Курсовой проект

Использование графовой информации для улучшения энкодер-подобных языковых моделей

Состав проектной группы и распределение ролей:

- **Холичева Ангелина** — разработка ноутбука и формирование архива эмбеддингов.
- **Дистлер Марина** — реализация модели и проведение серии экспериментов по обучению и валидации.
- **Конохова Екатерина** — анализ данных (EDA), визуализация, подготовка отчёта и финальной презентации.

Ссылка на GitHub: https://github.com/anghol/sna-project
Канал коммуникации: https://t.me/+TxFvELDZZww5MTZk

Москва, 2025

Аннотация

Целью данного проекта является исследование влияния объединения текстовых и графовых эмбеддингов на точность прогнозирования показателя вовлечённости (*engagement_score*) в постах Stack Overflow.

В качестве исходных данных использован публичный датасет `bigquery-public-data.stackoverflow.posts_questions`, охватывающий период с 1 января по 1 июня 2022 года. После этапов очистки и фильтрации было получено 115 800 вопросов, содержащих ключевые теги, связанные с Python и его экосистемой: `pandas`, `Django`, `NumPy`, `TensorFlow`, `PyTorch` и др. Каждый объект включает в себя следующие поля:

- `view_count` — количество просмотров,
- `score` — рейтинг (разность upvote и downvote),
- `answer_count` — число ответов,
- `comment_count` — число комментариев,
- `full_text` — текст вопроса,
- `tags_filtered` — список ключевых тегов,
- `creation_date` — дата публикации.

Для построения признаков были использованы следующие подходы:

- **Текстовые эмбеддинги:** эмбеддинги из предобученной модели `CodeBERT`.
- **Графовые эмбеддинги:**
  - Центральности: Degree, Betweenness, Closeness, Eigenvector,
  - SVD-разложения: матрица смежности и нормализованный лапласиан,
  - Графовые автоэнкодеры: GAE (на базе GCN и GAT), VGAE.

Результаты экспериментов показали, что гибридная модель, сочетающая текстовые и графовые признаки, демонстрирует более высокую точность по сравнению с базовой текстовой моделью.

Сводные результаты (по метрикам MSE, MAE, R^2):

Модель	MSE	MAE	R^2
`text_embedding`	0.0443	0.1404	0.0631
`centrality_embs`	0.0441	0.1381	0.0683
`gae_gat_embds`	0.0461	0.1484	0.0263
`svd_adj_embs`	0.0443	0.1298	0.0645
`vgae_gcn_embds`	0.0441	0.1372	0.0671
`gae_gcn_embds`	**0.0436**	0.1309	**0.0787**
`svd_lapl_embs`	0.0437	0.1332	0.0771

Таблица 1: Сравнение моделей по точности прогноза *engagement_score*.

Содержание

1. Введение

1.1. Актуальность задачи прогнозирования вовлечённости

В эпоху стремительного роста пользовательского контента на платформах типа Stack Overflow и в социальных сетях прогнозирование уровня вовлечённости (лайков, просмотров, комментариев) становится ключевым инструментом для оптимизации рекомендаций и повышения качества взаимодействия.

Традиционные подходы, основанные исключительно на анализе текста, нередко упускают структурные связи между объектами (пользователями, тегами, сообществами). Объединение текстовых эмбеддингов с графовыми признаками позволяет более полно учесть контекст распространения информации, выявить скрытые паттерны межтегов и сообществ, а значит — повысить точность прогнозов.

1.2. Исследовательские вопросы

1. **Какие графовые эмбеддинги дают наилучший результат?**
   Сравниваются классические центральности, SVD-разложение, графовые автоэнкодеры на базе GCN и GAT, а также подходы с вариационным автоэнкодером (VGAE) для выявления оптимального метода представления сетевой структуры.

2. **Как соотношение текстовых и графовых признаков меняется при разных масштабах и плотности сети?**
   Исследование проводилось на подвыборках с различным числом узлов и степенью внутренней связанности, чтобы оценить устойчивость и вклад каждого типа признаков в зависимости от размера и плотности графа.

