Пермский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Соломатин Роман Игоревич

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ПОИСКА ИСПОЛНИТЕЛЕЙ ПО ТЕХНИЧЕСКОМУ ЗАДАНИЮ ПРИКЛАДНОГО ПРОЕКТА

Курсовая работа

студента образовательной программы «Программная инженерия»
по направлению подготовки *09.03.04 Программная инженерия*

Руководитель
к.т.н., доцент кафедры Информационных технологий в бизнесе НИУ ВШЭ-Пермь

А. В. Бузмаков

Пермь, 2021 год

Аннотация

Ежедневно появляется множество тендеров, в которых можно принять учатие. Но на подготовку к нему отводится мало времени. Данная работа посвящена проблеме поиска исполнителей по текстовому описанию тендера.

В первой главе находится постановка задачи, обзор существующих решений, выбор языка программирования и системы управления базами данных.

Во второй главе представлено проектирование базы данных, информационной системы и сбор данных.

В третьей главе содержится описание процесса разработки информационной системы.

Количество страниц – 26, количество иллюстраций – 9, количество таблиц – 3.

Оглавление

Введение 4
Глава 1. Анализ предметной области 6
1.1 Формальная постановка задачи 6
1.1.1 Прецедента «Добавление тендера» 7
1.1.2 Прецедента «Просмотр тендеров» 7
1.1.3 Прецедента «Добавление исполнителя» 7
1.1.4 Прецедента «Поиск компетенций исполнителя» 7
1.1.5 Прецедента «Подбор исполнителя для тендера» 8
1.1.6 Прецедента «Редактирование информации об исполнителе» 8
1.1.7 Описание прототипа 8
1.2 Выбор языка программирования 9
1.3 Выбор СУБД 10
Глава 2. Проектирование системы 12
2.1 Проектирование базы данных 12
2.1.1 Приведение к 1НФ 15
2.1.2 Приведение к 2НФ 16
2.2 Сбор данных 17
2.3 Создание компетентностного профиля исполнителя 18
2.4 Возможные методы подбора исполнителя по ТЗ 21
Глава 3. Разработка и тестирование системы 23
3.1 Разработка интерфейсов 23
3.2 Тестирование приложения 23
Заключение **25**
Библиографический список **26**
ПРИЛОЖЕНИЕ А **27**

Введение

Каждый день выкладывается несколько десятков тендеров, сроки участия в которых очень сжаты, и потенциальным исполнителям надо быстро определиться смогут ли они выполнить тендер или нет. Для принятия решения надо ознакомиться с тендером, собрать команду профессионалов для участия в заявке и выполнения тендера, а также и подготовить заявку. Это все может занять много времени. Действительно, в «Высшей школе экономики» много людей с разными компетенциями, и надо по текстовому описанию тендера понять, кто его сможет сделать. Для этого нужно знать, в чем каждый из работников университета компетентен. Для того чтобы узнать компетенции каждого из работников, можно проанализировать тексты, в создании которых сотрудник университета принял участие.

В рамках ВШЭ сотрудники принимают участие в создании нескольких видов текстов, в том числе статьи в научные журналы и выпускные курсовые работы, которые пишут студенты под их руководством. Анализируя эти тексты можно выделить сферы интересов сотрудников. Для целей такого анализа на первом этапе были выбраны ВКР, потому что для доступа ко многим научным журналам требуется платная подписка и кроме того различные научные статьи выложены на различных сайтах, что затрудняет единообразную работу с ними. Кроме того, статьи пишутся на разных языках, что тоже затрудняет работу. А выпускные курсовые работы, напротив, находятся в свободном доступе и при этом похожи на научные статьи и содержат много текстовой информации для получения выявление компетенций сотрудников. Таким образом, можно получить сферу компетенций сотрудника, по которой в дальнейшем искать соответствие между пришедшем текстовым описанием задачи и профилем сотрудника. Получается, есть проблемы поиска исполнителей на тендер.

Таким образом, проблемой рассматриваемой в данной работе является автоматический поиск исполнителей под тендер, заданный текстовым описанием. В рамках работы будет проверяться гипотеза, можно ли автоматически на основании анализа текстов ВКР, написанных под руководством сотрудника НИУ ВШЭ, получить сферу компетенций преподавателя,по которой в дальнейшем искать соответствие с текстовым описанием тендера.

Объект исследования - процесс поиска исполнителей по текстовому описанию тендера.

Предмет исследования - автоматизация процесса поиска исполнителей по текстовому описанию тендера.

Цель работы – создать информационную систему для поиска исполнителя по текстовому описанию тендера.

Для достижения поставленной цели нужно сделать:

В первой главе постановка задачи.

Во второй главе описание проектирования системы.

В третьей главе пример работы приложения.

Работа оформлена в соответствии с правилами написания курсовых работ [1].

Глава 1. Анализ предметной области

1.1. Формальная постановка задачи

На данный момент достаточно затруднительно найти программные продукты, которые могут находить исполнителей по текстовому описанию тендера. Действительно, это нетипичная ситуация, когда есть много исполнителей с разными компетенциями, которые ограничено известны лицу, составляющему команду по заявки на тендер. Также редко в каких организациях по текстам, которые пишут исполнители можно составить представлении о их компетенциях. Сейчас это проблема решается вручную.

