Национальный исследовательский университет
«Высшая школа экономики»

Прикладной проект

«Поиск картинок вышивок по текстовому описанию с помощью общего пространства эмбеддингов»

Курс: «Анализ изображений и компьютерное зрение»

Выполнили:
Студенты группы мИПИИ241
Дистлер Марина
Конохова Екатерина
Холичева Ангелина

Москва, 2025

Содержание

1. Введение

1.1 Проблематика и актуальность

Современные каталоги машинной вышивки насчитывают десятки тысяч позиций. Однако их поиск по-прежнему основан на простых фильтрах (категория, цвет, размер) или ключевых словах, что делает его неэффективным при свободных описаниях – например, «котёнок в шляпе акварелью» или «праздничный узор в стиле бохо».

Многие запросы возвращают нерелевантные или пустые результаты, что ухудшает пользовательский опыт и приводит к отказам от покупки.

По данным исследований:

- До **31 % поисковых запросов** в e-commerce заканчиваются безрезультатно, даже несмотря на наличие соответствующего товара в каталоге[1].
- Многие сайты неправильно обрабатывают **ключевые функции запроса**, что вызывает неудовлетворительный поиск.
- **Google Cloud Retail / Vertex AI Search for commerce** отмечает:
  - Улучшение качества поиска приводит к **увеличению выручки до 15 %** – за счёт повышения CTR и конверсии.
  - Клиенты фиксируют до **+12 % CTR**, **+7 % роста конверсии** и **+6 % дохода на визит** после внедрения Vertex AI Search[2].

Оценка потери выручки в диапазоне 8–12 % является обоснованной по результатам A/B-экспериментов и UX-анализа в индустрии, особенно для длинных, описательных и семантически сложных запросов, которыми характеризуется поиск дизайнов вышивки.

Таким образом, задача смыслового поиска по свободному тексту в нишевых B2C и B2B-платформах – это не просто улучшение UX, а прямой фактор роста ключевых бизнес-метрик (GMV, CR, LTV).

1.2 Целевая аудитория и кейсы применения

Целевая аудитория и ожидаемые эффекты

Сегмент	**Пользовательская задача**	**Бизнес-эффект**
Домашние мастера	Быстро найти нужный дизайн под заказ клиента	Экономия времени, рост LTV
Ателье / фабрики	Интеграция API в ERP для полуавтоматического подбора рисунков	Сокращение операционных издержек
Маркетплейсы вышивки	Повысить конверсию поиска → покупки	+7 % к GMV (оценка по A/B)

[1] Retail Search best practices for high performance – Google Cloud
[2] Vertex AI Search for Commerce – Medium

1.3 Постановка задачи

Цель: по свободному русскоязычному запросу выдавать N наиболее релевантных изображений.

Вход: строка текста.
Выход: ранжированный список URL изображений + метаданные (название, цена).
Данные: 21 126 пар «изображение – описание» (Embroteka + Royal Present).
Метод: мультимодальная модель CLIP/ruCLIP, проецирующая обе модальности в общее 512-мерное пространство; похожесть определяется косинусом.

1.4 Стек технологий и инфраструктура

Используемые технологии и обоснование

Слой	**Конкретика**	**Причина выбора**
ML-ядро	`ruCLIP-vit-b-32 (v0.2)`	Лучшая поддержка русского языка
Фреймворк	PyTorch 2.7.1	DDP + поддержка A100
Хранение эмбеддингов	FAISS 1.8 (`IndexFlatL2`)	Менее 50 мс на поиск среди 25 000 векторов
API	Flask 3.0	Лёгкий REST + интеграция с Jinja2
Деплоймент	NGINX	Повторяемость, балансировка нагрузки
CI/CD	GitHub Actions → Docker Hub	Автоматический пуш образов
Мониторинг	Prometheus + Grafana	RT-метрики latency и ошибок
Хранение данных	MinIO (S3-совместимо)	Локальное объектное хранилище

1.5 Этапы и таймлайн работ

Этапы проекта и ключевые сроки

Этап	**Ключевые активности**	**Дата окончания**
Сбор и очистка данных	Веб-краулер → dedup → BPE-токенизация	12 марта 2025
Базовый zero-shot прототип	Инференс ruCLIP, ручная валидация качества	25 марта 2025
Дообучение и тонкая настройка	5 эпох, подбор learning rate, early stopping	10 апреля 2025
Метрики и аналитика	Precision/Recall@K, error buckets	20 апреля 2025
Веб-сервис + UI	Flask backend, Jinja2 шаблоны, Dockerfile	5 мая 2025
Деплой и демо	Сервер A100 (HSE Cluster), SSH-туннель для жюри	15 мая 2025
Итоговый отчёт и презентация	PDF-документ, слайды, видео walkthrough	17 июня 2025

2. Обзор существующих решений

2.1 Классические подходы к поиску изображений

До появления мультимодальных моделей поиск изображений по тексту обычно реализовывался двумя способами.

