МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра информатики и программирования

РАЗРАБОТКА СИСТЕМЫ ПОДБОРА ЗАДАЧ ДЛЯ ТРЕНИРОВКИ НАВЫКОВ АЛГОРИТМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы
направления 02.03.03 — Математическое обеспечение и администрирование
информационных систем
профиль «Технологии программирования»
факультета КНиИТ
Агарева Андрея Андреевича

Научный руководитель
ст. преп. кафедры ИиП _____ А. А. Казачкова

Заведующий кафедрой
доцент, к. ф.-м. н. _____ М. В. Огнева

Саратов 2025

ВВЕДЕНИЕ

Актуальность темы. В настоящее время широко распространены сервисы и платформы для решения задач по информатике и алгоритмам. Они нацелены на широкую аудиторию, включающую как работающих профессионалов, желающих развить свои навыки в области классических задач информатики, так и начинающих программистов и участников олимпиад. Однако многие из сервисов предоставляют ограниченное количество функций, позволяющих персонализировать обучение. Большинство платформ ограничиваются составлением небольших планов обучения. При всём этом такие методы машинного обучения, как рекомендательные системы, эффективно используются для персонализации обучения [1].

В то же время алгоритмические собеседования стали стандартом испытаний при приёме на работу в IT-компании [2]. Таким образом, разнообразное сообщество программистов нуждается в индивидуальном подходе к обучению, в зависимости от навыков каждого. Это подтверждает актуальность задачи построения систем, помогающих программистам ориентироваться среди тысяч задач.

Цель бакалаврской работы — реализация системы подбора задач для тренировки навыков алгоритмического программирования. Поставленная цель определила **следующие задачи**:

1. Рассмотреть существующие системы подбора задач по информатике.
2. Изучить подходы к реализации необходимого функционала системы: изучить алгоритмы рекомендаций, алгоритмы поиска похожих задач, алгоритмы поиска похожего исходного кода.
3. Реализовать серверную часть системы, включающую: алгоритм рекомендации, алгоритм поиска задач, наиболее похожих на заданную, алгоритм поиска примеров кода пользователя, похожих на заданный.
4. Реализовать веб-приложение, предоставляющее графический интерфейс для работы с сервисом.

Методологические основы построения систем, призванных персонализировать и ускорять обучение описаны в работах F. L. da Silva, B. K. Slodkowskiand, K. K. A. da Silva, S. C. Cazella, N. Manouselis, H. Drachslerand, V. Katrien, E. Duval.

Теоретическая значимость бакалаврской работы заключается в про-

ведённом сравнении алгоритмов рекомендательных систем в контексте задачи персонализации обучения решению задач по информатике.

Практическая значимость бакалаврской работы заключается в реализации сервиса для персонализации обучения решению задач по информатике.

Структура и объём работы. Бакалаврская работа состоит из введения, 7 разделов, заключения, списка использованных источников и 3 приложений. Общий объем работы — 69 страниц, из них 41 страниц — основное содержание, включая 10 рисунков и 5 таблиц, список использованных источников информации — 25 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Обзор решений для тренировки навыков алгоритмического программирования» посвящён обзору платформ для решения задач по информатике, сервисов для подбора задач для тренировки навыков программирования.

Среди популярных платформ выделяется Leetcode, предлагающий обширную библиотеку из около 2500 задач, охватывающих базовые алгоритмы (сортировка, поиск) и структуры данных (деревья, графы, хеш-таблицы). Задачи разделены по уровням сложности и темам, некоторые имитируют реальные собеседования. Система анализирует решения по времени выполнения и потреблению памяти, сравнивая результаты с другими пользователями, визуализирует статистику прогресса (количество решённых задач, точность, среднее время). Ключевое преимущество — приближённость к формату технических интервью. Однако дополнительные функции (например, задачи по компаниям) доступны только по подписке, а помощь в обучении ограничена ручными планами.

Платформа Codeforces, основанная в 2010 году, фокусируется на соревновательном программировании и подготовке к олимпиадам. Она предлагает архив из более чем 10 000 задач, фильтрацию по темам (динамическое программирование, теория графов) и сложности, регулярные раунды с детальными разборами. Система рейтинга ранжирует пользователей, а открытый API позволяет собирать статистику. Недостаток — отсутствие встроенных механизмов персонализации, что требует использования внешних ресурсов.

Сервисы, такие как CF Problem Recommender, Codeforces Recommendation Engine, используют данные Codeforces для рекомендаций задач на основе истории пользователя. Однако их алгоритмы неизвестны, рекомендации не обосновываются, а функционал ограничен. Анализ показал, что Codeforces предпочтительнее для интеграции из-за обширного архива и открытого API.

Во втором разделе «Основные принципы рекомендательных алгоритмов» рассматриваются основные подходы к построению рекомендательных систем.