2. Описание и предобработка данных

2.1. Источник датасета

Для построения ко-тегового графа и последующего моделирования была использована выборка из публичного набора `BigQuery`: `bigquery-public-data.stackoverflow.posts_questions`. Отфильтрованные данные содержат **119 820 записей**.

Отбор вопросов проводился по следующим критериям:

- **Дата:** дата создания вопрроса попадает в период с 1 января по 1 июня 2022 года.
- **Тип поста:** `post_type_id = 1` (только вопросы).
- **Ключевые теги:** наличие хотя бы одного из тегов: `python`, `pandas`, `django`, `numpy`, `tensorflow`, `pytorch` и др.
- **Язык текста:** английский (определяется через библиотеку `langdetect`).

2.2. Очистка и токенизация текста

Этап предобработки текста включал в себя:

- **Удаление HTML и нормализация регистра:** из поля `full_text` удаляются все HTML-теги, однако код окружается специальными символами: `[CODE_START]`, `[CODE_END]`, `[INLINE_CODE_START]`, `[INLINE_CODE_END]`.
- **Языковая фильтрация:** с помощью автоматического определения языка удалено **2 390 нерелевантных записей** ($\sim$2 %), что оставило **117 430** валидных вопросов.
- **Токенизация:** текст разбивается на токены с помощью токенизатора `CodeBERT`.

2.3. Построение графа

До построения графа, теги были отфильтрованы по частоте: все теги, встречающиеся меньше 50 раз, были удалены. Кроме того, если после данной операции у поста не оставалось тегов, то он также был удален.

- **Узлы:** каждый ключевой тег рассматривается как отдельная вершина.
- **Рёбра:** между двумя тегами проводится неориентированное ребро, вес которого соответствует числу совместных появлений тегов в одном вопросе.

В результате построен **взвешенный граф**:

- **11 580 узлов**, около **226 000 рёбер**.
- Граф содержит **гигантскую компоненту связности** (∼97 % всех узлов).
- Распределение степеней узлов приближено к степенному закону с показателем $\alpha \approx 2.4$.

2.4. Feature-engineering

Текстовые признаки: В начало каждого текста поста был добавлен список тегов.

Графовые признаки:

- Центральности: `Degree`, `Betweenness`, `Closeness`, `Eigenvector`.
- Сохраняются как векторы размерности 4: `centrality_embs`.

Таким образом, мы сформировали **полный набор признаков**, объединяющий текстовую и структурную (графовую) информацию, пригодный для дальнейшего обучения моделей.

3. Exploratory Data Analysis

3.1. Статистика просмотров, лайков и корреляции

На первом этапе EDA проанализированы распределения ключевых метрик вовлечённости: `view_count`, `score`, `answer_count`, `comment_count`.

- Все распределения оказались **сильно скошенными вправо**, что потребовало применения логарифмического преобразования: $\log(x+1)$.
- После трансформации показатели приобрели форму, близкую к нормальному распределению, что делает обоснованным использование моделей с MSE-ориентированной целевой функцией.