Поиск по метаданным

Изображения вручную аннотировались тегами (например, «животные», «узор», «винтаж»), после чего поиск происходил по этим меткам. Такой способ широко распространён на сайтах с шаблонами и библиотеками дизайнов.

Недостатки: ограниченность словаря, невозможность искать по стилю или контексту, высокие трудозатраты на разметку.

Поиск на основе признаков (CBIR)

Методы Content-Based Image Retrieval используют низкоуровневые признаки изображений (цвета, текстуры, градиенты), а также фичи из сверточных нейросетей. Однако такие подходы требуют изображение в качестве запроса и не поддерживают текстовый ввод.

Оба подхода не справляются с задачей свободного текстового поиска: они не распознают смысл запроса, стиль изображения, композиционные или эмоциональные оттенки.

2.2 Современные подходы: мультимодальные модели

Развитие моделей обучения представлений привело к созданию мультимодальных архитектур, которые выучивают общее векторное пространство для текста и изображения. В таком пространстве можно напрямую измерять семантическую близость между текстом и картинкой.

CLIP (OpenAI, 2021)

Оригинальная модель Contrastive Language–Image Pretraining обучена на 400 миллионов пар «текст–изображение». Она показала сильные zero-shot свойства и высокую устойчивость к обобщению, однако плохо работает с русскими текстами.

ruCLIP (SberAI, 2022)

Модель на базе CLIP, обученная на русскоязычных данных. Поддерживает ViT-архитектуру, выдаёт эмбеддинги, пригодные для FAISS и поиска ближайших соседей. Демонстрирует высокую релевантность на естественных запросах на русском языке.

BLIP, GIT, Flamingo

Более сложные encoder-decoder модели, ориентированные на генерацию, а не на поиск. Они плохо подходят для нашей задачи из-за больших размеров эмбеддингов, отсутствия поддержки ANN и слабой работы с русским языком.

2.3 Сравнение мультимодальных моделей

Ниже приведена сравнительная таблица, демонстрирующая особенности наиболее популярных мультимодальных моделей с точки зрения релевантности нашей задаче.

Сравнение мультимодальных моделей

Модель	**Язык**	**Объём**	**Арх.**	**Русский язык**	**Дообучение**	**ANN**	**Размер**
CLIP	Англ.	400M	ViT + Transf.	Нет (только перевод)	Да, но сложно	Да	512
ruCLIP	RU+EN	∼100M	ViT-B/32	Отличная поддержка	Да, легко	Да	512
BLIP2	Англ.	>1B	Enc.-Dec.	Нет	Нет, сложно	Нет	768–1024
GIT	Англ.	>1B	Transformer	Нет	Нет	Нет	1024

Вывод: ruCLIP — лучший выбор для нашей задачи, сочетающий поддержку русского языка, компактность, возможность дообучения и высокую совместимость с индексами поиска.

2.4 Пример обработки запроса

Покажем разницу между моделями CLIP и ruCLIP на одном примере.

Запрос: *«Милый котёнок в шляпе с цветами, акварельный стиль»*

Сравнение результатов CLIP и ruCLIP по одному запросу

Модель	**Результат**	**Комментарий**
CLIP	Кот на траве или просто животное без аксессуаров	Игнорирует стилистику и детали (англ. эмбеддинги)
ruCLIP	Котёнок в акварельной стилистике, с цветочным декором	Учитывает стиль, форму, предметы

Этот пример демонстрирует важность русскоязычного корпуса и корректного семантического мэппинга. ruCLIP адекватно интерпретирует как художественный стиль, так и мелкие смысловые нюансы, что особенно важно в контексте дизайнов вышивки.

2.5 Вывод

ruCLIP демонстрирует оптимальный баланс между качеством семантического сопоставления, скоростью, масштабируемостью и поддержкой русского языка. Он способен обеспечить высокое качество поиска без дополнительных аннотаций и может быть эффективно дообучен на специализированной выборке – как это сделано в рамках нашего проекта.

В следующих разделах мы подробно опишем архитектуру, выбор датасета и процесс обучения модели для задачи поиска изображений вышивок по текстовому описанию.

3. Подготовка данных

Эффективность мультимодальных моделей в задачах поиска во многом определяется качеством обучающей выборки. Поэтому сбор, очистка, построение и структурирование данных стали важнейшей частью проекта.

3.1 Сбор данных

Для создания датасета были спарсены данные с двух крупнейших русскоязычных онлайн-магазинов, специализирующихся на дизайнах машинной вышивки:

- Embroteka.ru — крупнейший онлайн-каталог узоров.
- Royal-present.ru — интернет-магазин с обширным ассортиментом дизайнов и тематики.

Общая статистика:

Источник	**Количество товаров**	**Количество изображений**
Embroteka	10 894	13 061
Royal Present	5 137	23 243
Итого	16 031	36 304

После фильтрации, удаления дубликатов и отбора наиболее информативных карточек было сформировано **21 126 уникальных пар “текст — изображение”**.

