерирует последовательность слов, которая описывает изображение. Декодеры, как правило, основаны на рекуррентных нейронных сетьях (RNN), такие как LSTM (Long Short-Term Memory) или GRU (Gated Recurrent Unit).

Одна из основных проблем такого подхода — это то, что RNN имеет фиксированную длину контекста и обрабатывает информацию последовательно. Это означает, что RNN видит только небольшой контекст, поэтому при генерации слов часто забывает старые результаты своей генерации. Так же такой подход не позволяет полностью учесть контекст изображения на каждом этапе генерации. В данном случае выделенные признаки изображения с помощью CNN лишь единожды подаются на вход LSTM блока, так что со временем генерация полностью теряет память о исходном изображении и начинает «додумывать самостоятельно». Для решения этих проблем применяется механизм внимания (от англ. attention)[**attention**], основная идея которого состоит в том, чтобы позволить модели обрабатывать входные данные, учитывая их важность и относительные взаимосвязи между элементами данных. Это позволяет модели более гибко и точно адаптироваться к различным задачам и условиям входных данных, улучшая качество и эффективность моделирования. Механизм внимания может быть реализован разными способами:

- В классической реализации RNN скрытый вектор объединяет в себе информацию, содержащуюся в векторах вывода текущей итерации, через среднее или суммирование, которое можно заменить определенным взвешиванием всех предыдущих векторов, участвующих в усреднении, т.е для получения k -го скрытого вектора производится взвешивание $k-1$ предыдущего вектора. Веса при этом подбираются в зависимости от важности того или иного слова, что позволяет модели сосредотачивать внимание на различных аспектах входных данных в соответствии с контекстом задачи.
 - Механизм внимания может быть интегрирован для учета контекста изображения при генерации каждого слова в описании. В этом случае кодер обрабатывает входное изображение с тремя цветовых каналов, и преобразует его в изображение меньшего размера с «обученными» каналами. Это уменьшенное изображение представляет собой сводное представление всех полезных данных из исходного изображения. Механизм внимания использует это закодированное изображение, что позволяет сети «обращаться» к различным частям изображения на каждом шаге генерации текста. Весь этот процесс можно интерпретировать как вычисление вероятности того, что пиксел является важным для генерации следующего слова (см. рисунок 5.9). Таким образом, контекст самого изображения не теряется в процессе генерации текста. Механизм внимания позволяет сети фокусироваться на разных частях изображения с различной степенью «важности» на каждом этапе генерации, что значительно улучшает качество финального описания.
- На рисунке 5.10 представлена архитектура декодера RNN с использованием механизма

внимания на части изображения.

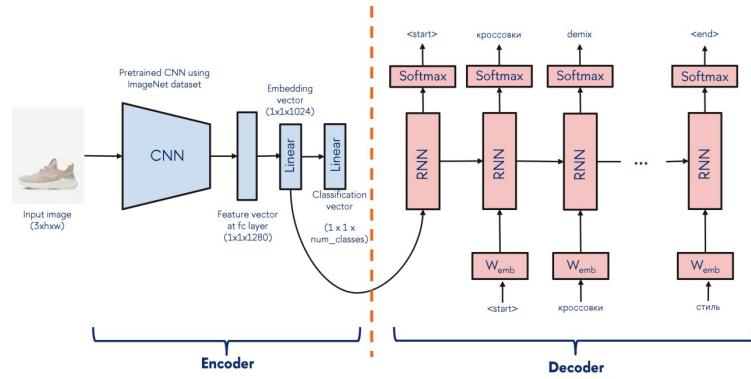


Рис. 5.8: Архитектура задачи Image Captioning с использованием комбинации CNN для обработки изображения и RNN для генерации подписи.

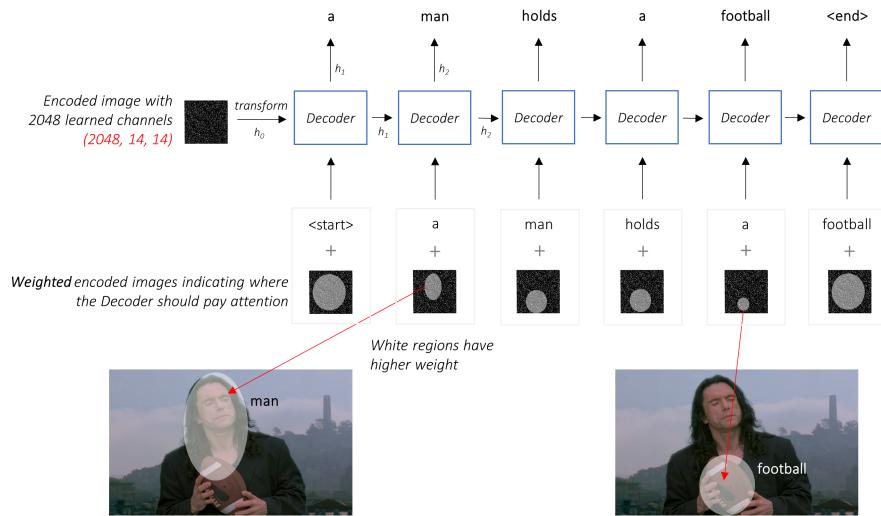


Рис. 5.9: Генерации последовательности с использованием механизма внимания на части изображения в задаче Image Captioning [cnn-rnn-image-captioning].

Преимущества реализации архитектуры с использованием комбинации CNN и RNN:

- CNN позволяет эффективно и точно выделять особенности изображения, что обеспечивает богатую информацию для генерации текста;
- модели CNN + RNN можно легко адаптировать и дообучать для различных задач и наборов данных.

К недостаткам данной архитектуры можно отнести следующее:

- Несмотря на улучшения в виде LSTM и GRU, RNN могут испытывать трудности с моделированием длинных текстов, что может влиять на качество генерации.

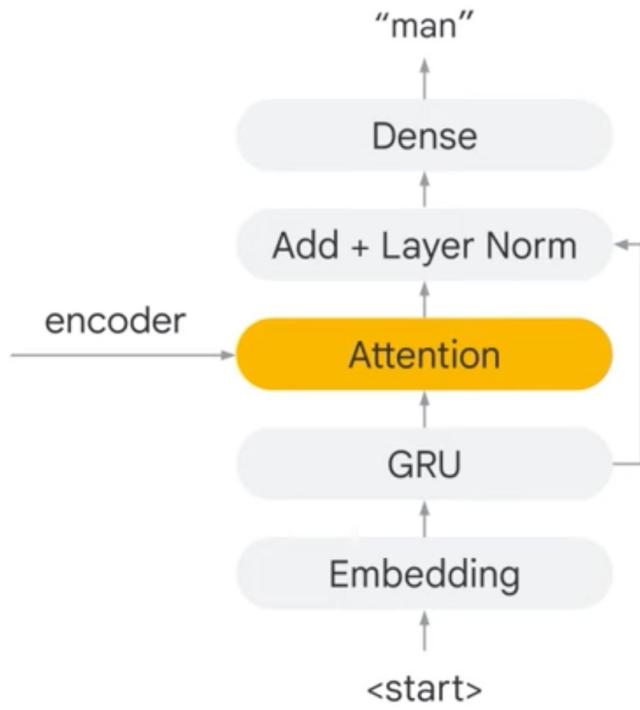


Рис. 5.10: Архитектура декодера RNN с использованием механизма внимания на части изображения в задаче Image Captioning [[cnn-rnn-image-captioning-google](#)].

- RNN могут генерировать текст, который не всегда полностью согласован с изображением. Иногда описание может быть частично правильным, но при этом не охватывать все важные детали изображения.

Архитектура с использованием трансформеров представлена на рисунке 5.11. В качестве кодера используется Vision Transformers (ViTs), который применяет механизмы самовнимания к патчам изображения для извлечения признаков. В качестве декодера используются модели GPT, которые также используют механизмы самовнимания для моделирования долгосрочных зависимостей в тексте, что позволяет генерировать более связные и правильные описания. Ключевым моментом является правильная подача признаков изображения, извлеченных из ViTs, в соответствующие слои самовнимания GPT, то есть преобразование этих признаков в формат ключей (keys), значений (values) и запросов (queries).

Преимущества реализации:

- способность обрабатывать большие объемы данных и извлекать сложные зависимости.
- модели GPT-2 обладают высоким качеством генерации текста благодаря своему предобучению на огромных объемах текстовых данных. Они способны генерировать раз-

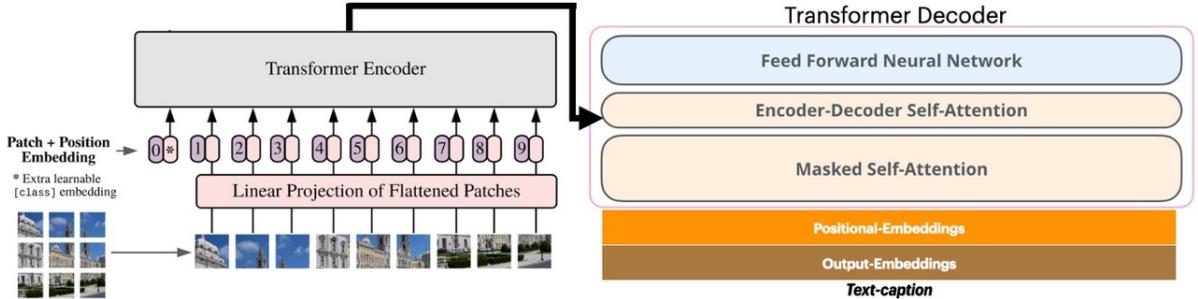


Рис. 5.11: Архитектура задачи Image Captioning с использованием трансформеров [vit-gpt2-image-captioning].

нообразные и грамматически правильные описания для изображений.

К недостатком данной архитектуры можно отнести то, обучение моделей Vision Transformers и GPT-2 требует значительных вычислительных ресурсов и времени из-за их огромного размера и сложности.

Стоит отметить, что совместить способы решения задачи нельзя, из-за фундаментальных различий в их архитектурных подходах и методах обработки данных. CNN и RNN работают на основе последовательной обработки данных, в то время как ViTs и GPT используют механизмы самовнимания, которые позволяют параллельно обрабатывать данные. Комбинация подходов может потребовать значительных изменений в архитектурах, что вряд ли приведет к улучшению результатов. Поэтому выбор архитектуры (CNN + RNN или ViT + GPT) зависит от конкретных требований проекта и условий, в которых будет использоваться модель. В некоторых случаях может быть полезным попробовать обе архитектуры и выбрать ту, которая лучше соответствует задачам и ресурсам. Для оценки качества генерации используются метрики, такие как BLEU, WER, METEOR и CIDEr.