Через ВШЭ проходит много тендеров, задаваемых техническими заданиями. Эти заявки проходят через несколько фильтров и попадают к специальному человеку, которые принимает решение, на сколько эта заявка подходит под профиль компетенций ВШЭ. Этот человек знает компетенции многих исполнителей НИУ ВШЭ и может оценить, насколько рассматриваемый тендер может быть решён силами НИУ ВШЭ. После этого, если этот человек думает, что тот или иной исполнитель обладает требуемыми компетенциями, то ему пишется письмо. Сотрудник через некоторое время отвечает готов ли он участвовать в заявке на тендер и в его исполнении. Если удалось найти команду, то за оставшееся, обычно короткое время, нужно подготовить проект заявки, задать уточняющие вопросы организатору тендера, найти недостающих исполнителей, а также выполнить много других задач. Данный процесс занимает много времени Эта система имеет недостатки:

- Много тендеров теряется, так как человек не знает все компетенции всех исполнителей
- Сложно искать тендеры
- Тяжело масштабировать, потому что необходимо знать много про разных людей
- Тратится много времени

Иногда применятся способ массовой рассылки требующихся исполнителей для тендера. У данного подхода также есть определённые недостатки:

- Массовую рассылку не все читают
- Не даёт полную информацию о тендере

Целевая система работе система поможет автоматизировать следующие процессы:

- Определение компетенций сотрудника
- Поиск сотрудников для выполнения задания

Варианты использования реализуемой системы:

- Просмотр активных проектов
- Поиск компетенций исполнителя
- Подбор сотрудника для проекта
- Редактирование информации о исполнителя

Так как в данной работе нужно проверить гипотезу, будет разрабатываться прототип программы. Диаграмма вариантов использования конечного программного продукта показана на рисунке 1.1. Ниже рассмотрим подробнее каждый из приведённых прецедентов.

1.1.1. Прецедента «Добавление тендера»

Предоставляется текстовое описание тендера и потом для него подбираются исполнители из базы данных.

1.1.2. Прецедента «Просмотр тендеров»

Выдаётся список тендеров, на которые были отправлены заявки об участии в тендере.

1.1.3. Прецедента «Добавление исполнителя»

Добавление исполнителя в базу данных, и запись данных о нем.

1.1.4. Прецедента «Поиск компетенций исполнителя»

Создание компетенций для заданного человека. Для этого надо обработать тексты ВКР с его профиля на сайте «Высшей школы экономики», которые хранятся в формате docx, doc или pdf. 1.2

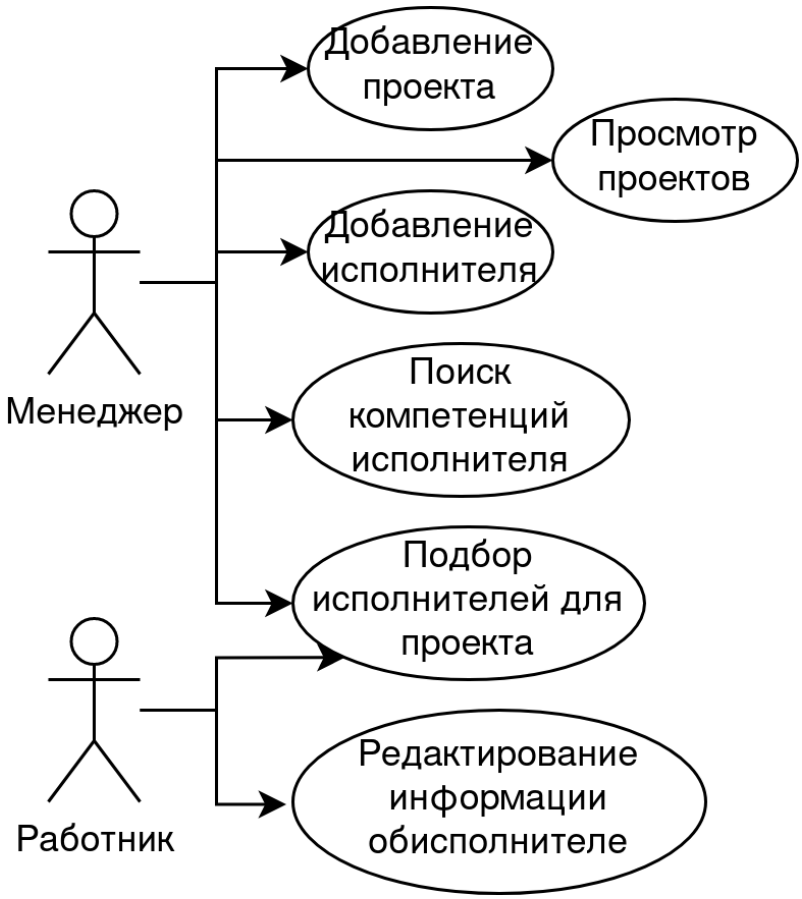

Рис. 1.1. Диаграмма вариантов использования

1.1.5. Прецедента «Подбор исполнителя для тендера»

Подбор исполнителя под текстовое описание тендера. 1.3

1.1.6. Прецедента «Редактирование информации об исполнителе»

Редактирование данных исполнителя.