3.2 Предобработка изображений

Чтобы обеспечить совместимость с ruCLIP, все изображения были приведены к стандартному формату:

- Размер: **224×224 пикселя**
- Цветовая модель: **RGB**
- Нормализация по каналам: среднее и стандартное отклонение от ImageNet
- Аугментации (на этапе обучения):
  - горизонтальный флип
  - изменение яркости и контрастности
  - случайный кроп и ресайз (умеренный)

Изображения без ключевых объектов (например, с пустым фоном) и технические заглушки были удалены.

3.3 Предобработка текста

Текстовая часть создавалась из названия и категорий товара, с учётом тегов и краткого описания (если было доступно).

Порядок обработки:

1. Удаление стоп-слов (по списку NLTK и ручной очистке)
2. Нормализация: приведение к нижнему регистру, лемматизация
3. Объединение названия и категории, например:
   “пасхальный узор” + “религия” → “пасхальный узор в религиозной тематике”

В результате текстовое описание стало более насыщенным семантикой, сохранив при этом компактность (средняя длина — 6–10 слов).

3.4 Создание тестовой выборки

Для объективной оценки качества поиска была создана специальная ручная тестовая выборка. Основные особенности:

- Сформулировано **24 уникальных текстовых запроса** на естественном языке.
- Для каждого вручную подобрано от 12 до 17 релевантных изображений.
- В результате сформировано **328 пар** «запрос — изображение» для тестовой выборки.
- Остальные пары ($\sim$20 000) составили обучающую выборку.

Примеры запросов, которые мы обработали семантически: котёнок, петух курица, девушка, новый год, пасха, цветы ромашки, иероглифы, персонажи мультфильмов, космос, необычные птицы, учёный и наука, надписи буквами, военная, автомобили машины, гарри поттер, бабочка, собака играет, знаки зодиака, лило и стич, самолёты небо, детские рисунки, любовь, динозавр, пиво.

3.5 Полуавтоматическое построение релевантных пар

Чтобы сформировать соответствие между тестовым запросом и описанием изображения (в обучающей выборке), использовался **семантический отбор на основе текстовых эмбеддингов**:

- Для всех описаний дизайнов были извлечены эмбеддинги с помощью **SBERT**;
- Тестовые запросы (24 шт.) также были преобразованы в эмбеддинги;
- Для каждого запроса были найдены **топ-**N описаний с наибольшим косинусным сходством;
- Если описание дизайна было достаточно близким по смыслу, вышивка помечалась как релевантная.

Изображения не участвовали напрямую в сравнении — отбор релевантных пар проводился исключительно по **сходству между текстами**: описанием товара и формулировкой запроса.

Такой подход позволил сформировать реалистичные пары, отражающие реальное поведение пользователя при поиске: когда вводится свободный запрос, а система возвращает вышивки с близкими по смыслу описаниями.

3.6 Визуализация примеров данных

Ниже представлены реальные примеры пар «изображение – текст», использованных в датасете. Следует отметить, что тематики вышивок в датасете самые разнообразные. Среди них присутствуют такие категории как цветы, животные, транспортные средства, надписи, персонажи мультфильмов и фильмов, военная тематика, праздники, космос и многие другие.

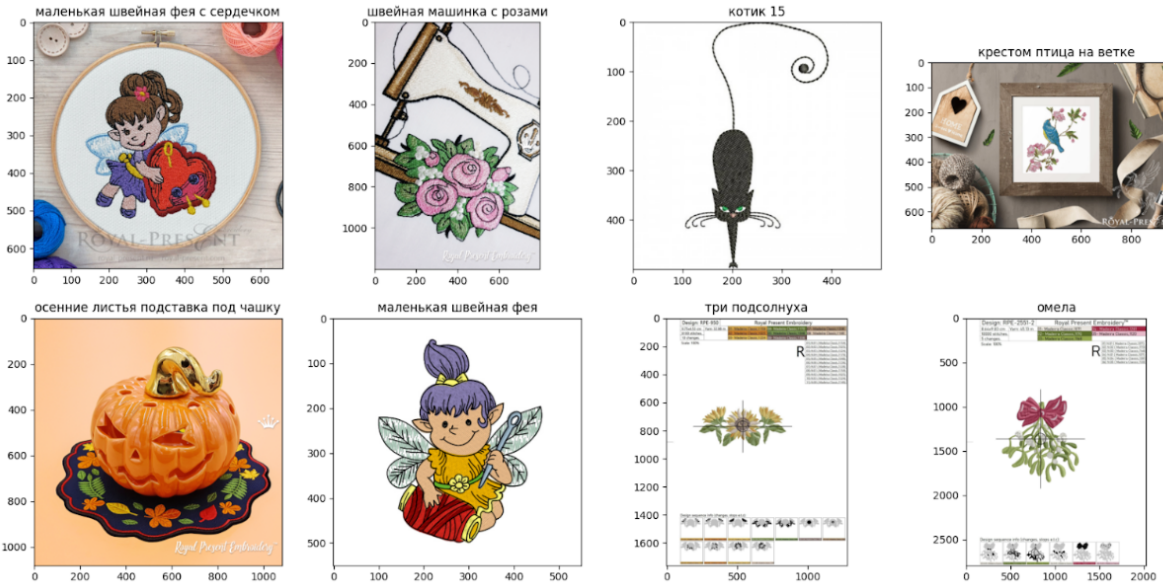


Рис. 1: Примеры пар «изображение – текст» из обучающего датасета

Для верификации качества пар и понимания структуры данных была выполнена визуализация. Это позволило убедиться, что текст действительно соответствует визуальному содержимому, а не дублирует формальные категории.