5.5 Обзор моделей для генерации текста

В данном разделе рассматриваются ключевые модели генерации текста, их архитектурные особенности, а также преимущества и недостатки.

5.5.1 Рекуррентные нейронные сети

Рекуррентные нейронные сети (Recurrent Neural Networks, RNN) — это класс нейронных сетей, специально разработанных для обработки последовательных данных. RNN широко применяются в задачах, где порядок и зависимость данных во времени играют важную роль, например, в таких как обработка текста и генерация последовательностей.

RNN способны учитывать временные зависимости в данных, обрабатывая последовательности входных сигналов. Это достигается благодаря наличию скрытых состояний (hidden states), которые обновляются на каждом шаге последовательности и хранят информацию о предыдущих шагах. В отличие от обычных нейронных сетей, RNN имеют петли обратной связи, позволяющие информации циркулировать внутри сети. Это позволяет RNN «помнить» предыдущие шаги и использовать эту информацию для текущих вычислений (см. рисунок 5.12).

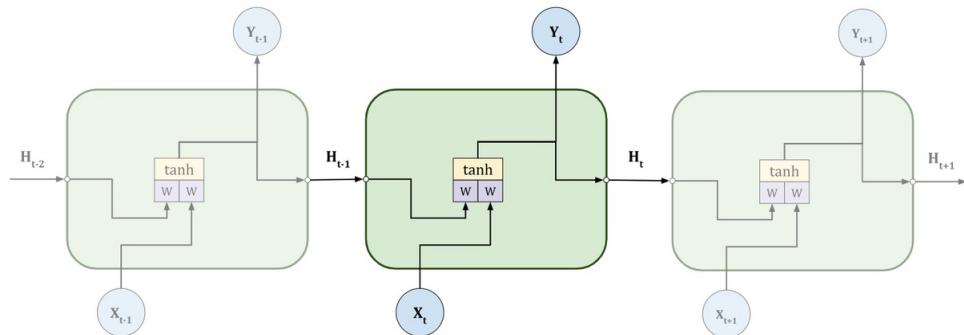


Рис. 5.12: Архитектура RNN.

На каждом шаге t входной вектор x_t и предыдущее скрытое состояние h_{t-1} используются для вычисления текущего скрытого состояния h_t :

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b_h) \quad (5)$$

где W_h и W_x — веса, b_h — смещение, и \tanh — активационная функция.

При обучении очень длинных последовательностей RNN могут столкнуться с проблемой затухания или взрыва градиентов. Это приводит к тому, что веса либо становятся очень маленькими, либо слишком большими, что затрудняет обучение.

Для решения проблем затухания и взрыва градиентов были предложены более сложные архитектуры, такие как Long Short-Term Memory (LSTM) и Gated Recurrent Unit (GRU).

LSTM — это разновидность RNN, специально разработанная для борьбы с проблемой затухания градиентов. LSTM используют специальные ячейки памяти и механизмы управления потоком информации (забывающие и обновляющие ворота), что позволяет им эффективно хранить и обрабатывать данные с долгосрочными зависимостями, что было недостижимо для традиционных RNN.

Ячейка LSTM состоит из трех основных компонентов:

1. Ворота забывания (от англ. Forget Gate), которые используются для управления про-

должительностью памяти в ячейке, решая, какую информацию следует забыть, а какую сохранить:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (6)$$

где f_t — значение ворот забывания, W_f и b_f — веса и смещения для ворот забывания.

2. Ворота ввода (от англ. Input Gate), которые определяют, какая часть новой информации должна быть сохранена в долгосрочной памяти ячейки LSTM. Они не только фильтруют входные данные, но и решают, какая информация достаточно важна для сохранения:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (7)$$

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C) \quad (8)$$

где i_t — значение входного ворота, W_i и b_i — веса и смещения для входного ворота, \tilde{C}_t — кандидат на обновление состояния.

3. Ворота вывода (от англ. Output Gate), которые определяют, какая информация из текущего состояния ячейки будет передана в выходной сигнал сети.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (9)$$

$$h_t = o_t \cdot \tanh(C_t) \quad (10)$$

где o_t — значение выходного ворота, W_o и b_o — веса и смещения для выходного ворота, h_t — скрытое состояние на текущем шаге, C_t — обновленное состояние ячейки.

Комбинируя результаты всех трех ворот, LSTM обновляет состояние ячейки и скрытое состояние следующим образом:

1. Обновление состояния ячейки:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad (11)$$

2. Обновление скрытого состояния:

$$h_t = o_t \cdot \tanh(C_t) \quad (12)$$

Архитектура ячейки LSTM представлена на рисунке 5.13

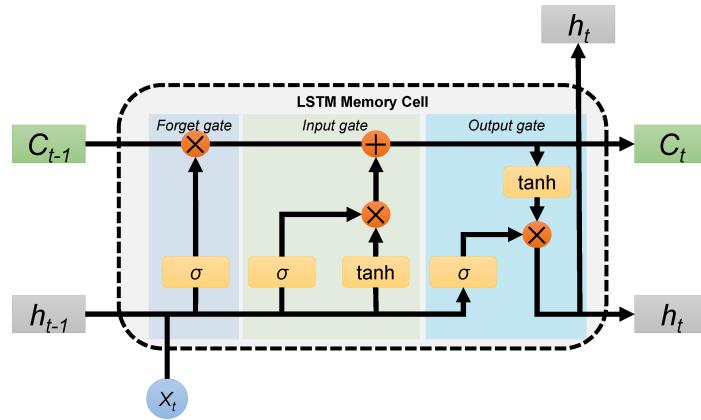


Рис. 5.13: Архитектура ячейки LSTM.

GRU — это упрощенная версия LSTM, которая объединяет забывающие и обновляющие ворота в одно целое, что делает архитектуру менее вычислительно затратной и более простой для реализации. В GRU нет отдельной ячейки памяти, как в LSTM. Вместо этого, она обновляет скрытое состояние напрямую. Ворота обновления помогают модели решить, какая часть предыдущего скрытого состояния должна быть сохранена. Это делается путем комбинирования прошлого состояния и новой информации.

GRU использует два основных компонента:

1. Ворота обновления (от англ. Update Gate), которые контролируют какая часть предыдущего скрытого состояния будет перенесена в текущее скрытое состояние.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \quad (13)$$

где z_t — значение обновляющего ворот, W_z и b_z — веса и смещения для обновляющего ворот.

2. Ворота сброса (от англ. Reset Gate), которые какая часть предыдущего скрытого состояния будет сброшена перед вычислением нового состояния.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \quad (14)$$

где r_t — значение сбрасывающего ворот, W_r и b_r — веса и смещения для сбрасывающего ворот.

Новое скрытое состояние вычисляется с учетом текущего ввода и предыдущего скрытого состояния, модулированного сбрасывающими воротами. Формула для кандидата на но-

вое скрытое состояние:

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h) \quad (15)$$

где \tilde{h}_t — кандидат на новое скрытое состояние, \odot обозначает поэлементное умножение.

Итоговое скрытое состояние вычисляется как взвешенная сумма предыдущего скрытого состояния и кандидата на новое скрытое состояние, где веса задаются обновляющими воротами:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (16)$$

где h_t — новое скрытое состояние, z_t — значение обновляющих ворот.

Архитектура ячейки GRU представлена на рисунке

Выбор между LSTM и GRU зависит от конкретных требований задачи. Если скорость обучения является приоритетом и доступные вычислительные ресурсы ограничены, то использование GRU может быть предпочтительнее благодаря своей более простой структуре и меньшему количеству параметров по сравнению с LSTM. Также использование GRU может быть достаточным для более простых или меньших наборов данных, так как она способна эффективно обучаться на таких данных без потери производительности. Однако если задача требует более детального управления информацией и долгосрочной памяти (например, в сложных задачах обработки естественного языка с длинными зависимостями), то LSTM может быть более подходящим выбором благодаря своей дополнительной сложности и дополнительному контролю над информацией.

5.5.2 Модели генеративного предварительно обученного трансформера

Модели генеративного предварительно обученного трансформера, (англ. Generative Pre-trained Transformer, GPT) — это семейство языковых моделей на основе глубокого обучения, разработанные командой OpenAI, которые используют архитектуру трансформера (см. рисунок 5.14) для генерации текста, учитывая контекст и структуру предложений. Модели GPT являются мощными инструментами для генерации текста и решения разнообразных задач в области естественного языка. Их успешное функционирование основано на комбинации трансформерной архитектуры, предварительного обучения на больших объемах текстовых данных и тщательного дообучения на конкретной задаче.

Архитектура GPT имеет два основных сегмента: кодировщик, который в основном работает с входной последовательностью, и декодер, который работает с целевой последовательностью во время обучения и предсказывает следующий элемент. Кодер определяет, какие части ввода следует выделить. Затем он вычисляет матрицу встраивания (встраивание

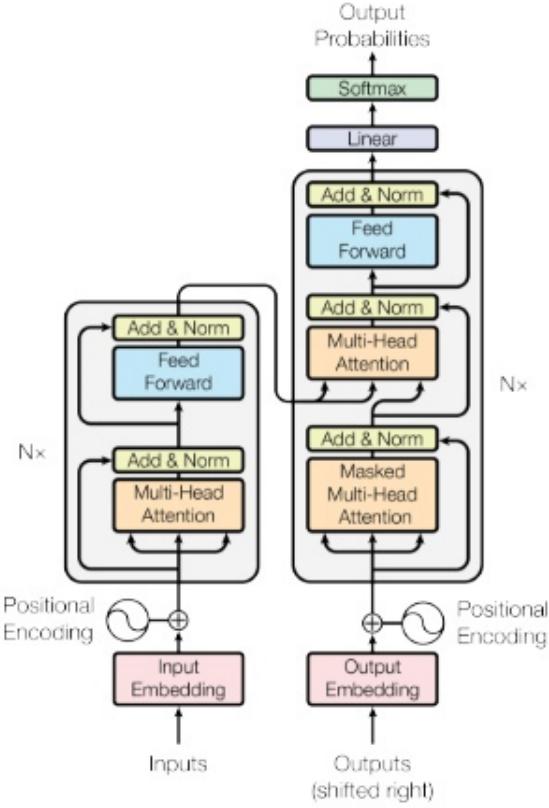


Рис. 5.14: Архитектура трансформера [transformers].