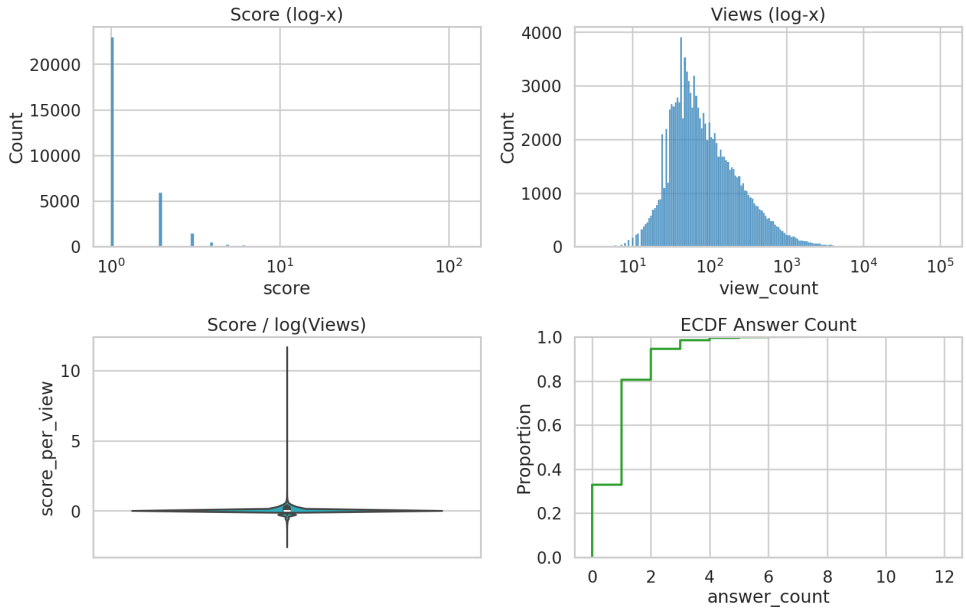

Рис. 1: Распределения `score` и `view_count` в лог-шкале, а также ECDF для `answer_count`.

Далее построена корреляционная матрица между основными метриками вовлечённости (`score`, `view_count`, `answer_count`, `comment_count`) и дополнительным признаком `score_per_view`.

- Между `score` и `score_per_view` наблюдается очень сильная положительная корреляция ($r = 0.93$).
- Между `score` и `view_count` — умеренная положительная корреляция ($r = 0.44$).
- `answer_count` демонстрирует слабую положительную корреляцию с `view_count` и `score` (около $r = 0.10 \ldots 0.14$).

- `comment_count` почти не связана ни с одним из показателей (минимальная корреляция $r \approx -0.04$ с `score_per_view`).

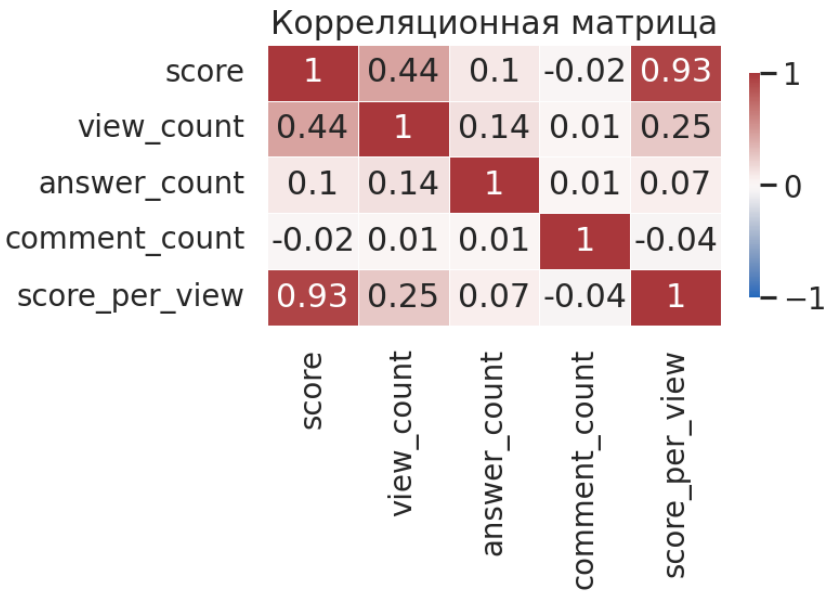

Рис. 2: Корреляционная матрица основных метрик вовлечённости и производных признаков (`score_per_view`).

Вывод Корреляционная матрица показывает, что:

- Умеренная положительная связь между `view_count` и `score` ($r = 0.44$) свидетельствует о том, что более просматриваемые вопросы в среднем получают больше голосов.
- Корреляции `comment_count` с другими метриками практически отсутствуют ($r \approx 0$), а число ответов (`answer_count`) слабо связано и с просмотрами, и с рейтингом ($r \approx 0.10 \ldots 0.14$).
- Показатель `score_per_view` очень сильно коррелирует с `score` ($r = 0.93$), что делает его информативным мерилом эффективности публикации.