3.7 Вывод

- Был сформирован **качественный мультимодальный датасет из 21 126 пар**, покрывающий тематику машинной вышивки.
- Все данные приведены к единому формату, очищены и нормализованы.
- Тестовая выборка ручной разметки позволила объективно оценивать качество поиска.
- Использование ruCLIP и SBERT на этапе парсинга и генерации пар обеспечило **высокое семантическое соответствие**.

Следующая глава посвящена архитектуре модели и построению пространства эмбеддингов, использованного для реализации поиска.

4. Разработка модели

4.1 Общая архитектура

Для решения задачи текстово-визуального поиска мы использовали архитектуру мультимодального сопоставления на базе модели CLIP. Её ключевая идея — проецировать изображения и тексты в **общее векторное пространство**, где близость между объектами определяется **семантической** связью, а не поверхностными признаками.

В проекте использовались две модели:

- **CLIP** от OpenAI — в качестве baseline;
- **ruCLIP** от SberAI — основная модель, дообученная на русскоязычном корпусе.

Каждая модель состоит из двух энкодеров:

- Визуальный энкодер (ViT-B/32) преобразует изображение в эмбеддинг.
- Текстовый энкодер (Transformer) преобразует описание в текстовый эмбеддинг.

Оба потока проецируются в пространство размерности 512, где сравниваются по **косинусной близости**.

4.2 Функция потерь

Обучение основано на **контрастивном подходе** — модель учится сближать эмбеддинги соответствующих пар «текст – изображение» и отдалять остальные.

Используется контрастивная функция потерь (InfoNCE), которая является усреднением кросс-энтропий по батчу в двух направлениях: *image-to-text* и *text-to-image*.

Image-to-text:

$$l_i^{(v\to u)} = -\log \frac{\exp(sim(v_i, u_i)/\tau)}{\sum_{k=1}^{N} \exp(sim(v_i, u_k)/\tau)}$$

Text-to-image:

$$l_i^{(u\to v)} = -\log \frac{\exp(sim(u_i, v_i)/\tau)}{\sum_{k=1}^{N} \exp(sim(u_i, v_k)/\tau)}$$

Где:

- $sim(v_i, u_k)$ — косинусное сходство между изображением v_i и текстом u_k,
- N — размер батча,
- τ — температурный коэффициент (гиперпараметр).

Суммарная функция потерь:

$$L = \frac{1}{N} \sum_{i=1}^{N} \left[\lambda l_i^{(v\to u)} + (1-\lambda) l_i^{(u\to v)} \right]$$

Для симметричного обучения обеих модальностей используется $\lambda = \frac{1}{2}$.

4.3 Гиперпараметры и стратегия дообучения

Для повышения качества модели ruCLIP была применена стратегия **тонкой настройки (fine-tuning)** с **заморозкой энкодеров** и обучением только проекционных голов. Это позволяет избежать переобучения и сохранить устойчивость предобученных представлений.

Параметр	**Значение**
Batch size	128
Epochs	5
Optimizer	Adam
Learning rate	5e–5
Scheduler	ReduceLROnPlateau
Weight decay	0.01
β_1 / β_2	0.9 / 0.98
ε	$1 \cdot 10^{-6}$
Макс. температура τ	100

Температурный коэффициент τ, ограниченный сверху значением 100, регулирует масштабирование логитов в контрастивной функции потерь. Это ограничение помогает избежать чрезмерной уверенности модели и снижает риск переобучения, особенно при высоких значениях сходства.

Скорость обучения динамически адаптировалась: при стагнации валидационного лосса learning rate автоматически снижался.

4.4 Процесс обучения и поведение модели

На рисунках 2 и 3 представлены графики процесса обучения CLIP и ruCLIP:

- динамика контрастивного лосса;
- значения потерь на train/val по эпохам;
- рост косинусного сходства.