Обозначим через U множество всех пользователей, а через I — множество предметов (товаров, видео, статей и т.д.). Каждый пользователь $u \in U$ взаимодействует с подмножеством объектов $I_u \subset I$, оставляя явные или неявные оценки $R_u = \{r_{ui} | i \in I_u\}$, где r_{ui} отражает степень интереса к предмету

i.

Цель рекомендательной системы — спрогнозировать значения r_{ui} для объектов $i \in I \setminus I_u$ (т.е. тех, с которыми пользователь ещё не взаимодействовал) и предложить ему наиболее релевантные предметы. Иными словами, алгоритм должен предугадать, какие предметы вызовут максимальный интерес, основываясь на исторических данных.

Контентная фильтрация анализирует характеристики объектов и пользователей. Например, если пользователь часто решает задачи на графы, система будет рекомендовать задачи этой тематики. Используются явные признаки (тема, сложность) и косвенные данные (история взаимодействий).

Коллаборативная фильтрация основана на выявлении паттернов в поведении групп пользователей. Проблемы метода включают «холодный старт» и разреженность данных в крупномасштабных системах [3].

Гибридные системы комбинируют коллаборативные и контентные методы.

Третий раздел «Алгоритмы рекомендательных систем» рассматривает некоторые алгоритмы для построения рекомендаций. Факторизационные машины (factorization machines, FM) способны моделировать взаимодействия между разнородными признаками, такими как идентификаторы пользователей, атрибуты объектов и контекстные данные. В основе подхода лежит идея представления каждого признака в виде латентного вектора, который кодирует скрытые паттерны взаимодействий. Например, при прогнозировании оценки фильма пользователем, ФМ учитывает не только прямую связь между пользователем и фильмом, но и косвенные факторы: жанр, режиссёра, время просмотра или устройство.

Нейросетевая архитектура DSSM (Deep Semantic Similarity Model) применяется в рекомендательных системах для оценки семантической близости между пользователями и объектами на основе их векторных представлений. Модель строится по принципу «двух башен» (two-tower architecture), где одна нейросеть кодирует информацию о пользователе (например, историю взаимодействий, демографические данные), а вторая — атрибуты объекта (метаданные, текстовые описания, эмбеддинги). Полученные эмбеддинги сравниваются через косинусное расстояние [4].

Матричные разложения занимают важное место среди методов построения рекомендательных систем, обеспечивая выявление скрытых взаимосвязей

между пользователями и объектами. Основная идея заключается в разложении матрицы взаимодействий, где строки соответствуют пользователям, столбцы — товарам, фильмам или иным сущностям, а значения отражают явные или неявные предпочтения (оценки, просмотры, покупки). Например, в сервисах видеоконтента такая матрица может содержать информацию о времени, проведённом пользователем за просмотром конкретного фильма [5].

Четвёртый раздел «Оценка качества рекомендательных систем» посвящён некоторым метрикам качества, применяемым для оценки рекомендательных систем.

- Mean Average Precision (MAP) — средняя точность на позициях релевантных задач.
- Normalized Discounted Cumulative Gain (NDCG) учитывает порядок и релевантность рекомендаций с дисконтированием по позициям.
- Recall — доля релевантных задач среди рекомендованных.

Пятый раздел «Применение энкодеров к задаче поиска похожего исходного кода» проводит сравнение современных моделей-энкодеров в контексте задачи поиска похожего кода.

Модель CodeRankEmbed (137 миллионов параметров) показала наилучшие результаты превзойдя аналоги CodeSage, Arctic-Embed. Обучение основано на контрастивном подходе: минимизация расстояния между эмбеддингами логически эквивалентного кода и максимизация для семантически различного.

Шестой раздел «Сравнение баз данных» проводит сравнение популярных баз данных.

В качестве базы данных был выбран PostgreSQL. Отличительные функции, из-за которых PostgreSQL наиболее подходящая для реализуемого сервиса:

- Используется клиент-серверная модель.
- Есть возможность работы с векторами чисел. Например, можно хранить вектора вещественных чисел, выполнять поиск векторов, ближайших к заданному.
- Есть возможность использования быстрых неточных индексов в задаче поиска векторов, ближайших к заданному.

Седьмой раздел «Построение сервиса для персонализированного подбора задач по информатике» посвящён реализации сервиса для персонализированного подбора задач по информатике.

Система реализована на Python с использованием Django для серверной части и Pyodide для веб-интерфейса. Компоненты:

— Веб-приложение: интерфейс на фреймворке puepy.
— Сервер: поддержка аккаунтов, API для рекомендаций, поиска задач, кода.
— База данных: PostgreSQL с поддержкой векторных операций (HNSW-индексы для косинусного расстояния).