1.1.7. Описание прототипа

Основной целью данной работы является проверка гипотезы об автоматическом поиске исполнителя на основании анализа текстов ВКР. Поэтому будет разрабатываться прототип, который будет включать в себя только следующие функции:

- Добавление исполнителей
- Поиск компетенций исполнителей

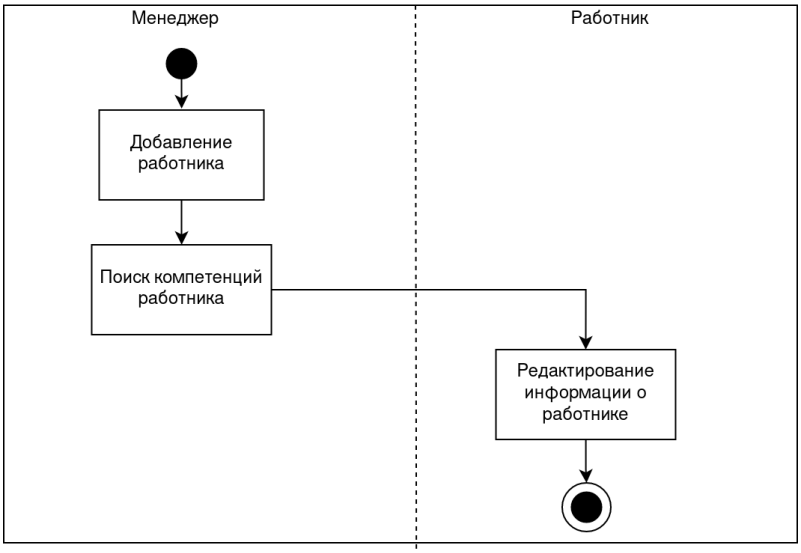

Рис. 1.2. Поиск компетенций исполнителя

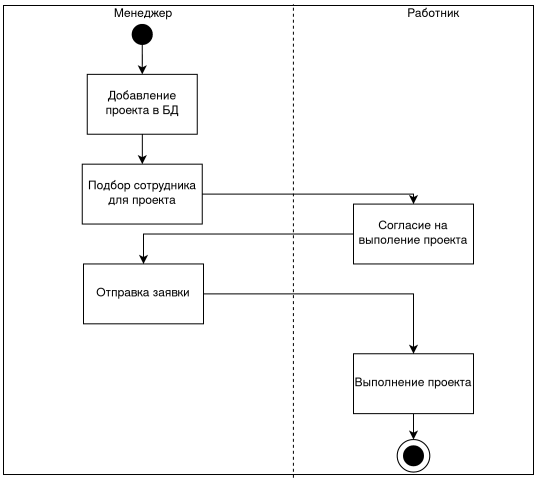

Рис. 1.3. Автоматизация бизнес-процесса "Подбор исполнителя для тендера"

1.2. Выбор языка программирования

Для прототипирования подходят многие языки программирования, вот некоторые из них: Python, C#, Java, JavaScript. Определимся с основными критериями для выбора языка:

- Библиотеки для обучения моделей. Для работы с моделями машинного обучения.
- Предобученные модели. Так как для обучения модели надо иметь большой объем данных и много вычислительных мощностей.
- Библиотеки для обработки сайта.

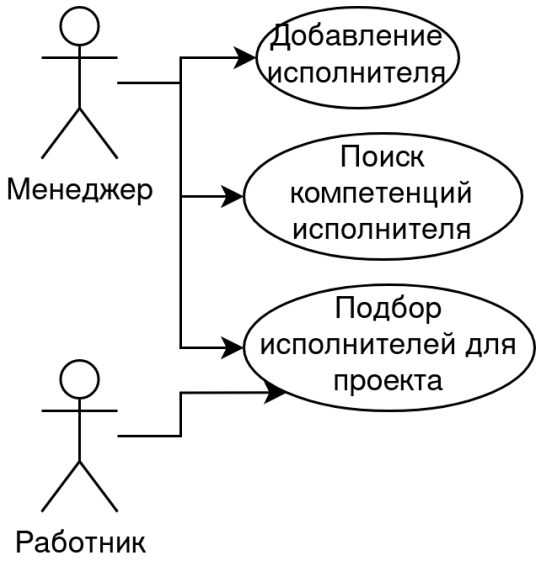

Рис. 1.4. Диаграмма вариантов использования

- Библиотека для обработки docx и pdf файлов и обработки сайтов.
- Интерактивный режим. Позволяет просмотреть данные, преобразовать их, вмешаться в исполнение кода. Выполнить в произвольном порядке.

Таблица 1.1. Таблица сравнения языков программирования

Язык программирования	Библиотека для МО	Предобученные модели	Работа с сайтом	Работа с файлами	Интерактивный режим
Python	++	++	+	+	+
C#	+	+	+	+	-
Java	+	+	+	+	-
JavaScript	+	+	+	+	?

Под эти критерии подходит Python. Он простой для написание и отладки, так как это интерпретируемый язык программирования, у него есть интерактивный режим. Также есть фреймворк PyThorch и библиотека с предобученными моделями HuggingFace Transformers, в частности RuBERT [2], а также есть библиотека Bert Extractive Summarizer [3] для удобной работы с моделей.

1.3. Выбор СУБД

Для Python существуют библиотеки для работы с любыми системами управления базами данных. Для курсовой работы был выбран SQLite потому

что его легко настраивать, нет необходимости устанавливать ничего дополнительно, что достаточно для прототипирования с небольшим количеством данных: 300 сотрудников и 1700 выпускных квалификационных работ.

Глава 2. Проектирование системы

2.1. Проектирование базы данных

Для разработки системы необходимо хранить информации о сотрудниках и выпускных квалификационных работах, где они были руководителями была разработана база данных. Необходимо было хранить:

Код сотрудника – уникальный код сотрудника для его идентификации в базе данных

ФИО сотрудника – фамилия имя и отчество сотрудника

Ссылка на профиль сотрудника – ссылка на профиль сотрудника на сайте Высшей школы экономики, для последующей проверки в ручную при подборе на тендер

Код статуса – код статуса для последующей связи в базе данных

Код кафедры – код кафедры для последующей связи в базе данных

Компетенции – компетенции сотрудника полученные путём автоматического анализа текстов ВКР в текстовом виде

Эмбеддинги – компетенции сотрудника полученные путём автоматического анализа текстов ВКР в векторном пространстве для дальнейшего подбора исполнителя для тендера

Код статуса – уникальный код статуса для идентификации в базе данных

Наименования статуса – доцент, старший научный сотрудник и тд. Для представление информации о сотрудника.