3.2. Временные паттерны

Из признака `creation_date` были извлечены:

- **Час публикации**
- **День недели**
- **Месяц публикации**

Анализ временных шаблонов показал:

- **По часам суток:** максимальный средний `score_per_view` наблюдается около **9:00 UTC**, минимальный — около **5:00 UTC**.
- **По дням недели:** наибольшее число вопросов и высокий средний `engagement_score` приходится на **вторник и среду**, наименьшее — на **субботу и воскресенье**.

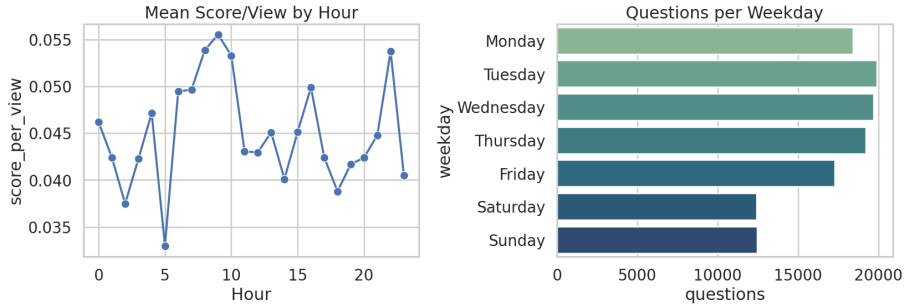

Рис. 3: Распределение количества публикаций и среднего `score_per_view` по часам суток и по дням недели.

- **По месяцам:** пик числа вопросов приходится на **март**, после чего наблюдается снижение в **апреле** и **мае**.

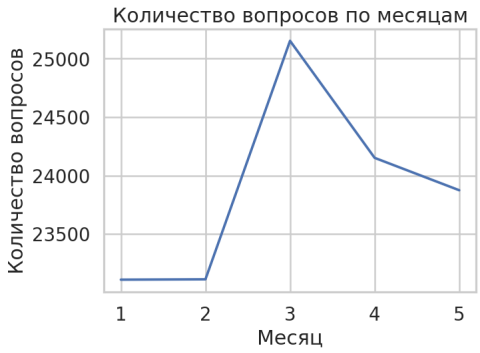

Рис. 4: Распределение количества публикаций по месяцам.

3.3. Анализ и кластеризация тегов

Изучение семантики тегов:

- **Pareto-диаграмма:** 20% самых частых тегов обеспечивают ∼80% всех упоминаний.

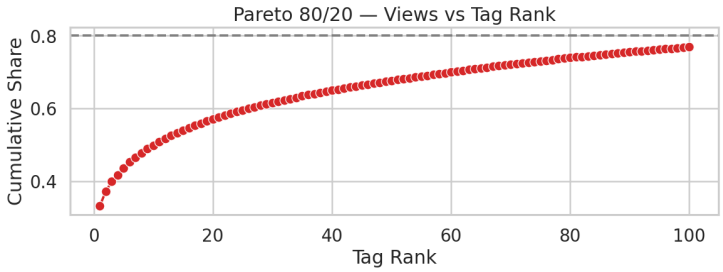

Рис. 5: Pareto-диаграмма: топ-20 тегов vs накопленная доля упоминаний.

- **Облако слов (WordCloud):** наглядно выделяет популярные и нишевые технологии.

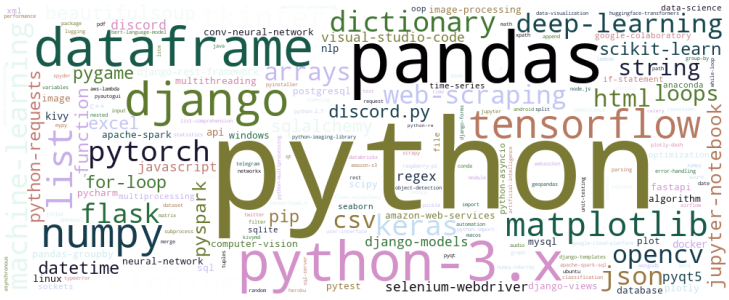

Рис. 6: WordCloud

- **Тепловая карта co-occurrence:** выявляет частые пары (например, `python` + `pandas`, `tensorflow` + `keras`).