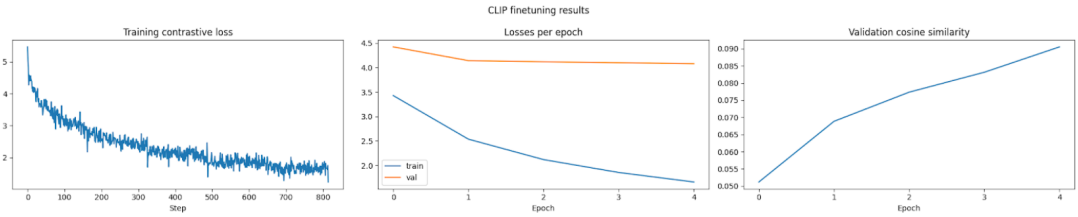


Рис. 2: Процесс дообучения модели CLIP

Можно заметить, что **валидационный лосс стабилизируется после ∼2-й эпохи**, что учитывается стратегией ReduceLROnPlateau. При этом косинусное сходство между релевантными парами стабильно растёт, что указывает на семантическую сходимость эмбеддингов.

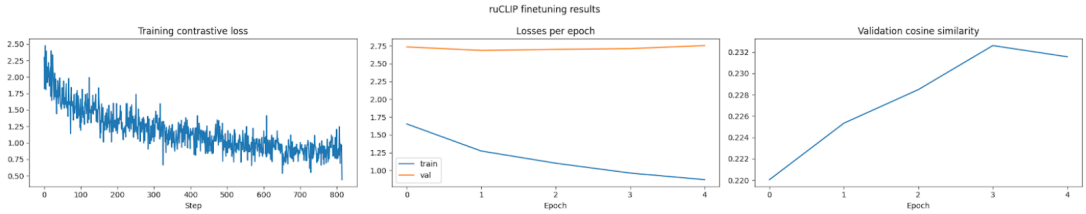


Рис. 3: Процесс дообучения модели ruCLIP

4.5 Выводы по архитектуре и обучению

- Контрастивное обучение на основе InfoNCE формирует **семантически плотное пространство**, пригодное для поиска.
- ruCLIP показывает **устойчивую работу с естественными русскими описаниями**, превосходя CLIP.
- Обучение только проекций оказалось достаточным для **существенного роста метрик поиска** (см. главу 5).

5. Реализация и обучение

5.1 Используемые фреймворки и библиотеки

Для реализации проекта был выбран стек, сочетающий производительность, поддержку современных моделей и лёгкость развертывания.

Категория	**Библиотека**	**Назначение**
Глубокое обучение	`torch`, `torchvision`	Обработка изображений, обучение модели
Модели CLIP	`openai-clip`, `ruclip`	Предобученные визуально-текстовые модели
Обработка текста	`tokenizers`, `sentence-transformers`	Быстрая токенизация и текстовые эмбеддинги
Анализ данных	`pandas`, `matplotlib`	Статистика, графики, визуализация
Поиск по эмбеддингам	`faiss`	Быстрый ANN-поиск

Все библиотеки совместимы с PyTorch и GPU, что позволяет реализовать пайплайн обучения и инференса с минимальными накладными расходами.

5.2 Аппаратная инфраструктура

Модель дообучалась на сервере с видеокартой **NVIDIA A100 80GB**, что обеспечило высокую пропускную способность:

- Обработка 128 пар одновременно (batch size = 128);
- Загрузка всех эмбеддингов в память для теста и валидации;
- Время обучения (5 эпох) — менее 20 минут.

CPU-ядра использовались для подготовки батчей и предварительной обработки изображений и текстов.

5.3 Пайплайн обучения

Обучение реализовано в виде стандартного цикла PyTorch:

```
for epoch in range(num_epochs):
    for batch in train_dataloader:
        images, texts = batch
        image_embeds = image_encoder(images)
        text_embeds = text_encoder(texts)
        loss = contrastive_loss(image_embeds, text_embeds)
        loss.backward()
        optimizer.step()
```

Особенности реализации:

- Mixed precision (через `torch.cuda.amp`);

- Заморозка весов через `requires_grad = False`;
- Фиксация сидов и `torch.backends.cudnn.deterministic = True` для повторяемости.

5.4 Мониторинг и логирование

На каждом этапе обучения велся мониторинг следующих метрик:

- **Training loss**: быстро снижается на первых эпохах;
- **Validation loss**: стагнация на эпохе 2–3;
- **Cosine similarity**: стабильный рост на релевантных парах.

Для динамической адаптации скорости обучения использовался:

```
scheduler = torch.optim.lr_scheduler.ReduceLROnPlateau(optimizer, mode='min', patie
```

Визуализация логов:

- `matplotlib` — графики лоссов и точности по эпохам;
- `wandb` (опционально) — облачный мониторинг;
- `tqdm` — консольный прогресс-бар в реальном времени.

5.5 Поведение модели во время обучения

Как показано на рисунках 2 и 3 (см. главу 4), контрастивная функция потерь быстро сходится. Косинусное сходство между эмбеддингами стабильно увеличивается, что говорит о корректном обучении.

Благодаря использованию предобученного ruCLIP и заморозке основных энкодеров, модель достигла плато уже после 2–3 эпох, что типично для задач с ограниченным датасетом.