в NLP позволяет словам с похожим значением иметь одинаковое представление) и преобразует ее в серию векторов внимания. Модели GPT используют многоголовое внимание (англ., multi-head attention), которое создает векторы внимания, позволяя модели одновременно фокусироваться на разных частях последовательности. Это позволяет захватывать различные аспекты контекста языка. Созданные вектора внимания проходят через слой нормализации и затем передаются в полносвязный слой. Нормализация помогает стабилизировать и ускорить процесс обучения. Перед передачей в декодер снова выполняется нормализация. Во время обучения кодер работает непосредственно с целевой выходной последовательностью. Декодер вычисляет отдельные векторы встраивания для каждого слова предложения. Поскольку трансформеры не имеют встроенной последовательности обработки данных, то дополнитель-но применяется позиционный энкодер с синусоидальными и косинусоидальными функциями. Кроме того, модель GPT обучается предсказывать следующий токен в последовательности. Для этого используется механизм маскированного внимания (англ., masked attention), который позволяет модели видеть только предшествующие токены. Во время генерации текста модель принимает на вход начальный токен или последовательность токенов и последова-тельно предсказывает следующие токены, генерируя текст по одному слову за раз. Процесс генерации продолжается до достижения определенного критерия остановки, например, та-

кого как достижение максимальной длины текста или специального токена окончания.

Прежде чем приступить к выполнению конкретной задачи, модели GPT проходят через фазу предварительного обучения на больших корпусах текстовых данных. Во время предварительного обучения модель учится «понимать» структуру и смысл текста, а также создавать внутреннее представление о языке. После предварительного обучения модель может быть дообучена или донастроена на конкретной задаче, что улучшает ее производительность и качество результатов.

На данный момент существует несколько версий GPT, каждая из которых приносит свои улучшения и оптимизации:

1. GPT-1 — первая модель из семейства GPT, которая содержит 117 миллионов параметров. Сложность обучения GPT-1 варьируется в зависимости от доступных вычислительных ресурсов и объема данных для обучения, но в то же время остается доступной для исследователей и специалистов благодаря наличию предобученных моделей.
2. GPT-2 содержит 1.5 миллиарда параметров. Обучение GPT-2 требует больших объемов данных и вычислительных ресурсов, что делает его более сложным в обучении, чем GPT-1. Для многих исследовательских групп доступ к достаточным ресурсам может быть препятствием для обучения GPT-2, хотя предобученные модели могут быть доступны для использования.
3. GPT-3 содержит 175 миллиардов параметров. Обучение GPT-3 требует огромных вычислительных ресурсов и больших объемов данных, что делает его крайне сложным в обучении. Для большинства исследовательских групп доступ к таким ресурсам может быть недостижимым. Кроме того, обучение GPT-3 может требовать значительных финансовых затрат на инфраструктуру и вычислительные ресурсы.
4. GPT-4 — мультимодальная большая языковая модель, которая способна обрабатывать запросы в виде картинок и текста, а затем выдавать текстовые ответы. В качестве трансформера GPT-4 была предварительно обучена прогнозировать следующий токен (используя как общедоступные данные, так и «данные, лицензированные сторонними поставщиками»), а затем была доработана с помощью обучения с подкреплением на основе отзывов людей. В техническом отчете GPT-4 явно воздерживаются от указания размера модели, ссылаясь на «конкурентную среду и последствия для безопасности крупномасштабных моделей», но согласно оценки разных источников GPT-4 имеет около 1.76 триллиона параметров.

Для обучения любой модели GPT требуется тщательное планирование и анализ с целью определения оптимальных методов и ресурсов. Это включает выбор подходящего объема данных для обучения, оптимизацию гиперпараметров модели, выбор архитектуры и распределение вычислительных ресурсов. Обучение моделей GPT может быть довольно сложным, но при правильном планировании и наличии необходимых ресурсов эти сложности возможно преодолеть.

5.6 Используемые метрики в задаче создания подписей к изображениям

5.7 Вывод

Этапы развития работы: 0) Классификатор на основе efficient net на 1.5k классов 0.0) Если из коробки качество и скорость классификации на все классы сразу страдают (а они скорее всего будут), то внедрить многоуровневую иерархию моделей попроще.

1) RNN-based архитектура, где энкодер представлен efficient net, а декодер - соответствующей частью LSTM/GRU. Можно улучшать комплексность модели: 1.0) количество параллельных LSTM, чей результат в итоге агрегируется 1.1) многоуровневая LSTM, где скрытые состояния k-ого уровня передаются входом в k+1ый 1.2) добавить механизм внимания (attention) к модели

2) Трансформеры, где мы будем бить картинку на последовательность частей и получать соответствующие эмбеддинги для этих самых частей. Затем информацию из энкодера будем подмешивать через attention в декодере, который может быть представлен LLAMA/GPT-2

6 Проектирование архитектуры программного продукта (в процессе)

В 2022 году граждане Российской Федерации столкнулись с трудностями работы с магазинами приложений для смарт-устройств. Так, например, с 24 февраля из российского раздела App Store были удалены почти 7000 мобильных приложений [russia_app]. Поэтому при построении архитектуры приложения был выбран подход к созданию клиент-серверного приложения, которое будет работать практически на любом устройстве (стационарный компьютер, ноутбук, планшет или смартфон). Одним из способов реализации такого подхода является написание telegram бота. В настоящее время это одно из трендовых направлений в ИТ сфере. В 2024 году Telegram посещают 900 миллионов человек в месяц. По количеству аудитории он входит в пятерку самых популярных мессенджеров в мире. Выбор в пользу написания телеграм-бота вместо веб-приложения может быть обусловлен рядом факторов, включая удобство использования, скорость разработки, доступность, безопасность, а также гибкость и масштабированность для обработки большого количества пользователей.

За логику, работоспособность и правильное функционирование бота отвечает серверная часть (англ. backend), которая скрыта от пользователя. Серверная часть включает в себя:

- модуль для обработки запросов от пользователя;
- модуль для взаимодействия с обученной моделями классификации для определения категории товара и генерации его описания.

В контексте работы с моделями машинного обучения и их применения в системах автономного управления, вынос предсказаний моделей в очередь сообщений может быть полезно для оптимизации работы и повышения устойчивости системы.

Для проектирования, разработки и обучения нейронных сетей использовался следующий стек технологий:

- Язык программирования Python — язык общего назначения, с помощью которого можно решать сложные задачи машинного обучения и быстро создавать прототипы для последующей их отладки. Язык гибкий и мультиплатформенный, имеет обширный набор библиотек для искусственного интеллекта. Python удобно использовать для обработки и подготовки обучающих данных.

- Pytorch — библиотека глубокого обучения с открытым исходным кодом, написанная на языке Python и созданная на базе Torch. Используется для решения различных задач машинного обучения: компьютерное зрение, NLP, создания и обучения нейронных сетей [pytorch].

Вычисления происходили с использованием аппаратных ускорений на GPU Nvidia Tesla V100 и A100.

Для реализации серверной части был выбран Flask — это небольшой и легкий веб-фреймворк, написанный на языке Python, предлагающий полезные инструменты и функции для облегчения процесса создания веб-приложений с использованием Python [flask].

Docker — инструмент для создания и запуска контейнеров.

Kubernetes нужен, когда у вас много контейнеров и узлов, которыми нужно управлять. Также Kubernetes подходит, если вам нужна распределенная отказоустойчивая система.

На рисунке ?? представлена структура приложения.

7 Направление дальнейшего развития

A Приложение 1

Распределение данных по категориям второй вложенности.

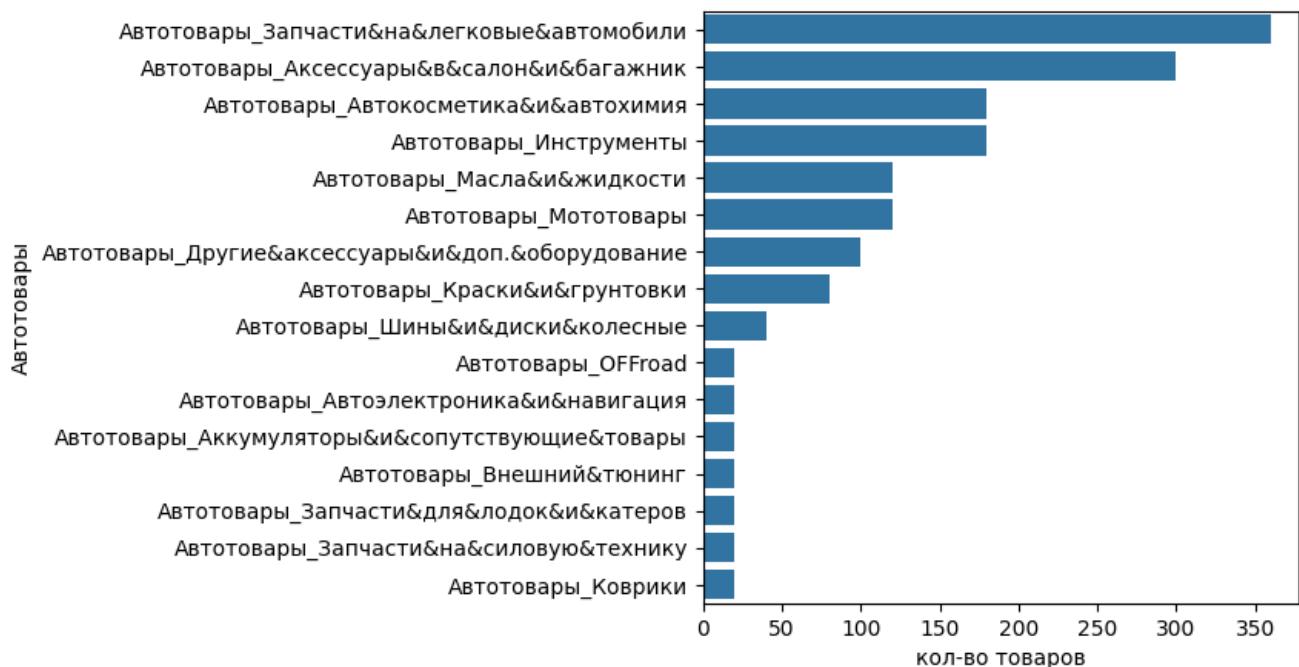


Рис. А.1: Распределение данных в категории «Автотовары».