Кластеризация тегов:

- **Алгоритм Louvain** применён к графу совместного использования тегов; число кластеров определяется автоматически.
- **UMAP-проекция** (2-D) показала чёткое тематическое разделение на шесть сообществ: *«Base Python»*, *«Data & APIs»*, *«Web-frameworks / DevOps»*, *«Mobile / Bots»*, *«Scientific Computing»*, *«Machine Learning»*.
- Средний силуэтный коэффициент для тестового K-Means-варианта составил ≈ 0.38, что подтверждает внутрекластерную согласованность.

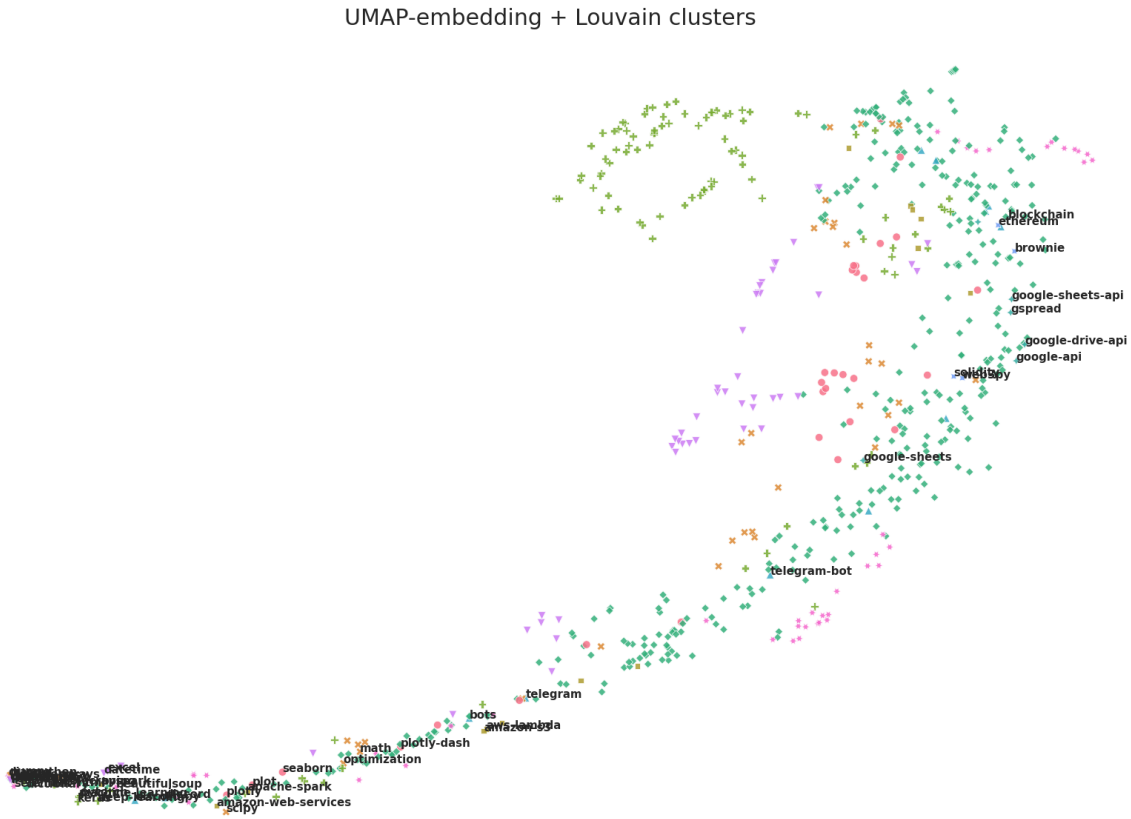

Рис. 7: UMAP-проекция

3.4. Интерактивная визуализация

Для удобной навигации по данным создан интерактивный **дашборд на Plotly**, включающий:

- **Облако тегов** с подсветкой и дополнительной статистикой при наведении.
- **UMAP-scatter plot** с динамической фильтрацией и возможностью исследовать отдельные кластеры.
- **Гистограммы временных признаков** с настройкой диапазонов и учётом часового пояса.