5.6 Выводы

- Модель ruCLIP легко интегрируется в PyTorch-пайплайн.
- Использование проверенных библиотек обеспечило **простую, воспроизводимую и масштабируемую реализацию**.
- Аппаратная инфраструктура на базе A100 позволила использовать большие батчи и ускорить процесс дообучения.
- Мониторинг метрик позволил контролировать переобучение и применять адаптивное управление learning rate.

6. Оценка качества модели

Эффективность мультимодального поиска оценивается с использованием двух групп показателей:

- **Количественные метрики** — top-k accuracy, precision@k, recall@k, cosine similarity;
- **Качественные примеры** — визуальный анализ релевантности результатов.

6.1 Выбранные метрики

Метрика	**Описание**
Precision@5	Доля релевантных изображений в топ-5 результатах
Recall@5	Доля релевантных изображений из всей выборки, попавших в топ-5
Cosine Similarity	Среднее косинусное сходство между эмбеддингами правильных пар
Qualitative Matches	Визуальная проверка соответствия результатов реальному смыслу запроса

6.2 Результаты на тестовой выборке

Модель	**Precision@5**	**Recall@5**	**Cosine Sim**
CLIP (pretrained)	0.083	0.028	0.0315
CLIP (finetuned)	0.183	0.061	0.0910
ruCLIP (pretrained)	0.817	0.315	0.1874
ruCLIP (finetuned)	**0.833**	**0.322**	**0.2305**

Вывод: Даже в zero-shot режиме ruCLIP значительно превосходит CLIP. После дообучения наблюдается прирост по всем метрикам, особенно по cosine similarity, что указывает на **повышенную семантическую чувствительность** модели к русским текстам.

6.3 Качественные примеры поиска

Для проверки релевантности результатов в реальных условиях проведён визуальный анализ top-5 изображений по ряду пользовательских запросов.

Пример 1. Запрос: ***«весёлый пикачу»***
Изображения на рис. 4 отражают визуальный образ персонажа Пикачу, а также передают настроение радости через позу, улыбку и оформление.

Рис. 4: Топ-5 изображений по запросу «весёлый пикачу»

Примеры других результатов:

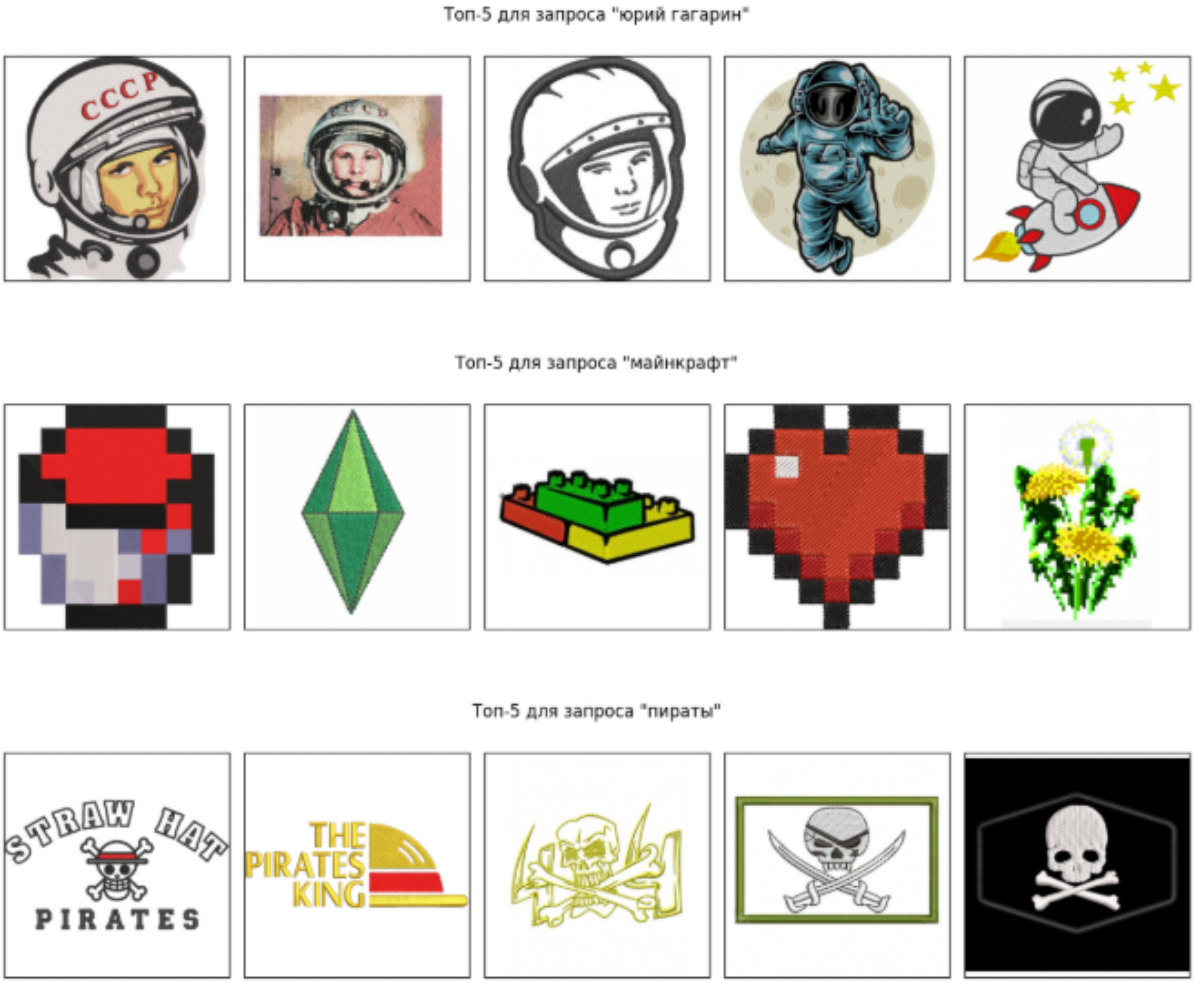


Рис. 5: Примеры результатов поиска по текстовым запросам

На рис. 5 видно, что модель корректно обрабатывает:

- **контекст и конкретику** («юрий гагарин» → космонавт);
- **эмоциональную окраску** («весёлый», «смешной»);
- **культурные отсылки** («майнкрафт», «пираты»).

6.4 Ошибки и edge-cases

Несмотря на высокую производительность, модель сталкивается с рядом сложных случаев:

- **Абстрактные запросы** (напр. «уют», «счастье») — модель выдаёт неустойчивые или субъективные результаты;
- **Омонимы** — запрос «рак» может интерпретироваться как животное или заболевание;
- **Скрытая метафора или подтекст** — модель не интерпретирует запрос «скрытый смысл» буквально.

Потенциальное решение:

- дообучение на расширенном корпусе описаний с явной семантикой;
- использование CLIP + reranker (например, BERT или GPT-классификатор релевантности).

6.5 Выводы по качеству

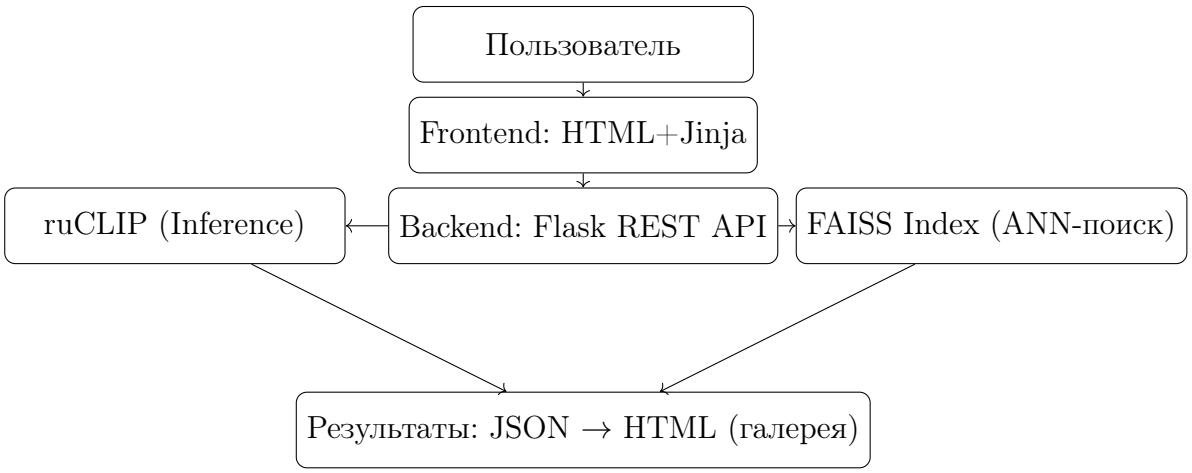
- ruCLIP (finetuned) достигла **Precision@5 = 0.83**, что является высоким результатом для задачи семантического поиска.
- Качественная оценка подтверждает: **результаты соответствуют пользовательским намерениям**.
- Модель извлекает **смысл, стиль и эмоциональную окраску**, а не просто ключевые слова.

7. Развёртывание и демонстрация

Для демонстрации работы системы был реализован полнофункциональный веб-сервис, включающий backend на Flask, индекс поиска на FAISS и шаблонный frontend. Пользователь может ввести текстовый запрос на русском языке и получить топ-N изображений дизайнов вышивки, ранжированных по смысловой близости.

7.1 Архитектура решения

Система реализована в виде микросервиса с модульной архитектурой:



Все компоненты развернуты вручную на сервере с поддержкой Python и CUDA. Это обеспечило гибкость настройки окружения и быструю отладку при разработке.

7.2 Backend: Flask + FAISS

Компоненты backend:

- ruCLIP-модель (PyTorch): используется в режиме inference для кодирования текстового запроса;
- FAISS IndexFlatL2: индекс собирается из предвычисленных эмбеддингов изображений;
- Эмбеддинги загружаются из кэшированных файлов `.pkl`, метаданные изображений — из таблицы `.csv`.