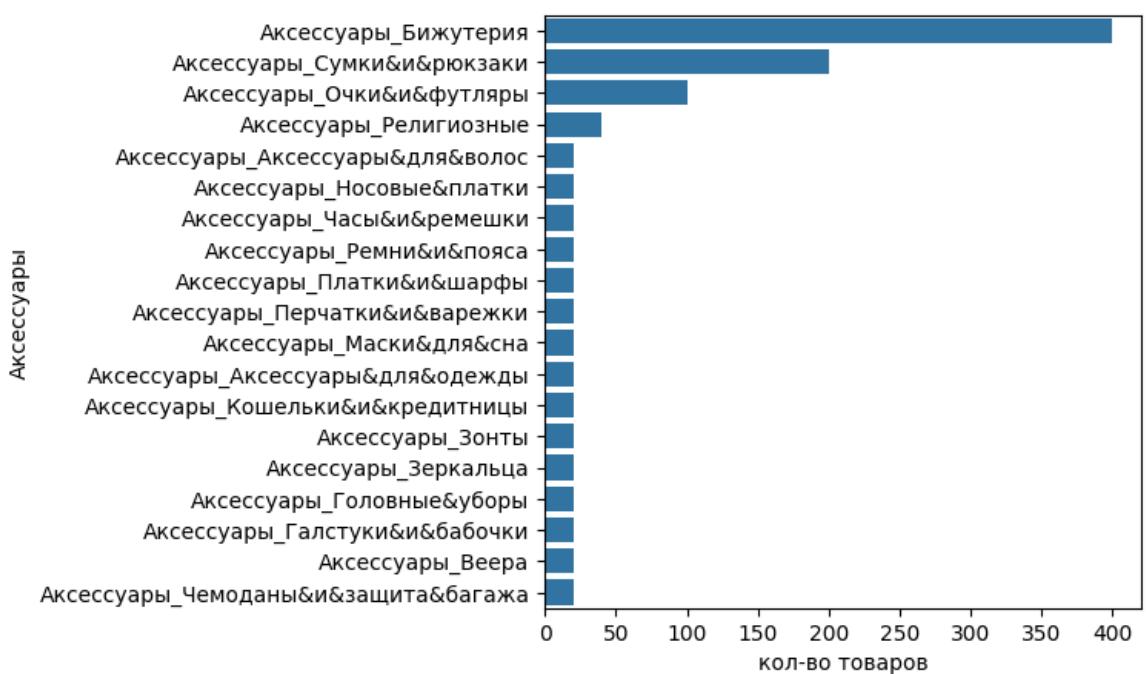


Рис. А.2: Распределение данных в категории «Аксессуары».

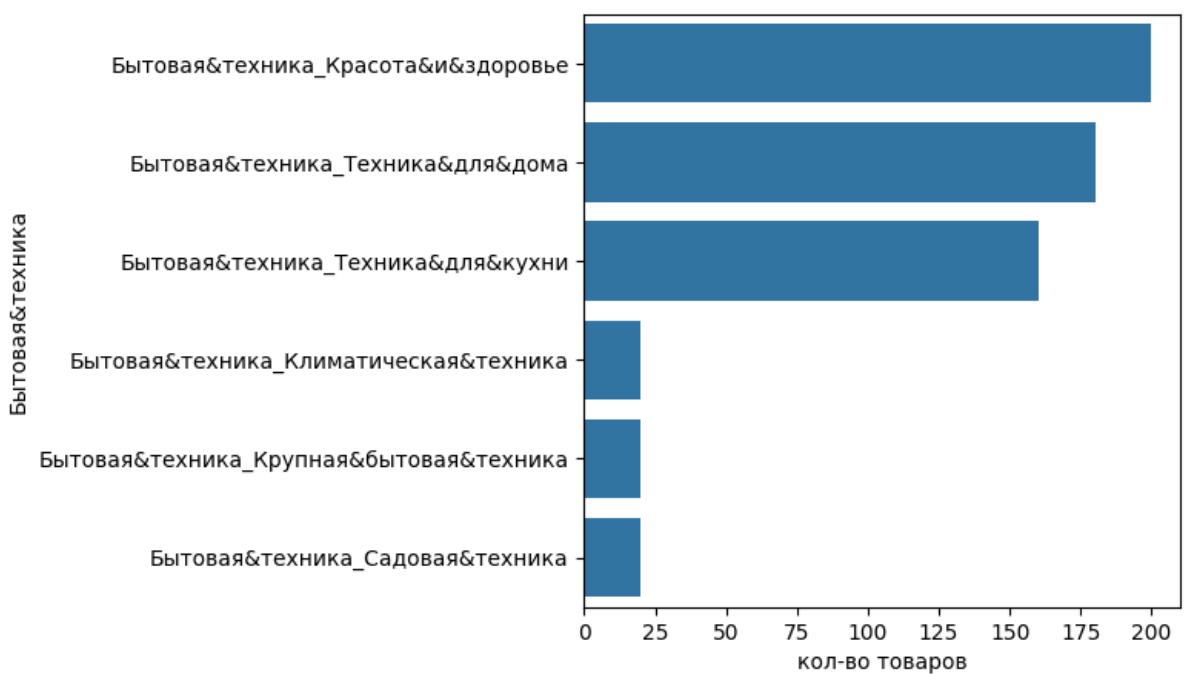


Рис. А.3: Распределение данных в категории «Бытовая&техника».

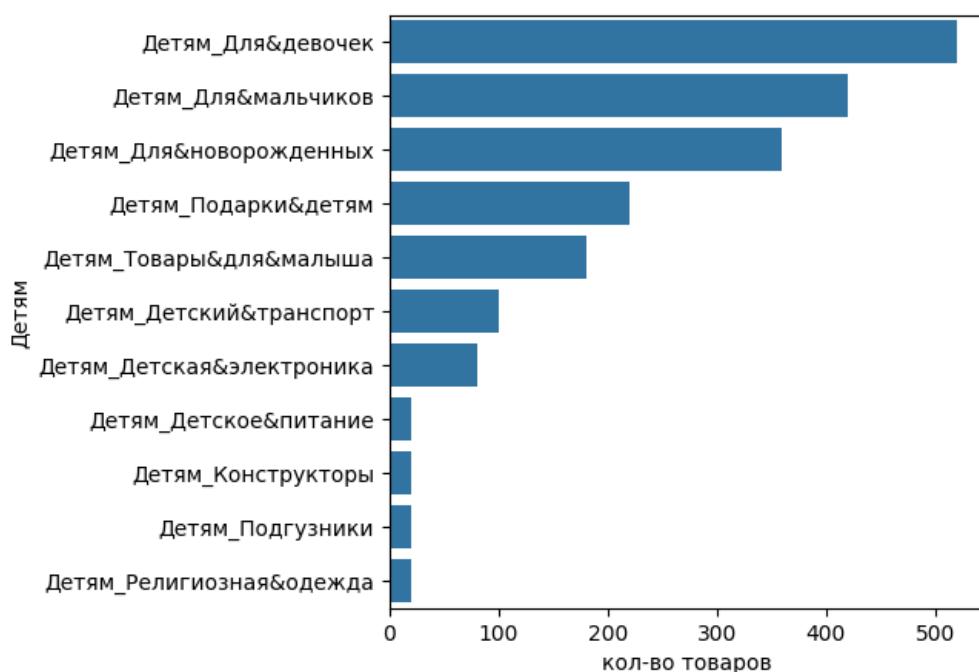


Рис. А.4: Распределение данных в категории «Детям».

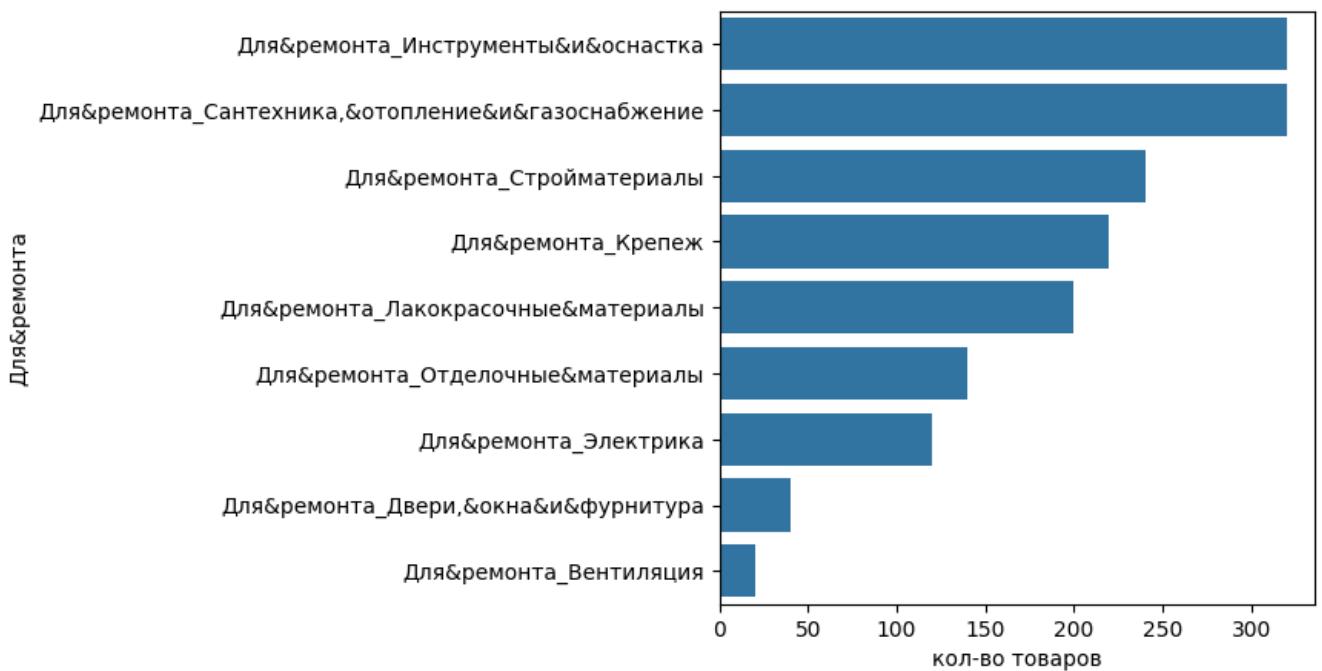


Рис. А.5: Распределение данных в категории «Для&ремонта».