Код кафедры – уникальный код кафедры для идентификации в базе данных

Наименование кафедры – название кафедры, где работает сотрудник. Для представление информации о сотруднике.

Код кампуса – уникальный код для идентификации в базе данных

Наименования кампуса – название кампуса. Для представление информации о сотруднике.

Код ВКР – уникальный код выпускной квалификационной работы для идентификации в базе данных

Название ВКР – название выпускной квалификационной работы. Необходимо для дальнейшего соотнесения в сотрудника ручном режиме

Научный руководитель – код сотрудника для последующей связи в базе данных

Ссылка на ВКР – ссылка на выпускную квалификационную работу на сайте ВШЭ для проверки корректности сбора информации

Ссылка на полный текст ВКР – ссылка на файл для загрузки с полным текстом ВКР

ФИО студента – фамилия имя и отчество студента написавшего ВКР

Код ОП студента – код образовательной программы студента для последующей связи в базе данных

Код кампуса – код кампуса для последующей связи в базе данных

Код образовательной программы – уникальный код для идентификации в базе данных

Код факультета – код факультета для последующей связи в базе данных

Наименование ОП – название образовательной программы, где обучается студент

Код факультета – уникальный код факультета для идентификации в базе данных

Наименование факультета – название факультета

Описание данных для проектирования БД 2.1.

Таблица 2.1. Таблица атрибутов

Имя атрибута	Тип данных	Значение по умолчанию	Формат ввода	Ограничение на значения
Код сотрудника	Число	Нет	Нет	Нет
ФИО сотрудника	Строка	Нет	Нет	Нет
Ссылка на профиль сотрудника	Строка	Нет	Нет	Нет
Компетенции	Строка	Нет	Нет	Нет
Эмбеддинги	Число	Нет	Нет	Нет
Код ВКР	Число	Нет	Нет	Нет
Название ВКР	Строка	Нет	Нет	Нет
Ссылка на ВКР	Строка	Нет	Нет	Нет
Ссылка на полный текст ВКР	Строка	Нет	Нет	Нет
ФИО студента	Строка	Нет	Нет	Нет
Код ОП студента	Число	Нет	00.00.00	Нет
Код статуса	Число	Нет	Нет	Нет
Наименования статуса	Строка	Нет	Нет	Нет
Код кафедры	Число	Нет	Нет	Нет
Наименование кафедры	Строка	Нет	Нет	Нет
Код кампуса	Число	Нет	Нет	Нет
Наименования кампуса	Строка	Нет	Нет	Нет
Код образовательной программы	Число	Нет	00.00.00	Нет
Код факультета	Число	Нет	Нет	Нет
Наименование ОП	Строка	Нет	Нет	Нет
Код факультета	Число	Нет	Нет	Нет
Наименование факультета	Строка	Нет	Нет	Нет

2.1.1. Приведение к 1НФ

Данные находятся в первой нормальной форме, если атрибуты атомарны и неделимы, состоят из элементарных составляющих и отсутствуют дубликаты.

1. Код сотрудника
2. ФИО сотрудника
3. Ссылка на профиль сотрудника
4. Компетенции
5. Эмбеддинги
6. Код статуса сотрудника
7. Код кафедры сотрудника
8. Код ВКР
9. Название ВКР
10. Код сотрудника
11. Ссылка на ВКР
12. Ссылка на полный текст ВКР
13. ФИО студента
14. Код кампуса
15. Код ОП студента
16. Код статуса
17. Наименование статуса
18. Код кафедры
19. Наименование кафедры
20. Код кампуса

21. Наименования кампуса
22. Код образовательной программы
23. Код факультета
24. Наименование ОП
25. Код факультета
26. Наименование факультета

Данные атрибуты находятся в 1НФ.

2.1.2. Приведение к 2НФ

Данные находятся во второй нормальной форме, если они в первой нормальной форме и отсутствует частичная функциональная зависимость не ключевых атрибутов от ключа. В соответствии с описанными выше зависимостями можно сделать вывод, что в описанном отношении имеются следующие частичные зависимости:

1. Код сотрудника определяет:
    - ФИО сотрудника
    - Ссылка на профиль сотрудника
    - Компетенции
    - Эмбеддинги
    - Код статуса сотрудника
    - Код кафедры сотрудника
2. Код ВКР определяет:
    - Название ВКР
    - Код сотрудника
    - Ссылка на ВКР
    - Ссылка на полный текст ВКР
    - ФИО студента

- Код кампуса
- Код ОП студента

3. Код статуса сотрудника определяет:
    - Наименование статуса
4. Код факультета определяет:
    - Наименование факультета
5. Код кампуса определяет:
    - Наименование кампуса
6. Код кафедры определяет:
    - Наименование кафедры
7. Код образовательной программы определяет:
    - Название образовательной программы

2.2. Сбор данных

Для работы программы необходимо собрать данные о сотрудниках и ВКР, которые у них писались. Для этого надо было обрабатывать информация находящуюся на сайте ВШЭ. Такая обработка возможна с помощью библиотеки BeautifulSoap, который позволяет получать html любого сайта.