Такое решение делает интерфейс EDA **интерактивным и наглядным**, облегчая принятие решений при построении моделей и выборе признаков.

4. Методология эмбеддингов

4.1. Графовые эмбеддинги

Для представления структурных свойств ко-тегового графа были реализованы и протестированы следующие подходы:

Классические меры центральности

Для каждой вершины (тега) были вычислены четыре базовые меры:

- **Degree centrality** — количество инцидентных рёбер.
- **Betweenness centrality** — доля кратчайших путей, проходящих через вершину.
- **Closeness centrality** — обратная сумма расстояний до всех других узлов.
- **Eigenvector centrality** — влияние вершины с учётом значимости её соседей.

Результаты объединялись в единый вектор размерности 4: `centrality_embs`.

SVD-разложение

На основе:

- Матрицы смежности (A)
- Нормализованного лапласиана (L_{norm})

были построены эмбеддинги размерности 64:

- `svd_adj_embs` — для A
- `svd_lapl_embs` — для L_{norm}

Использовалась проекция узлов в пространство ведущих сингулярных векторов, что позволило учитывать **глобальную структуру графа**.

Графовые автоэнкодеры (GAE и VGAE)

- **GAE (GCN)**:
  - Кодировщик — двухслойный GCN
  - Декодировщик — скалярное произведение эмбеддингов
  - Размер скрытого слоя — 64

  - Результат: `gae_gcn_embds`

- **VGAE (GCN)**:
  - Архитектура аналогична GAE
  - Дополнительно моделируются среднее и лог-дисперсия латентных переменных
  - Результат: `vgae_gcn_embds`, размерность 128

- **GAE (GAT)**:
  - Механизм внимания (Graph Attention Networks) вместо GCN
  - Позволяет взвешивать вклад соседей адаптивно
  - Скрытый размер — 64
  - Результат: `gae_gat_embds`

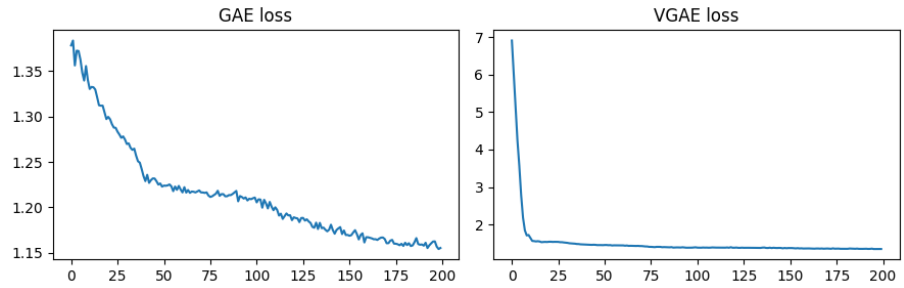

Рис. 8: Кривые обучения графовых автоэнкодеров: GAE (слева) и VGAE (справа).

Все графовые эмбеддинги обучались на задаче реконструкции графа — минимизации функции потерь по восстановлению рёбер (*link prediction*). Полученные векторы сохранялись и использовались как **структурные признаки** при прогнозировании метрик вовлечённости.

4.2. Текстовые эмбеддинги

Для представления семантики текста вопросов применялся следующий подход: **CodeBERT**: извлечение CLS-эмбеддинга (размерность 768) из последнего слоя модели. Эта модель захватывают **глубинные лингвистические зависимости** и предоставляют богатое контекстное представление, далее используемое в гибридной архитектуре. Также стоит заметить, что часть слоев **CodeBERT** не была заморожена во время обучения на нашу задачу. А именно, слой эмбеддингов и последний слой.