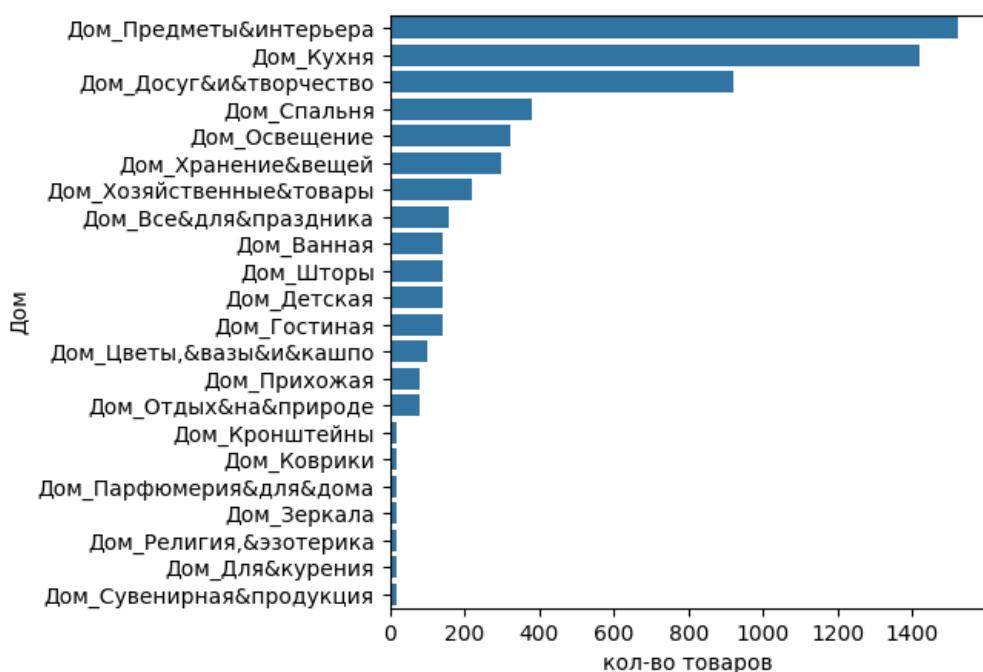


Рис. А.6: Распределение данных в категории «Дом».

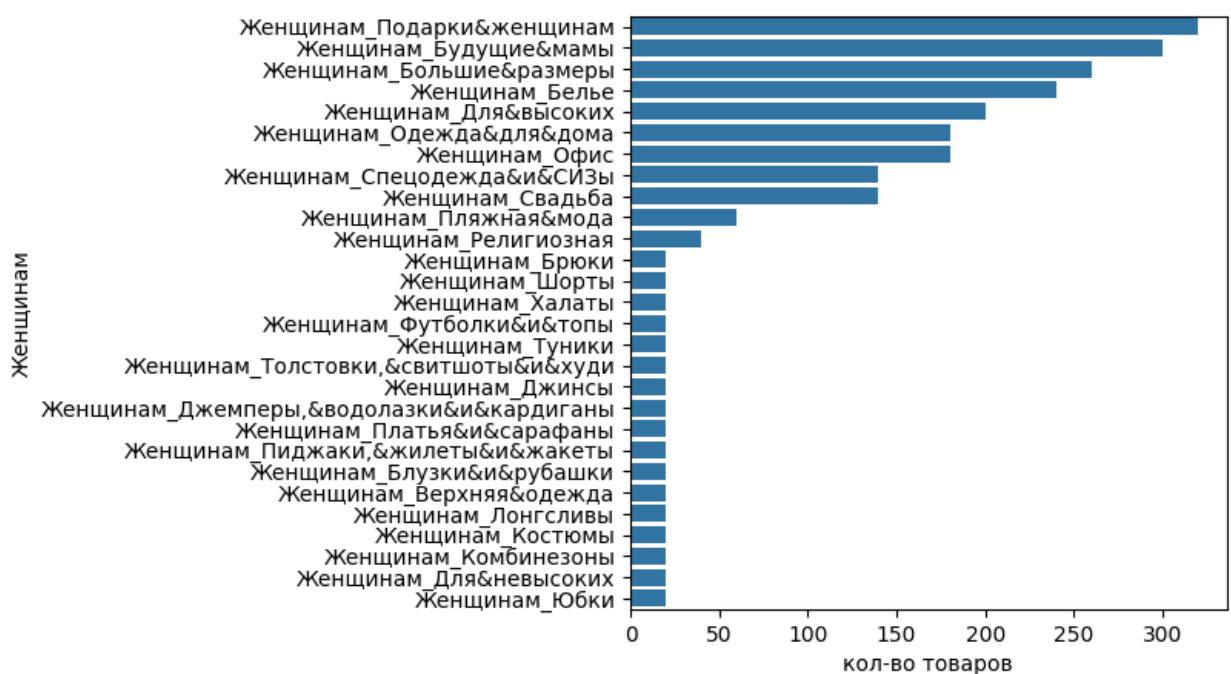


Рис. А.7: Распределение данных в категории «Женщинам».

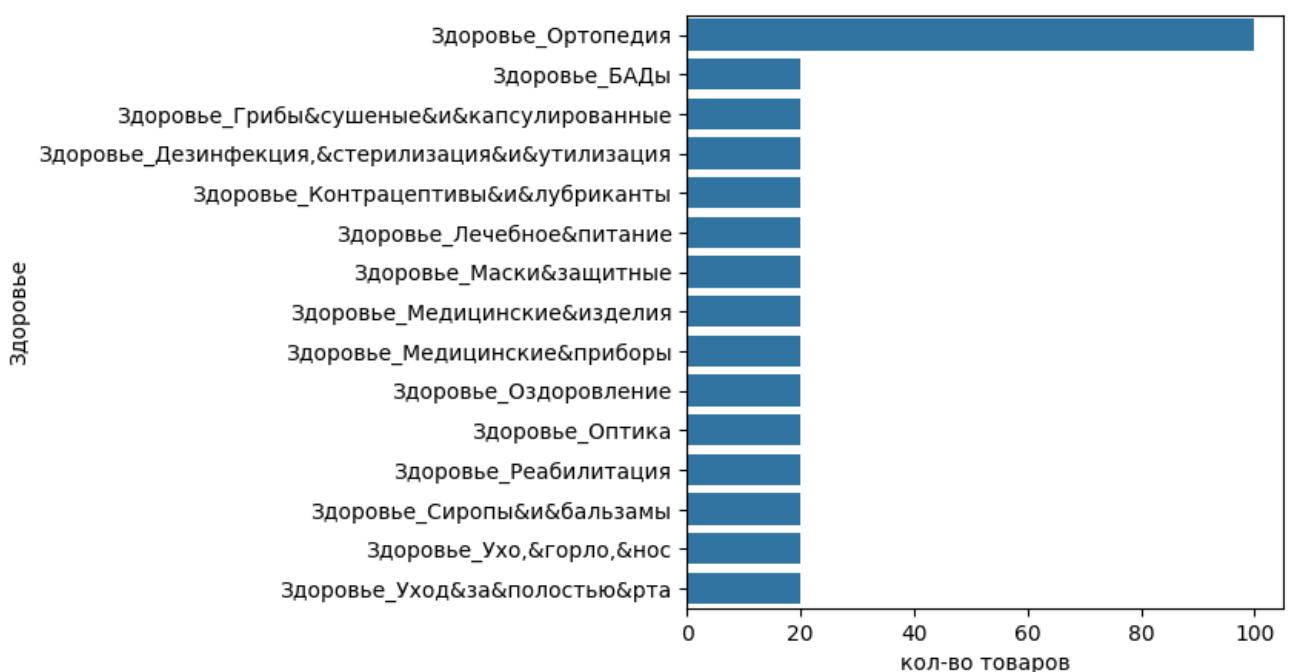


Рис. А.8: Распределение данных в категории «Здоровье».

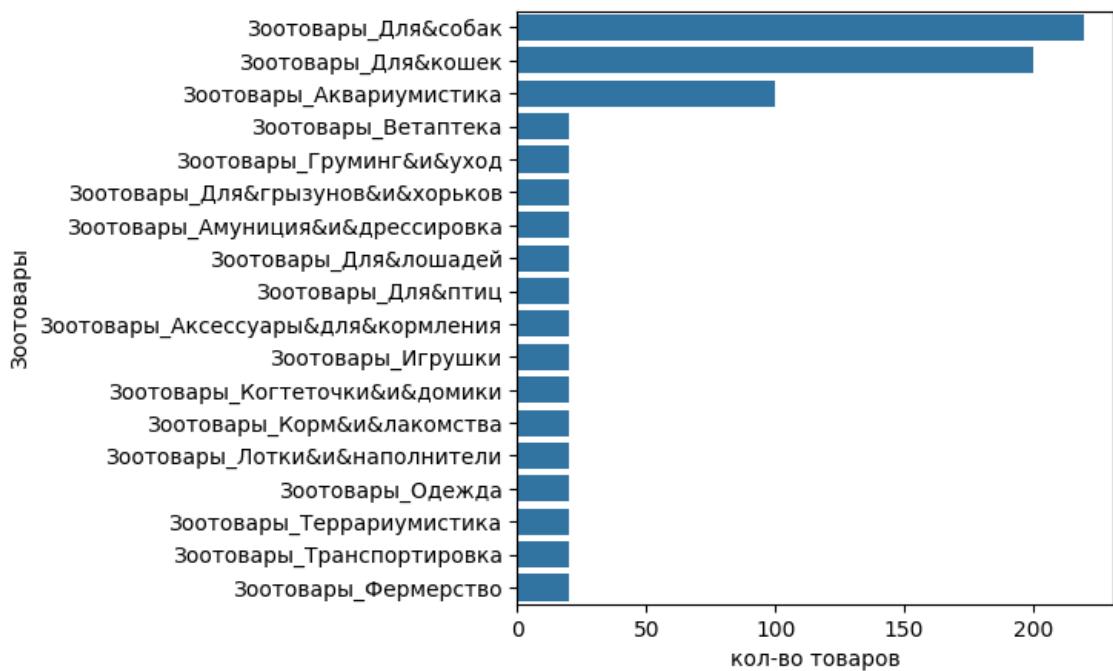


Рис. А.9: Распределение данных в категории «Зоотовары».