В первую очередь надо было получить список сотрудников и ссылки на их страницы. Для этого обрабатывался сайт со списком сотрудников, но там можно получить список сотрудников, фамилии которых начинаются на 1 букву. Что бы решить эту проблему с сайты получались ссылки на буквы. А потом с каждой такой ссылки собирался список сотрудников и ссылки на их страницы. Для этого брались все элементы страницы с атрибутом "a" и классом "link" , в которых значение атрибута "href" было "/org/persons/" или "/staff/". В базу данных записывались ФИО сотрудника и ссылку на их страницу.

Далее надо было получить список ВКР для каждого сотрудника. Для этого брались все элементы страницы с атрибутом "a" и классом "link" , в которых значение атрибута "href" было "/edu/vkr/". Итого получилось собрать информацию о 321 сотруднике и 1708 ВКР.

Потом надо было получить информацию о ВКР, но их данные ВКР, в отличие от данных сотрудников, подгружались после загрузки основой страницы. Чтобы решить эту проблему пришлось эмулировать работу браузера с помощью библиотеки Selenium и драйвера Gecko (движок браузера FireFox). Библиотека Selenium не использовалась раньше, так как для ее работы надо больше времени и ресурсов, поэтому данная библиотека использовалась только на данном этапе. Для полной загрузки страницы ставилась задержка в 2 секунды. И бралась информация атрибута "p" с классом "vkr-card__item". Также бралась информация о научном руководителе, он искался в базе данных, и потом в базу данных уже записывался Код сотрудника.

Чтобы получить тексты из файлов, они обрабатывались с помощью textract, которая позволяет получить информацию из pdf, doc и docx файлов, и помещались в базу данных.

2.3. Создание компетентностного профиля исполнителя

Для создание профиля человека рассматривались TFiDF [4] и суммаризация текста с помощью BERT [5] (Bidirectional Encoder Representations from Transformers).

Подход TFiDF заключается в том что бы для каждого слова посчитать его отношение числа вхождений некоторого слова к общему числу слов документа (TF) и частоты, с которой некоторое слово встречается в документах коллекции (DF). Частоты в рамках коллекции рассматриваются для того что бы выявить распространённые слова в данных документах и снизить их важность, например "Информационная система и наоборот найти слова которые находят важную тему, например "Машинное обучение". Данный подход часто используется для кластеризации текстов. Но данный метод плохо подходит для данной задачи, потому что он не рассматривает контекст используемых слов. Это значит у него проблемы с омонимами и синонимами. Например, в предложениях "Ключ открыл замок"и "Рыцарь штурмовал замок"слова "зАмок"и "замОк"будут считаться как одно и тоже слово, хотя по контексту это разные слова.

BERT – это нейросетевая языковая модель, которая рассматривает контекст, в котором находится слово и кодирует данное слово в своём векторном пространстве. Для этого модели подаётся на вход предложение, и каждое сло-

во разделяется на отдельную часть предложения (токенизируется). Потом в начало и конец предложения добавляются специальные токены ([CLS], [SEP]), затем все токены предложения приводятся к нормальной форме слова (лемматизируются). Потом каждому слову присваивается вектор (эмбеддинг), которым кодируется слово. Для предложений также строится свой эмбединг. Цель работы данного алгоритма – предсказывать следующее слово или понять идут предложения рядом или нет. Пример работы алгоритма 2.1.

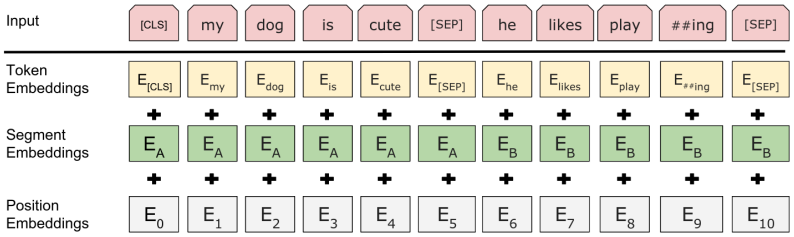

Рис. 2.1. Обработка предложения

При обучении задачи на предсказание следующего слова нейронная сеть закрывает 15% слов маской, и пытается предсказывает какие слова были замаскированы на основе контекста. Для этого нейронная сеть находит вектора слов, которые находятся рядом в предложении, и берет близкое слово в векторном пространстве, относительно соседних слов 2.2.

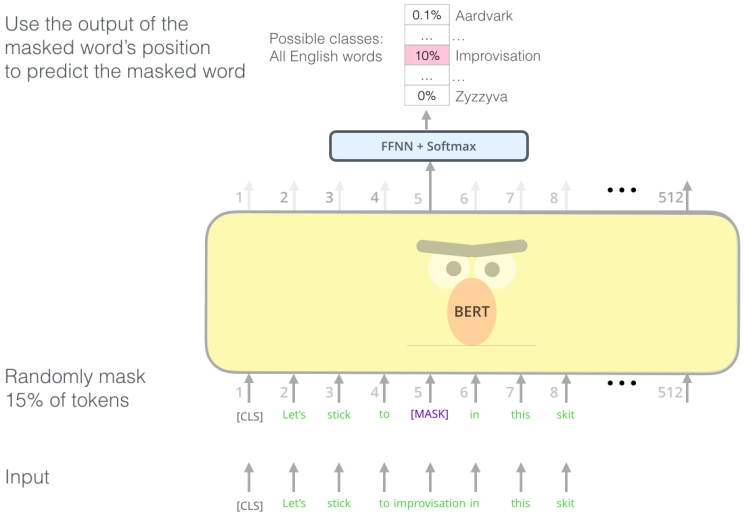

Рис. 2.2. Поиск "замаскированного"слова

В данной работе использовался предобученный BERT от DeepPavlovAI RuBert [2], который обучался на русских текстах.