4.3. Гибридная модель

Конкатенация эмбеддингов

- Для каждого примера объединяются:
  - **Текстовый эмбеддинг** — размерность 768
  - **Графовый эмбеддинг** — размерность от 4 до 128
- Итоговый вектор подаётся в нейронную сеть.

Fusion-слой

- Два полносвязных слоя: 512 и 256 нейронов.
- Активация: ReLU, дропаут: 0.3.
- Назначение — адаптивное взвешивание текстовых и структурных признаков.

Выходной слой

- **Регрессия**: один нейрон без активации — предсказание.

Обучение

- Лосс: **MSE** (регрессия).
- **Оптимизатор**: `Adam`, $\mathrm{lr} = 1 \times 10^{-5}$,
- Совместное обучение (*end-to-end*) всей архитектуры.

4.4. Метрики и протокол валидации

Разбиение данных

- **Train**: 80%,
- **Test**: 20%.

Метрики качества

Для регрессии:

- MSE (среднеквадратичная ошибка)
- MAE (средняя абсолютная ошибка)
- R^2 (коэффициент детерминации)

Дополнительно

- Раннее останавливание: мониторинг `MSE` на валидации с `patience = 3`.
- Повторяемость: фиксированные `random seeds` (NumPy, PyTorch), логирование версий библиотек и моделей.

5. Эксперименты и подбор гиперпараметров

5.1. Целевая переменная

Для предсказания мы выбрали $\frac{score}{\log view\_count}$. Такким образом мы избавились от зависимости от часстоты встречаемости тегов и сосредоточились на качестве поста.

5.2. Baseline: только текст

В качестве опорной модели использовалась `CodeBERT`; таким образом модель использует исключительно текстовый CLS-вектор **CodeBERT** (768), построенный на архитектуре RoBERTa.

- **Заморозка слоёв**: все параметры CodeBERT кроме эмбеддинг слоя и последнего слоя фиксированы, обучается также линейная “голова”.
- **Оптимизатор**: `Adam`, $\mathrm{lr} = 1 \times 10^{-5}$,
- **Loss**: MSE.
- **Batch size**: 8.
- **Эпохи**: 5 (подбор по кривым валидации).

Результаты (валидация): MSE_0, MAE_0, R_0^2 (см. Раздел 10).

5.3. Сравнение графовых эмбеддингов

Для каждого типа графовых эмбеддингов была проведена отдельная тренировка той же архитектуры с конкатенацией графовых векторов к CLS-вектору Roberta.

Общие настройки

- **Оптимизатор**: `Adam`, $\mathrm{lr} = 1 \times 10^{-5}$,
- **Функция потерь**: MSE
- **Batch size**: 8
- **Эпохи**: 5

Эмбеддинги

1. Centrality (4-мерный)
2. SVD по матрице смежности (64-мерный)
3. SVD по нормализованному лапласиану (64-мерный)
4. GAE на GCN (64-мерный)
5. VGAE на GCN (128-мерный)
6. GAE на GAT (64-мерный)

Метрики: MSE, MAE, R^2 на валидационном наборе.

6. Настройка и воспроизводимость обучения

6.1. Среда и оборудование

- **Язык и библиотеки**: Python 3.8+, Pandas, NumPy, scikit-learn, PyTorch, PyTorch Geometric, SciPy, Transformers, NetworkX, UMAP, langdetect, TensorBoard, Plotly.
- **Операционная система**: *[Ubuntu 20.04]*
- **CPU**: *[Intel Xeon Gold 6226R @ 2.90GHz]*
- **GPU**: *[NVIDIA A100 80GB]*

6.2. Скрипты обучения

Все эксперименты запускались через `train_model()` (см. `train_models.ipynb`), где реализована логика:

- **Заморозка слоёв**: Все слои Roberta, кроме слоя эмбеддингов и выходного линейного слоя.
- **Оптимизатор**: `Adam`, $\mathrm{lr} = 1 \times 10^{-5}$,
- **Функция потерь**: MSELoss.
- **Размер батча**: 8.
- **Число эпох**: 5
- **Логирование**: TensorBoard с логированием loss и сохранением чекпоинта лучшей модели по минимальному значению валидационного MSE.