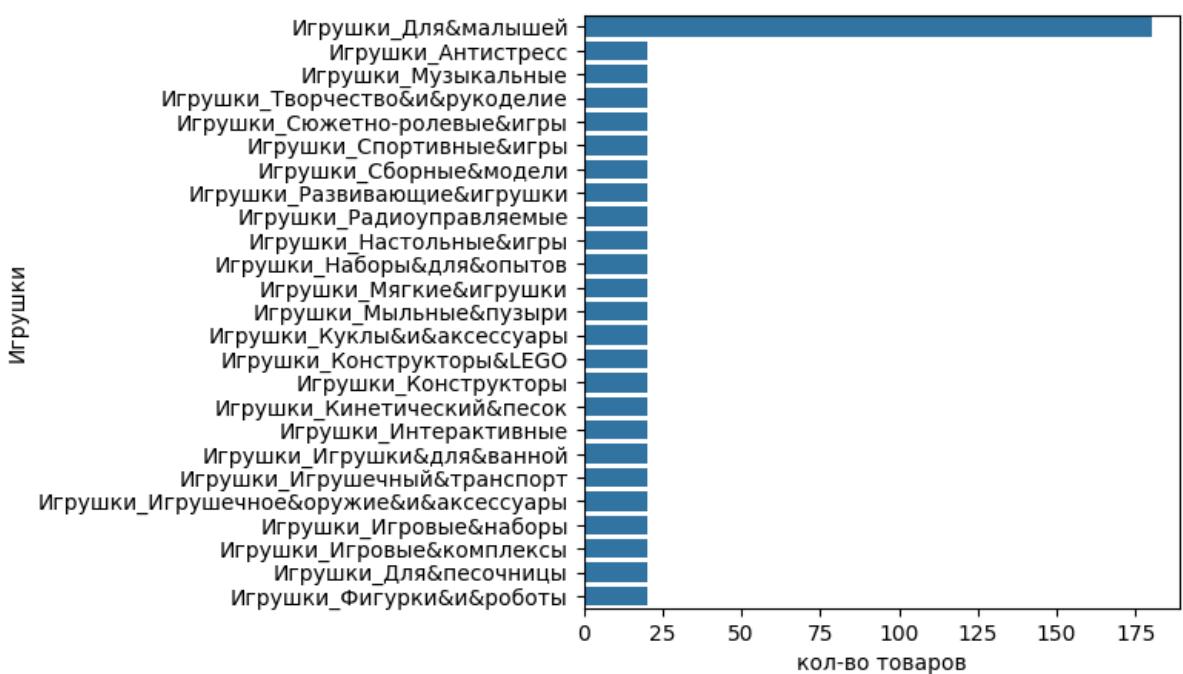


Рис. А.10: Распределение данных в категории «Игрушки».

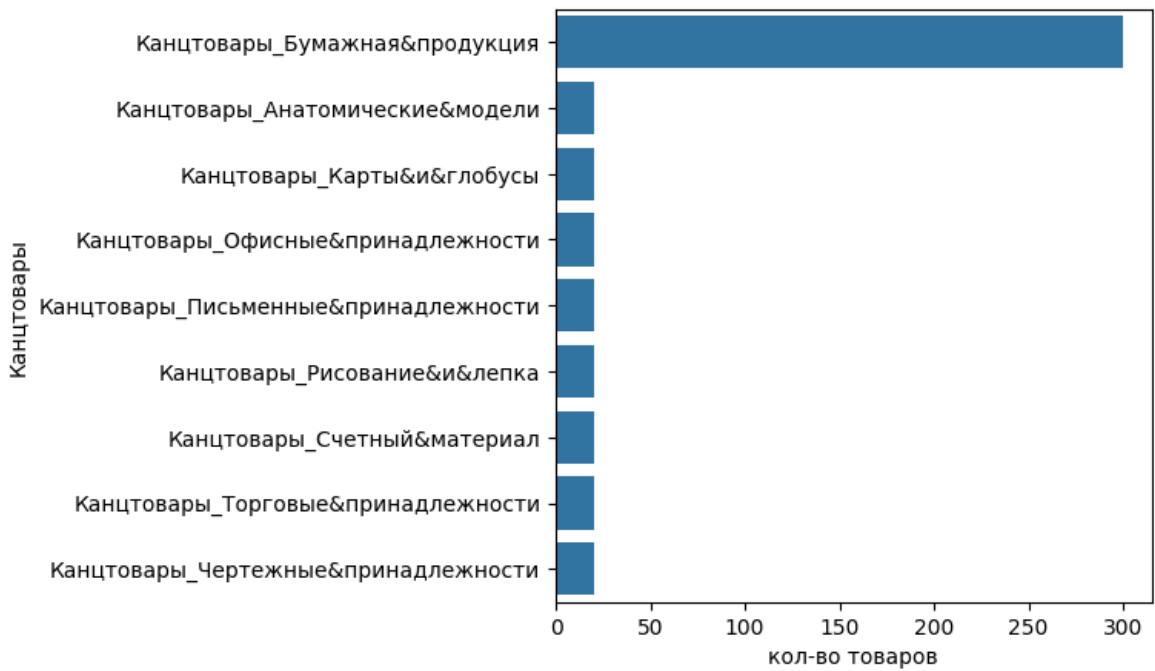


Рис. А.11: Распределение данных в категории «Канцтовары».

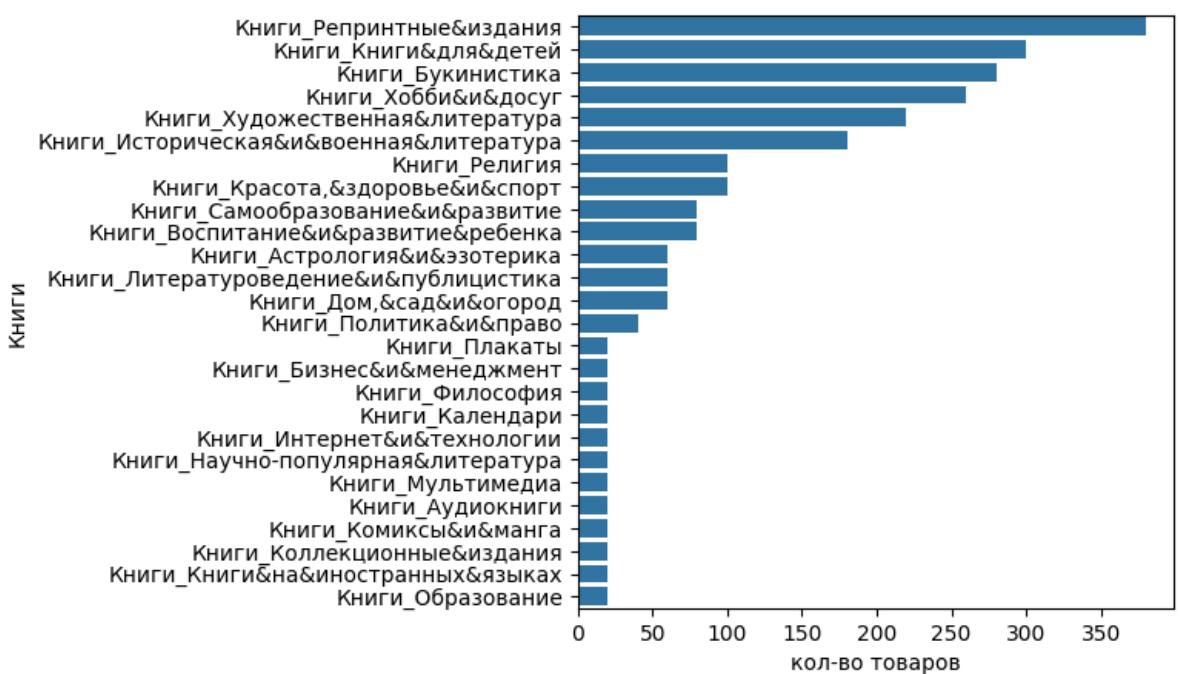


Рис. А.12: Распределение данных в категории «Книги».

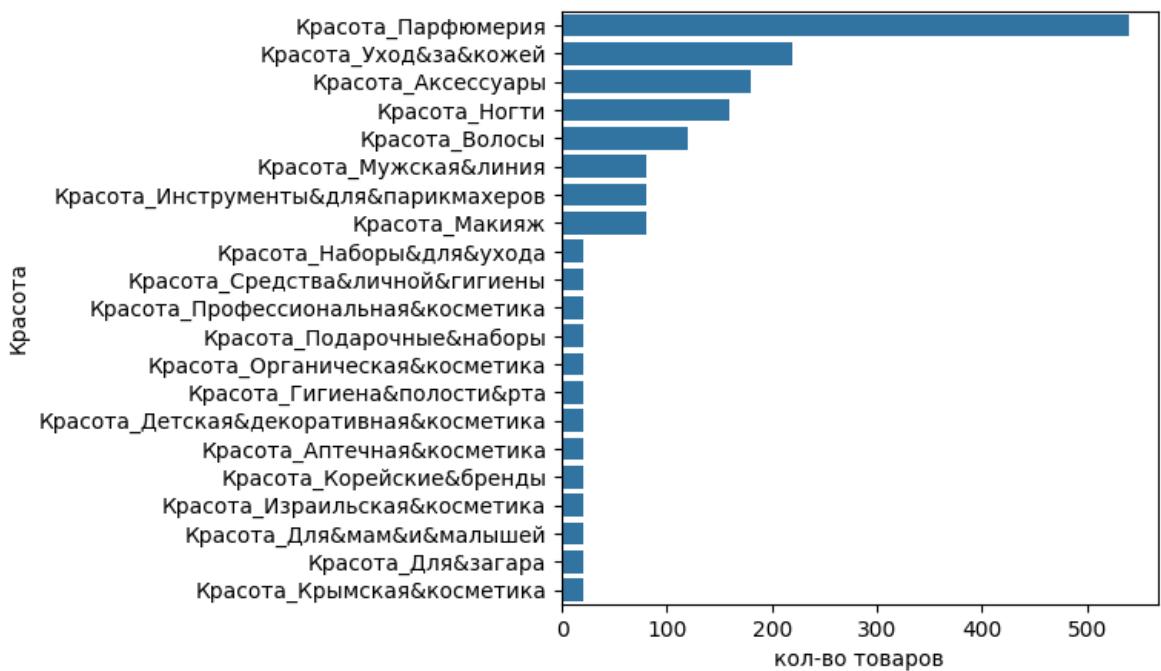


Рис. А.13: Распределение данных в категории «Красота».