Для задачи определения последовательности предложений в модель часть предложений поступает в последовательно, а другая часть в случайном порядке. И модель пытается предсказать какие из этих предложений находятся рядом в оригинальном тексте 2.3.

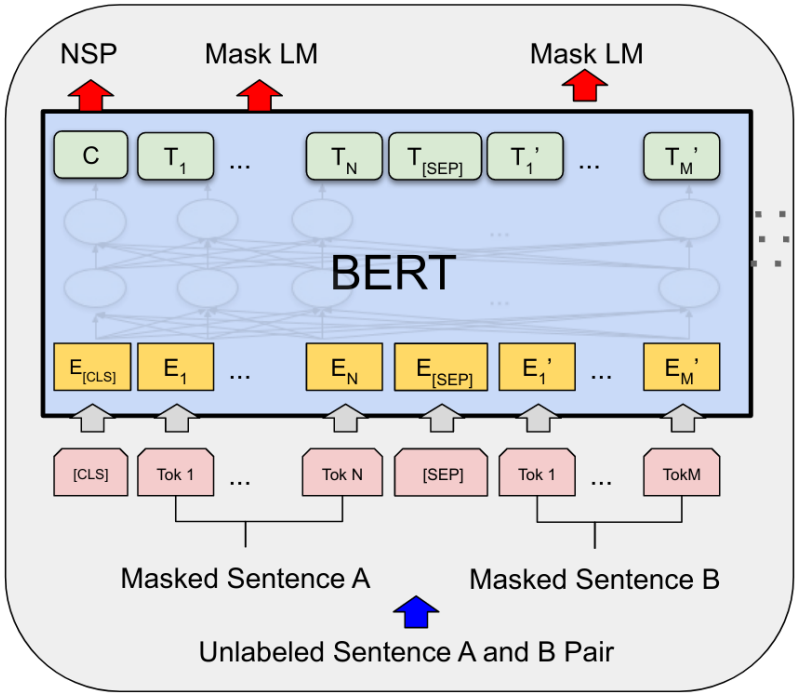

Рис. 2.3. Схема работы BERT

BERT обрабатывал ВКР, которые написаны у сотрудников. С его помощью мы получили эмбединги предложений из ВКР. Получилось очень много предложений и надо получить короткое описание текстов фиксированного размера для каждого сотрудника. Чтобы сделать это можно взять самые типовые предложения, чем и занимается библиотека BERT Extractive summarizer [3].

BERT Extractive summarizer ищет эмбединги предложений, которые находятся рядом в векторном пространстве и объединяет их в группы (кластеры). И результатом общений текстов будет самое близкое предложение к центру кластера 2.4, что позволяет получить краткое содержание текста. Таким образом получается описание текста в виде эмбединга, для удобства работы алгоритмически, и текста, для проверки в ручную, остаётся сопоставить описание сотрудника и текстового задания.

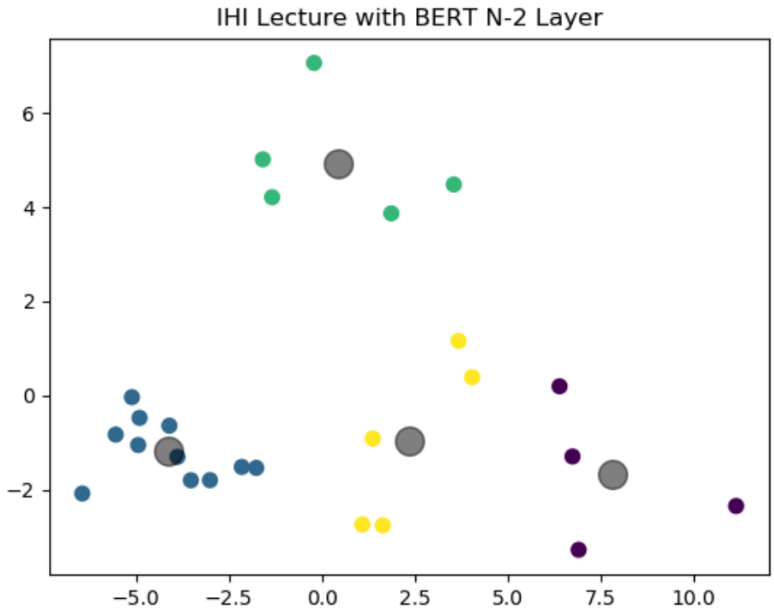

Рис. 2.4. Пример поиска содержания предложений

2.4. Возможные методы подбора исполнителя по ТЗ

Для подбора исполнителя сравнивались эмбединнги текстового задания и сотрудника. Для лучшего подбора подходящего человека были разработаны разные функции поиска расстояний. Результатом работы алгоритма является исполнитель с наименьшим расстоянием между описанием задания и компетенциями исполнителя. Реализованные функции расчёта расстояний:

- Средняя разница элементов – в данном методе поиска расстояний, отрицательное расстояние может скомпенсировать положительное.
- Модуль разницы элементов (Манхэтонновское расстояние) – этот метод поиска расстояний будет штрафовать за расстояния, которые находятся близко сильнее, чем расстояние Евклида.
- Квадрат разницы элементов (Евклидово расстояние) – этот метод поиска расстояний будет штрафовать за расстояния, которые находятся далеко сильнее, чем Манэтонновское расстояние.