7. Результаты и обсуждение

Модель	**MSE**	**MAE**	R^2
Text-only (baseline)	0.0443	0.1404	0.0631
Centrality + Text	0.0441	0.1381	0.0683
SVD-adj + Text	0.0443	0.1298	0.0645
SVD-lapl + Text	0.0437	0.1332	0.0771
VGAE-GCN + Text	0.0441	0.1372	0.0671
GAE-GCN + Text	**0.0436**	**0.1309**	**0.0787**
GAE-GAT + Text	0.0461	0.1484	0.0263

Таблица 2: Сравнение моделей по валидационному MSE, MAE и R^2

Даже простейшие графовые признаки (центральности) дают улучшение относительно baseline. Наибольший прирост R^2 достигается при использовании **GAE-GCN** (+24.7% относительно текстовой модели), почти наравне с **SVD-laplacian** ($R^2 = 0.0771$). Механизм внимания GAT, напротив, приводит к переобучению и снижению всех метрик.

8. Заключение и дальнейшие шаги

Основные выводы

- **Графовые эмбеддинги важны**: добавление структурных признаков стабильно улучшает качество относительно чисто текстовой модели.
- **Лучшее качество — у GAE-GCN**: автоэнкодер на базе GCN дал максимальный $R^2 = 0.0787$; близкий результат показал SVD по нормализованному лапласиану ($R^2 = 0.0771$).
- **Сложность имеет предел**: внимание GAT переусложнило модель, что привело к переобучению и ухудшению метрик.

Будущие направления

- **Динамические графы**: Инкрементальное обновление эмбеддингов при поступлении новых данных.
- **Обобщение**: Проверка на других сегментах Stack Overflow и смежных платформах (например, Reddit).
- **Продакшен**: Интеграция в систему онлайн-рекомендаций для платформ вопросов и ответов.

9. References

Список литературы

[1] Devlin J., Chang M.-W., Lee K., Toutanova K.
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
arXiv:1810.04805, 2019.
https://arxiv.org/abs/1810.04805

[2] Kipf T.N., Welling M.
Semi-Supervised Classification with Graph Convolutional Networks.
arXiv:1609.02907, 2017.
https://arxiv.org/abs/1609.02907

[3] Kipf T.N., Welling M.
Variational Graph Auto-Encoders.
arXiv:1611.07308, 2016.
https://arxiv.org/abs/1611.07308

[4] Veličković P., Cucurull G., Casanova A., Romero A., Lio P., Bengio Y.
Graph Attention Networks.
arXiv:1710.10903, 2018.
https://arxiv.org/abs/1710.10903

[5] Zhang Z., Zheng Y., Cui P., Wang L.
Graph-BERT: Only Attention is Needed for Learning Graph Representations.
arXiv:2001.05140, 2020.
https://arxiv.org/abs/2001.05140

[6] McInnes L., Healy J., Melville J.
UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.
arXiv:1802.03426, 2018.
https://arxiv.org/abs/1802.03426

[7] Blondel V.D., Guillaume J.-L., Lambiotte R., Lefebvre E.
Fast Unfolding of Communities in Large Networks.
J. Stat. Mech., 2008.
https://arxiv.org/abs/0803.0476

[8] Google BigQuery Public Dataset: Stack Overflow posts_questions.
Accessed: 26 June 2025.
https://cloud.google.com/bigquery/public-data

10. Приложения

- `data_preprocessing.ipynb` — обработка датасета и создание графа.
- `eda.ipynb` — анализ структуры графа и визуализация.
- `embeddings.ipynb` — обучение графовых эмбеддингов.
- `train_models.ipynb` — обучение гибридной модели и валидация.
- `requirements.txt` — список зависимостей окружения.