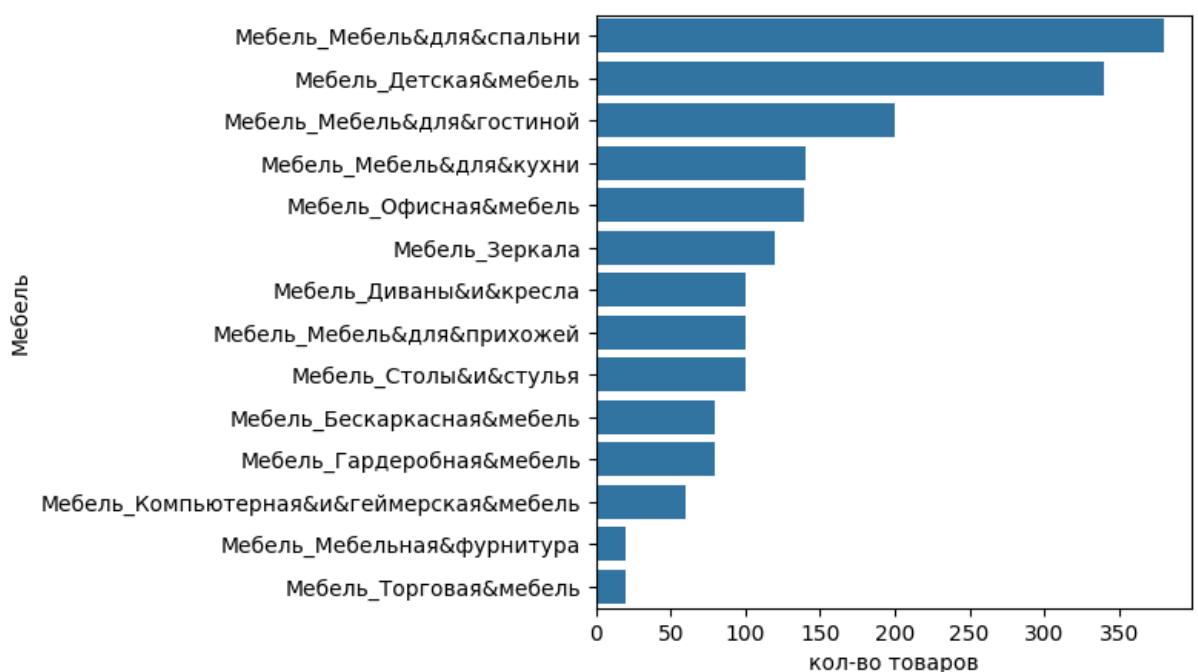


Рис. А.14: Распределение данных в категории «Мебель».

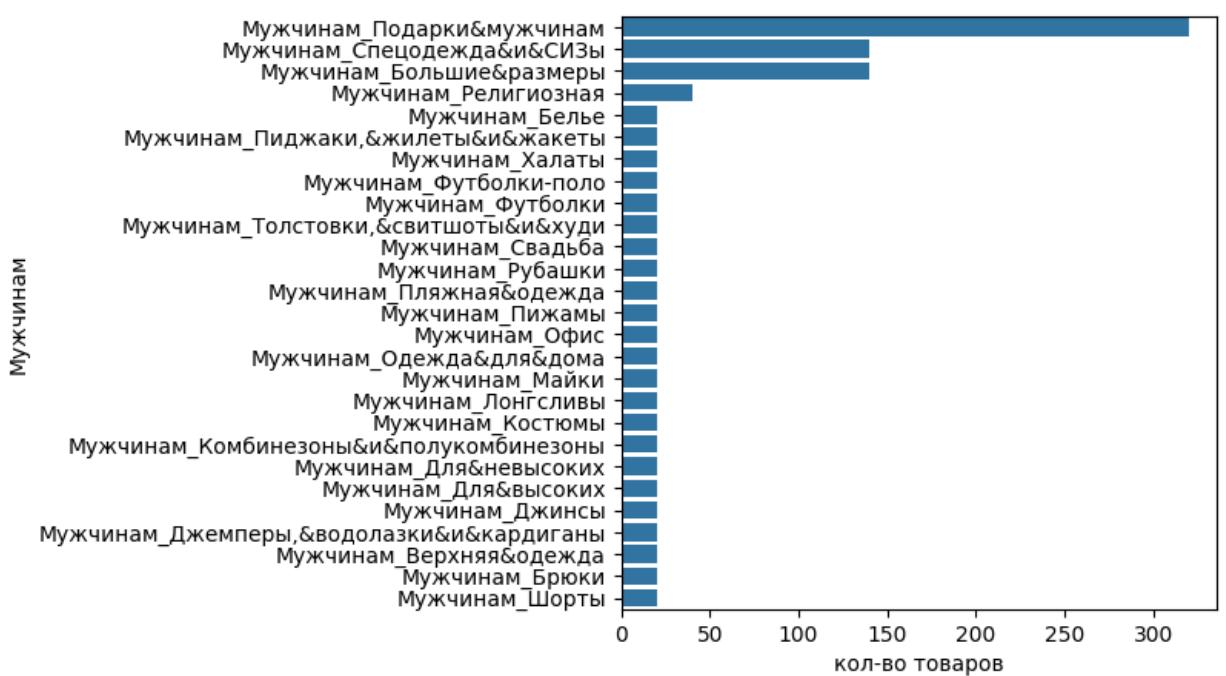


Рис. А.15: Распределение данных в категории «Мужчинам».

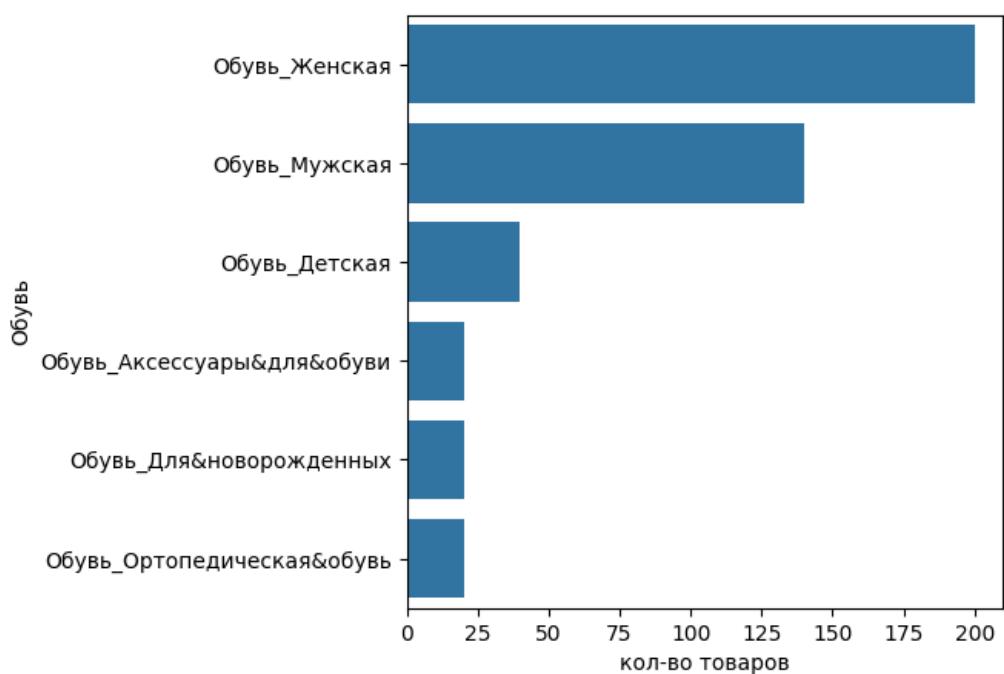


Рис. А.16: Распределение данных в категории «Обувь».

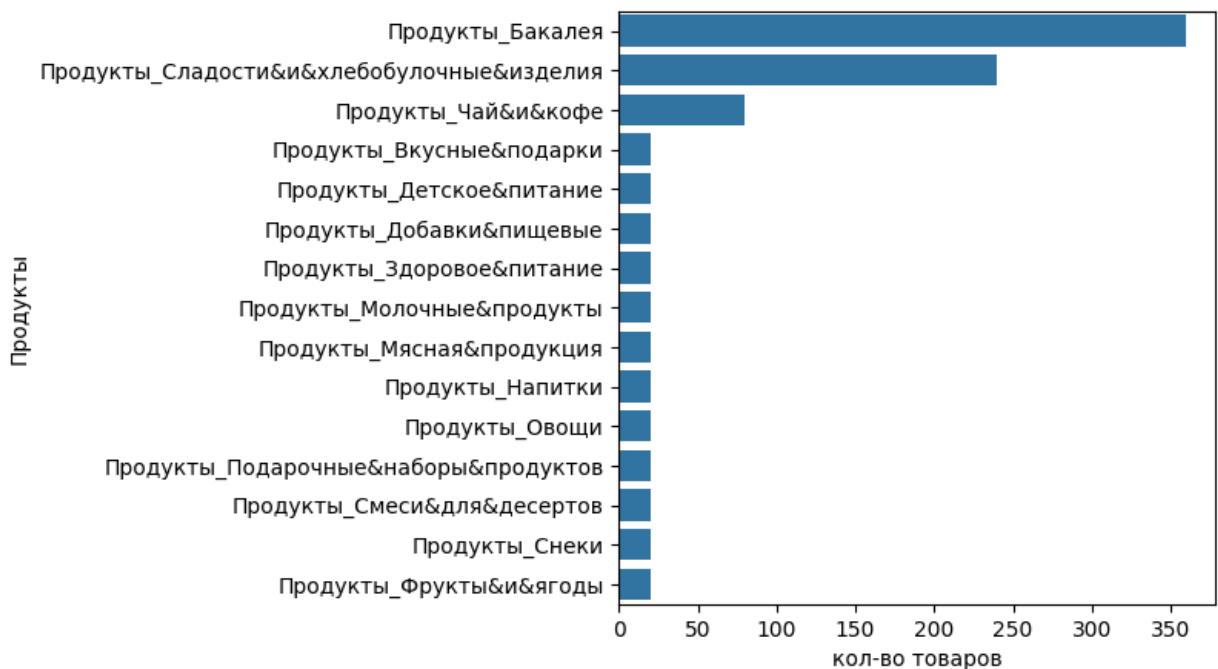


Рис. А.17: Распределение данных в категории «Продукты».