Нужно было сопоставить 10 эмбедингов текстового описания задания и 10 эмбедингов сотрудника. Для этого было реализовано несколько алгоритмов:

- Все со всеми. Каждый эмбединг текста сопоставляется с эмбеднигами сотрудника и считается сумма их расстояний.

- Поиск минимального с повторами. Для каждого эмбеддинга текста находится минимальное расстояние с эмбедингом сотрудника и считается сумма каждой такой пары.
- Поиск минимального без повторов. Для каждого эмбеддинга текста находится минимальное расстояние с эмбедингом сотрудника, если уже была пара с таким эмбедингом сотрудника, то берётся другой эмбединг с минимальный расстоянием, и считается сумма каждой такой пары.
- Усреднение эмбеддингов. Эмбеддинги сотрудника и текста усредняются и считается расстояние между ними.

Схемы подбора находятся в Приложении А.

Глава 3. Разработка и тестирование системы

3.1. Разработка интерфейсов

Для удобства был сделан графический интерфейс 3.1. С помощью него можно ввести текстовое описание задачи и подобрать походящего исполнителя.

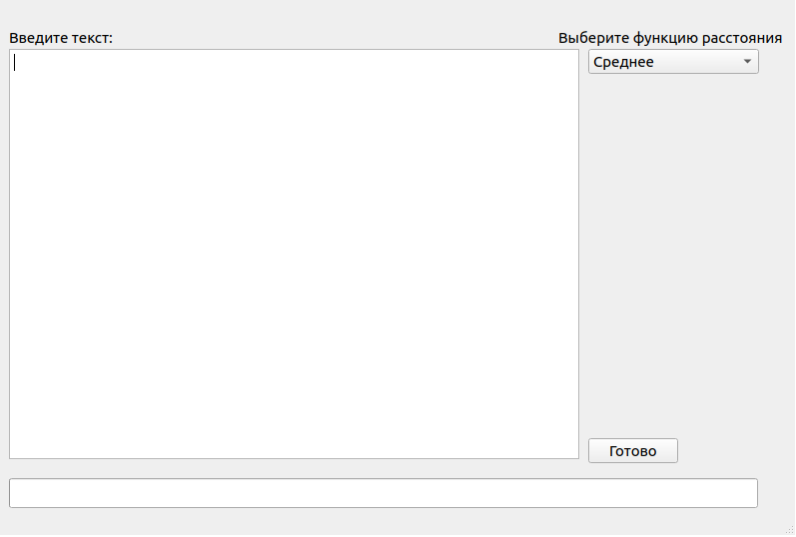

Рис. 3.1. Интерфейс приложения

На данном окне есть текстовое поле куда вносится весь текст задания.

Для выбора функции ошибки есть ComboBox, в котором можно выбрать одну из следующих функций ошибки:

- Среднее расстояние
- Манхэтонновское расстояние
- Евклидово расстояние

При нажатии кнопки Готово начинается подбор исполнителя, это занимает в среднем примерно 5 минут, для удобства отслеживания времени работы алгоритма был добавлен ProgressBar в низ приложения.

3.2. Тестирование приложения

Для тестирования приложения была выбрана выпускная квалификационная работа Абросимовой П. С. с темой «Разработка средств автоматизации

расширения онтологии на основе данных интернет-источников» руководителем была Лядова Л. Н. Для тестирования в коде было сделано так чтобы, данный научный сотрудник не был решением алгоритма. Результаты работы разных алгоритмом и функций расстояния 3.1.

Таблица 3.1. Результатов работы программы

Тип алгоритма	Среднее расстояние	Манхэттоновское расстояние	Расстояние Евклида
Все со всеми	Кычкин А. В.	Кушев В. О.	Кушев В. О.
Поиск минимального с повторами	Божья-Воля А. А.	Кычкин А. В.	Кычкин А. В.
Поиск минимального без повторов	Божья-Воля А. А.	Божья-Воля А. А.	Божья-Воля А. А.
Усреднение эмбеддингов	Кычкин А. В.	Кузнецов Д. Б.	Кузнецов Д. Б.

Получилось 4 сотрудника Кычкин А. В., Кушев В. О., Божья-Воля А. А., Кузнецов Д. Б., из которых 3 сотрудника работают на кафедре информационных технологий в бизнесе и похожей сферой интересов с Лядова Л. Н.

Заключение

В данной работе была разработана информационная система, целью которой было проверить гипотезу о том можно ли подобрать исполнителя под текстовое описание тендера.

Для этого было проведено проектирование и развёртывание базы данных. На основании базы данных была разработана и реализована информационная система, с помощью которой собирались данные с сайта ВШЭ и заполнились профили сотрудников НИУ ВШЭ.

Лучший результат показало расстояние Евклида.

В рамках работы в систему были загруженные реальные данные. Дальше было проведено тестирование и было показано, что разные функции тестирования дают разные результаты.

На основании текстового описания был произведён поиск путём расчёта расстояния между подаваемым текстом и профилем сотрудников. Было предложено несколько методов расчёта этого расстояния и было показано, что расстояние эвклида показало лучшее качество работы на нескольких примерах. На основании разработанной системы в дальнейшем требуется провести более детальное исследование функций расстояния и выбрать ту, которая покажет наилучшее качество на большой выборке данных.