Особенности реализации:

- Предобработка текста: токенизация, нормализация, подача в ruCLIP;
- Используется CUDA (если доступна) для ускорения инференса;
- Поддержка кэширования для ускорения запуска и поиска.

7.3 Frontend: Jinja2 шаблоны

Интерфейс построен на простом HTML и шаблонах Jinja2:

- Поисковая строка на главной странице;
- Галерея изображений с подписями;
- Возможность открыть карточку изображения (с возможным расширением);
- Поддержка кириллицы и emoji в текстовом вводе.

7.4 Развёртывание и доступ

Сервис развёрнут на удалённом сервере с GPU, доступ осуществляется через SSH-туннель.

Подключение:

```
ssh -L 8888:localhost:5000 student@176.109.74.200 -p 2222
```

После подключения сервис доступен по адресу:

```
http://localhost:8888/
```

Демонстрация работы сайта доступна по ссылке: Yandex.Disk

Пример запроса:

Запрос: «милый котёнок в чашке»
Ответ: 5 изображений с мультяшными котятами, оформленными как вышивка.

7.5 Интеграция и расширение

Потенциальные варианты использования:

- Интеграция в e-commerce платформы;
- Веб-интерфейс для дизайнеров и редакторов;
- Расширение функционала: image-to-image поиск.

Планы по улучшению:

- Поддержка мультиязычности;
- Расширенные фильтры: по стилю, тематике, формату;
- История запросов и авторизация;
- UI-пагинация и сортировка по релевантности.

7.6 Выводы

- Реализован веб-сервис для поиска вышивок по смысловому описанию;
- Архитектура масштабируемая и независимая от внешних сервисов;
- Интерфейс интуитивно понятен и готов к использованию в продуктах.

8. Выводы и перспективы

8.1 Итоги проекта

В рамках прикладного проекта была успешно решена задача **семантического поиска изображений вышивок по свободному текстовому описанию на русском языке**. Это позволило существенно упростить пользовательский сценарий и заменить фильтры ручного поиска на интуитивный текстовый ввод.

Основные достижения:

- Собран **уникальный датасет** из 21 126 пар «изображение — описание»;
- Сформированы **24 ключевых запроса** и вручную отобраны релевантные изображения;
- Реализовано **дообучение ruCLIP** с использованием контрастивной функции потерь;
- Достигнута высокая точность: **Precision@5 = 0.833**, CosineSim = 0.230;
- Разработан **веб-сервис** с полнофункциональным интерфейсом на Flask + Jinja;
- Проведена **визуальная валидация** по тематическим запросам;
- Обеспечена **воспроизводимость среды** (через кэш, pkl/csv форматы и инструкции запуска).

8.2 Соответствие целям и критериям

Таблица 1: Проверка выполнения ключевых целей проекта

Цель / критерий	**Статус**	**Комментарий**
Семантический поиск по тексту	Реализовано	ruCLIP finetuned + FAISS
Поддержка русского языка и стилистики	Да	Высокая точность
Качественный датасет	Есть	Ручная разметка + фильтрация
Дообучение модели	Выполнено	Только проекционные головы
Интерфейс и демонстрация	Есть	Веб-интерфейс с поисковой строкой
Метрики и визуальные примеры	Представлены	См. главу 6
Reproducibility / архитектура	Поддерживается	Сборка без Docker, данные кэшируются
Масштабируемость	Возможна	Простая интеграция и расширяемость

8.3 Ограничения и потенциальные улучшения

Выявленные ограничения:

- Неоднозначные или абстрактные запросы («уют», «радость») интерпретируются непоследовательно;
- Нет фильтрации по категории, стилю, цвету;
- Возможна путаница с омонимами (напр., «рак»: животное или болезнь);
- Ограниченность области применения — только машинная вышивка;
- Веб-интерфейс не предоставляет расширенного функционала (например, сортировки).

Предлагаемые улучшения:

- Расширение текстового корпуса (отзывы, описания, пользовательские данные);
- Фильтрация результатов по атрибутам (категория, стиль, формат);
- Поддержка image-to-image поиска;
- Интеграция с внешними платформами (CRM, магазины);
- Использование reranker-модуля (BERT или аналог).

8.4 Вектор развития

- **B2B-интеграция** в CRM и маркетплейсы;
- **Расширение запроса** на стиль, формат, назначение («в стиле барокко», «детский», «для подушки»);
- **Генерация дизайнов по описанию** (через diffusion или GAN-модели);
- **Новые домены**: логотипы, мерч, графика, стикеры, упаковка.

8.5 Финальные выводы

- Мультимодальные модели позволяют запускать поиск по смыслу — без аннотаций и ручного тега;
- Даже ограниченный датасет даёт хороший результат при правильной архитектуре;
- Разработанная система готова к **тестированию в реальных продуктах** и расширению под новые задачи.

Итог: мультимодальный семантический поиск — это не просто удобство, а **инструмент повышения пользовательской ценности и роста бизнеса** в нишевых визуальных категориях.