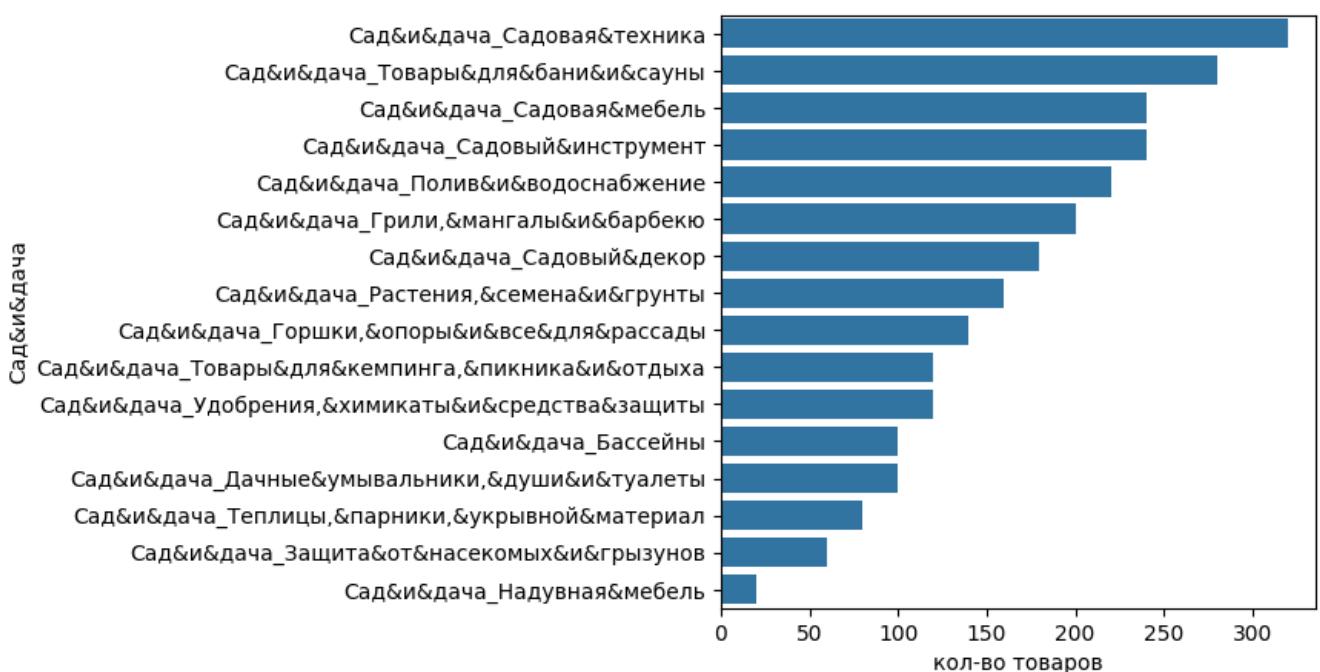


Рис. А.18: Распределение данных в категории «Сад&и&дача».

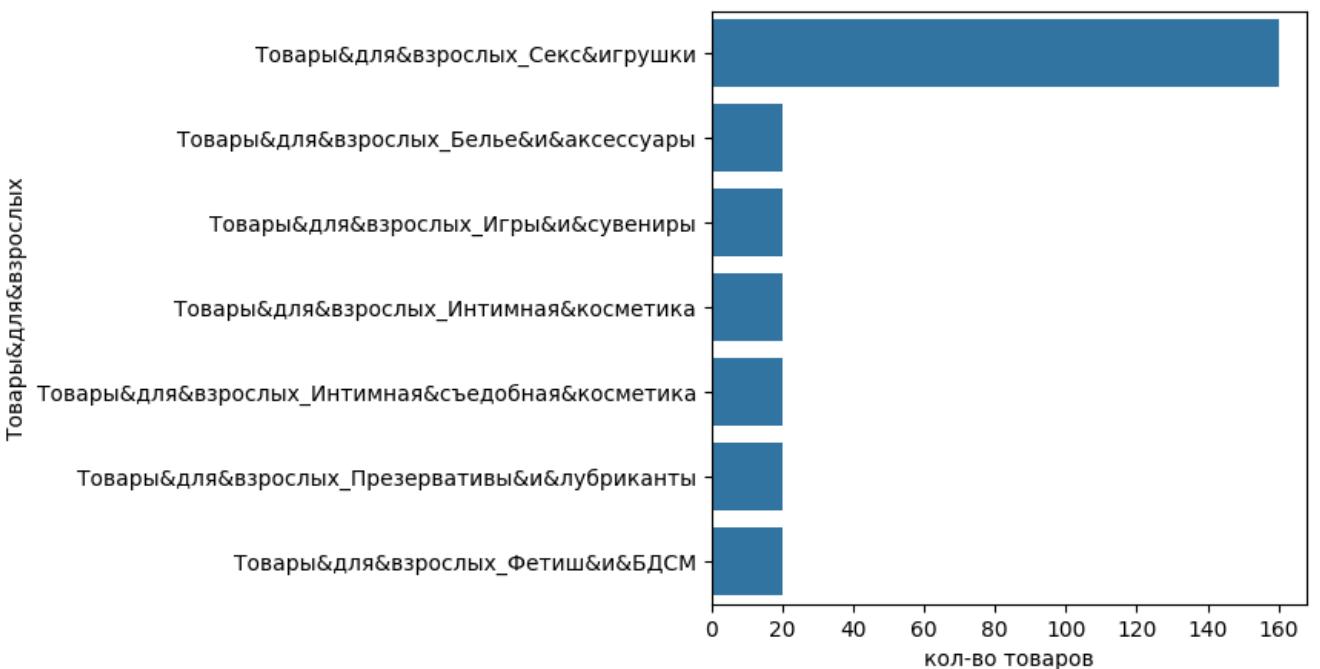


Рис. А.19: Распределение данных в категории «Товары&для&взрослых».

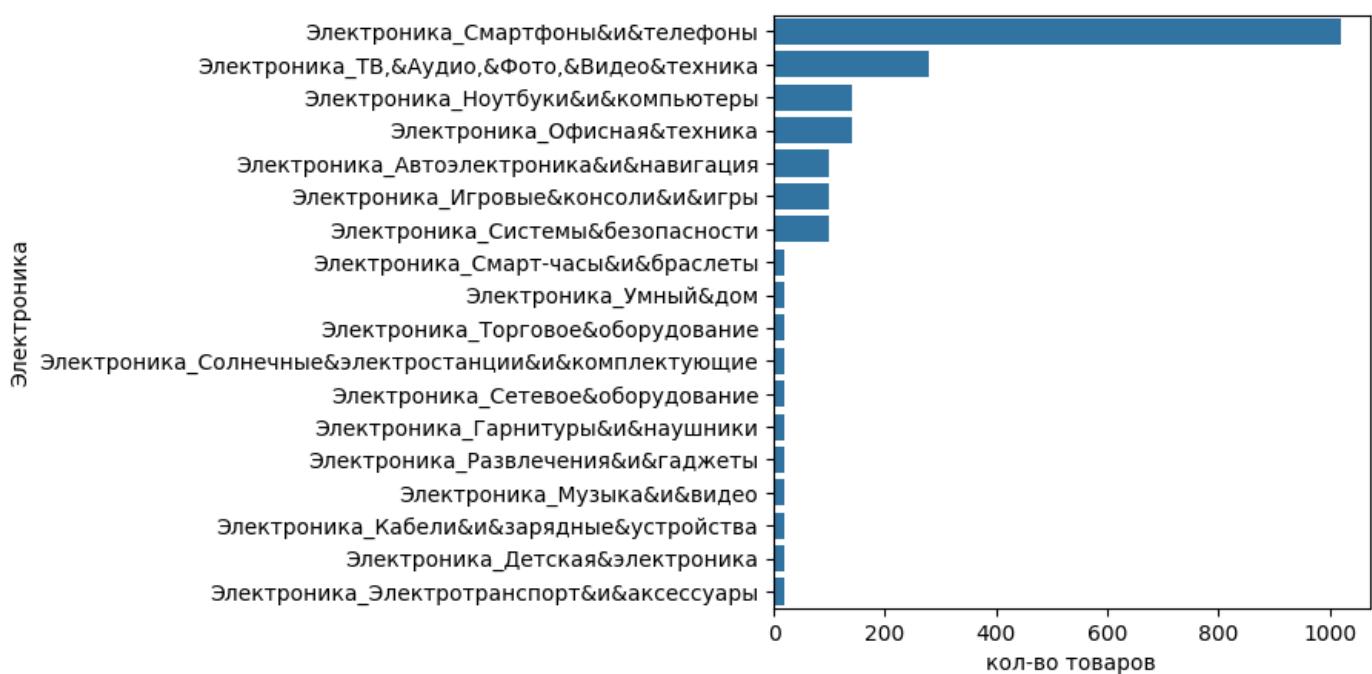


Рис. A.20: Распределение данных в категории «Электроника».

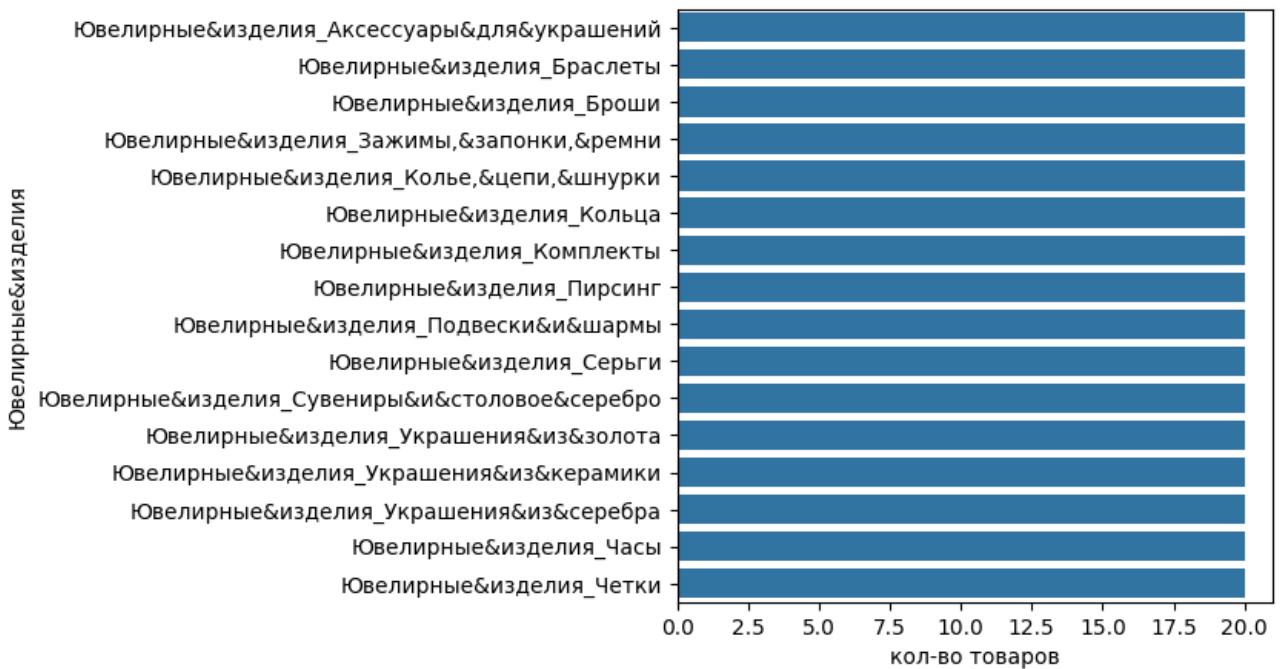


Рис. A.21: Распределение данных в категории «Ювелирные&изделия».