Библиографический список

1. *Кафедра ИТБ НИУ ВШЭ-Пермь*. Правила написания и оформления курсовых работ студентов основной образовательной программы бакалавриата «Программная инженерия» по направлению подготовки 09.03.04 Программная инженерия. — 2020. — URL: `https://www.hse.ru/data/2020/12/10/1356616064/%D0%9F%D1%80%D0%B0%D0%B2%D0%B8%D0%BB%D0%B0%20%D0%BF%D0%BE%D0%B4%D0%B3%D0%BE%D1%82%D0%BE%D0%B2%D0%BA%D0%B8%20%D0%9A%D0%A0%20%D0%9F%D0%98%20(11.2020).pdf` (дата обр. 25.03.2021).
2. *Kuratov Y.*, *Arkhipov M.* Adaptation of Deep Bidirectional Multilingual Transformers for Russian Language. — 2019.
3. *Miller D.* Leveraging BERT for Extractive Text Summarization on Lectures. — 2019.
4. *Das B.*, *Chakraborty S.* An Improved Text Sentiment Classification Model Using TF-IDF and Next Word Negation. — 2018.
5. *Devlin J.*, *Chang M.-W.*, *Lee K.*, *Toutanova K.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. — 2019.

ПРИЛОЖЕНИЕ А

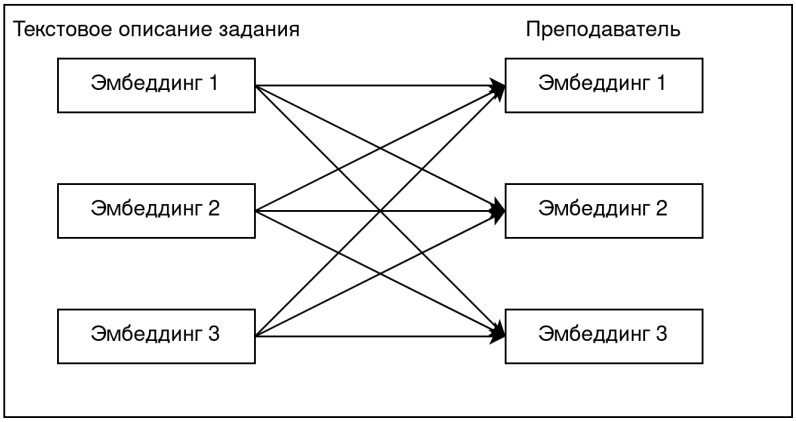

Рис. 3.2. Все со всеми

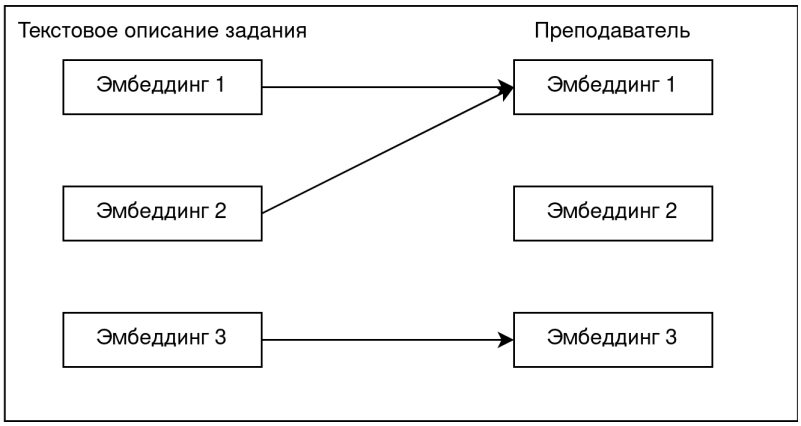

Рис. 3.3. Поиск минимального с повторами

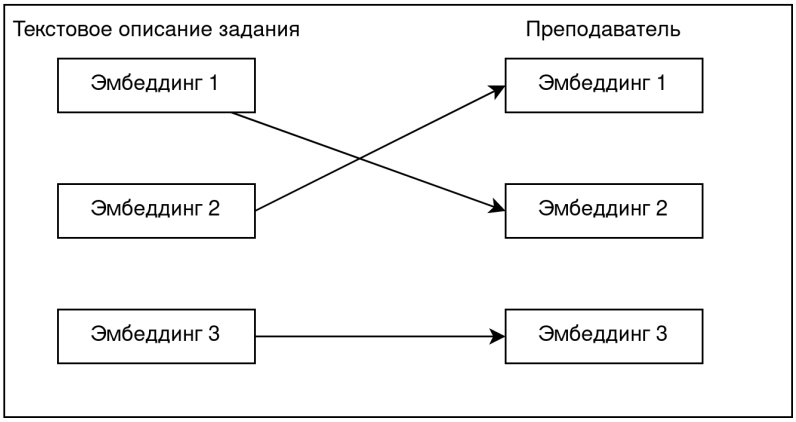

Рис. 3.4. Поиск минимального без повторов

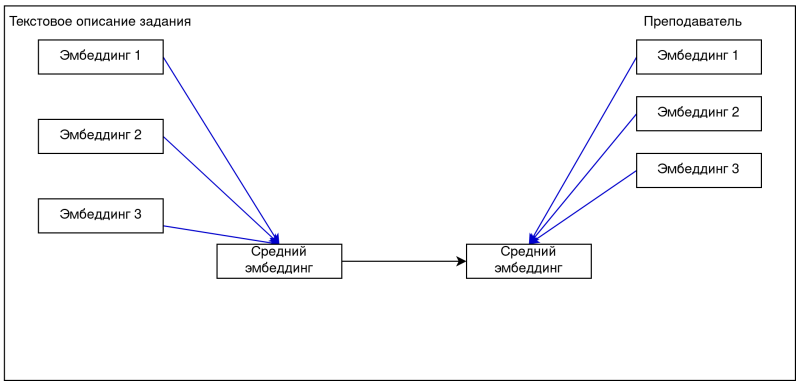

Рис. 3.5. Усреднение эмбеддингов