Начальное обучение рекомендательного алгоритма проведено на наборе данных «Codeforces Users Submissions Dataset» из 17 млн строк, включающего 14 тыс. пользователей, 10 тысяч задач. Каждая строка — попытка пользователя решить задачу. После очистки (удаление пропусков, дубликатов) осталось 6 млн строк, 14 тыс. пользователей, 10 тысяч задач.

Пусть IR — рейтинг пользователя, PR — рейтинг задачи, UR — рейтинг пользователя момент попытки, t — момент времени попытки, σ — $const$. Путём перебора была найдена формула:

$$IR = \exp\left(\frac{-(PR - UR)^2}{2\sigma^2}\right) \cdot \frac{1}{1 + \exp(-t)} \tag{1}$$

Эта формула была использована для расчёта неявного рейтинга.

Сравнение моделей показало преимущество метода чередующихся квадратов для неявного рейтинга с параметрами $k = 512$, $\alpha = 2^{10}$, $\lambda = 2^{15}$, с количеством итераций 8.

Реализован поиск похожих задач. Задачи характеризуются вектором признаков: рейтинг, количество решивших, двоичные столбцы соответствий задачи каким-то из 36 тем. Поиск реализован методом k-ближайших соседей (kNN) по нормализованным данным.

Начиная работу с сервисом, пользователь привязывает аккаунт платформы Codeforces, что даёт возможность сервису обновлять историю решённых задач, рейтинг и список решённых задач. Для «холодных» пользователей, не имеющих решённых задач, рекомендуются популярные задачи.

Реализован интерфейс сервиса:

— Страница статистики: визуализация прогресса (график решённых задач по дням, распределение по темам).
— Рекомендации: таблица задач, график, характеризующий активность решения пользователем рекомендованных задач.

— Поиск похожих задач: поиск по заданной задаче, вывод тем задач и рейтинга.
— Репозиторий кода: загрузка кода, поиск семантически близких фрагментов.

ЗАКЛЮЧЕНИЕ

В ходе проведённой работы были рассмотрены существующие платформы для тренировки навыков алгоритмического программирования и навыков решения задач по информатике. Рассмотрена теория построения рекомендательных систем, теория компактного представления программного кода в виде векторов. Найден набор данных, необходимый для обучения рекомендательного алгоритма. Выполнен разведочный анализ данных, выбран, обоснован и обучен рекомендательный алгоритм для рекомендации задач с платформы Codeforces. Реализован сервис из нескольких компонентов: рекомендательной системы, предлагающей пользователю ещё не решённые задачи, алгоритма поиска похожих задач, функционала хранения пользовательского кода с возможностью поиска похожих друг на друга экземпляров. Реализованный сервис способен в режиме онлайн учитывать изменения данных пользователей, обновлять рекомендательную систему. Все алгоритмы серверной части доступны через реализованный в виде веб-приложения графический интерфейс.

Реализованный сервис можно улучшить, применив для работы рекомендательной системы нейросети, объединив их с рекомендациями алгоритмов, основанных на фильтрации контента.

Первый раздел работы посвящён обзор решений для тренировки навыков решения задач по информатике: распространённых сервисов, имеющих архив задач, и сайтов с рекомендательными системами. Во втором разделе рассматриваются основные подходы к рекомендациям: контентно-ориентированная фильтрация, коллаборативная фильтрация, гибридные алгоритмы, ассоциативные правила. В третьем разделе рассмотрены распространённые эффективные рекомендательные алгоритмы. В четвёртом разделе вводятся некоторые метрики качества для рекомендательных систем. Пятый раздел содержит анализ применимости моделей-энкодеров к задаче поиска похожего кода, сравнительный анализ современных энкодеров, подходящих для задачи поиска похожего кода. Шестой раздел посвящён сравнению некоторых современных баз данных. Седьмой раздел содержит описание и обоснование необходимого функционала реализуемого сервиса. Также в седьмом разделе описываются все шаги построения сервиса для подбора задач по информатике.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 A systematic literature review on educational recommender systems for teaching and learning: research trends, limitations and opportunities / F. L. da Silva, B. K. Slodkowskiand, K. K. A. da Silva, S. C. Cazella // *Education and information technologies*. — 2023. — Т. 28, № 3. — С. 3289–3328.

2 Яндекс.Практикум. Подготовка к алгоритмическому собеседованию. [Электронный ресурс]. — URL: https://start.practicum.yandex/algorithms-interview (Дата обращения 20.05.2025).

3 Recommender systems for learning / N. Manouselis, H. Drachslerand, V. Katrien, E. Duval. — Springer Science & Business Media, 2012.

4 Unsupervised Deep Structured Semantic Models for Commonsense Reasoning / W. Shuohang, Z. Sheng, S. Yelong и др. — 2019. https://arxiv.org/abs/1904.01938.

5 *Falk, K.* Practical recommender systems / K. Falk. — Simon and Schuster, 2019.