Описание системы интеллектуального поиска вакансий на основе технологии машинного обучения (Smart Humax)

# Предназначение и общее описание системы

Система интеллектуального подбора резюме предназначена для помощи кадровому работнику при подборе кандидатов на вакансии. Кадровый работник создает вакансию, вводит там некоторые параметры. На основе этих параметров система подбирает в базе резюме (сайт headhunter.ru или аналогичные) несколько резюме (примерное количество подобранных резюме - 20). После этого кадровый работник выбирает из предложенных вакансий подходящие. Информация о выборе загружается в систему и используется для ее обучения. В результате с течением времени система буде выдавать все более подходящие резюме.

# Состав системы

1. База на основе 1с:Документооборот (WFP)
2. Модуль машинного обучения на Python
3. База данных на СУБД Mongo DB
4. HTTP сервис на ОС Linux и обработчики вспомогательных функций

# Взаимодействие компонентов системы

После того как кадровый работник создал вакансию, заполнил ее параметры и нажал кнопку интеллектуального подбора, из базы 1с идет http запрос к http сервису. Запрос обрабатывается кодом на языке Python. В процессе обработки запроса идет обращение к базе на СУБД Mongo DB, куда предварительно загружены все резюме с сайта вакансий. С помощью специально обученной нейронной сети (модуль машинного обучения) из всех вакансий выбираются подходящие. После этого данные о выбранных вакансия передаются в базу 1с. В 1с эти данные записываются в справочник нфп\_Резюме (предварительное название).

Когда кадровый работник из подобранных резюме выбирает подходящих кандидатов, информация о выборе также передается в модуль машинного обучения, который ставит пометки в базе Mongo DB на выбранных вакансиях. Это необходимо для обучения системы.

Это целевая картина, на данный момент не весь функционал реализован, некоторый функционал реализован в упрощенном виде.

# 1c документооборот (WFP)

База 1с документооборот доработана. В числе доработок присутствует справочник нфп\_Вакансии (уже реализовано). На форме элемента справочника должна быть кнопка для подбора резюме с помощью систем интеллектуального подбора. При нажатии на кнопку формируется HTTP запрос к HTTP сервису (компонент системы).

HTTP запрос передается методом POST, параметры запроса берутся из реквизитов справочника нфп\_Ваканасии (ВозрастОт, ВозрастДо, Пол, ОпытРаботы, Образование, Требования, Обязанности).

HTTP запрос после обработки возвращает массив структур, содержащих подобранные резюме. Обработка HTTP запроса более подробно будет описана в описании модуля машинного обучения.

После обработки на основе полученных данных должны быть сформированы элементы резюме (справочник нфп\_Резюме, справочник не реализован, возможно это буде какая-то другая структура).

Далее кадровый работник должен иметь возможность создать кандидата (справочник нфп\_Кандидаты) на основании загруженного резюме. При создании кандидата информация о том, что кандидат создан на основании данного резюме, должна отправляться с помощью HTTP запроса на HTTP сервис. Эта информация должна быть записана в Mongo DB использована для обучения нейронной сети (модуль машинного обучения). Данный функционал не реализован

Интерфейс подбора, HTTP запрос, обработка полученных данных не реализованы.

Пример HTTP запроса реализован в обработке ТестHTTPЗапроса.efp Обработка вместе с кодом других модулей выложена в git: <https://github.com/darkh14/HR-Product.git>

# База данных на СУБД Mongo DB

База выполняет роль хранилища коллекции резюме, загруженных из сайтов-баз резюме (headhunter и др.). База размещена в облаке на ОС Linux. Управление базой осуществляется посредством библиотеки PyMongo языка программирования Python. Название базы – ‘hr’. Название основной коллекции, содержащей полученные из сайта резюме – ‘cv’. Обновление данных в коллекции осуществляется по регламенту или принудительно (обновление данных реализовано не полностью).

Принудительное обновление осуществляется путем запроса к HTTP сервису с параметром RequestType = update\_cvs (функционал не реализован).

Вместе в резюме в отдельном поле хранятся метки о том, что по данному резюме создан кандидат. (не реализовано)

Также в базе могут храниться вспомогательные и частично обработанные данные для машинного обучения (не реализовано)

Состав полей резюме на данный момент:

* address
* gender
* salary
* valuta
* age
* position
* about\_me
* category
* specialization
* resume\_link

# Модуль машинного обучения

Данный модуль написан на языке программирования Python. Модуль осуществляет интеллектуальный подбор резюме, хранящихся в базе Mongo DB, на основании запроса из базы 1с.

В модуле заложено 2 алгоритма поиска:

* Простой поиск
* Интеллектуальный поиск

Простой поиск осуществляется на основании прямого сравнения параметров, полученных из 1с и значений в строка таблицы резюме. Длинные строки сравниваются с использованием библиотеки difflib языка Python. Данный поиск дает гораздо менее точный результат и не обучается. Но его достоинством является простота реализации и стабильность работы.

Интеллектуальный поиск основан на нейронной сети. Библиотеки, используемые при реализации интеллектуального поиска: numpy, pandas, sklearn, keras, tensorflow и другие.

Архитектура нейронной сети на данный момент не до конца продумана. Проблемы архитектуры: большинство данных резюме текстовые, нейронные сети работают только с числовыми данными, простые нейронные сети не подразумевают параметризацию (работают только для одного набора значений параметров). Поэтому сейчас включен простой поиск

Так же для нейронной сети необходимо периодическое обучение (подбор коэффициентов нейронов). Обучение должно осуществляться по регламенту или принудительно (не реализовано).

# HTTP сервис и обработчики вспомогательных функций

HTTP сервис реализован с помощью технологии uwsgi, которая осуществляет связь HTTP сервиса и модуля с кодом на Python.

Процедура обработки HTTP запроса принимает POST запрос с параметрами, закодированными в json. Обязательный параметр RequestType – тип запроса. В зависимости от типа запроса осуществляется различная обработка запроса.

Варианты обработки запроса:

* get\_fitting\_cvs– подбор подходящих резюме. Подбор резюме по параметрам из запроса и передача данных о резюме в ответе
* get\_all\_ids – возврат всех резюме (не реализовано)
* get\_id – возврат одного резюме (не реализовано)
* leanr\_NN – обучить нейронную сеть (не реализовано)
* update\_cvs – обновить все резюме с сайта (не реализовано)
* Возможны другие варианты

# Описание python модулей проекта

* main.py – модуль для теста и отладки, имитирует выполнение функции обработки http запроса
* wsgi.py – модуль, содержащий функцию обработки http запроса. Выполняется с помощью uwsgi. Обращается к модулю http\_procession.py для дальнейшей обработки http запроса
* http\_procession.py – модуль для обработки http запроса. В зависимости от параметра RequestType выполняет различные действия и обращается к другим модулям
* mongo\_connection.py – модуль для связи, получения и записи данных в базу Mongo DB
* machine\_learnin\_py – модуль машинного обучения. Осуществляет поиск резюме, обучение и настройку нейронной сети
* web\_parser.py – модуль для загрузки резюме с сайта (реализовано частично, нет в проекте)

Модули выложены в Git по адресу: <https://github.com/darkh14/HR-Product.git>

Сейчас HTTP сервис и все модули развернуты в облаке Google:

Ip 34.74.148.6

Обращение к HTTP сервису: 34.74.148.6: 8000

База Mongo DB развернута на том же облаке

Строка соединения: mongodb://Admin:123@34.74.148.6:27017/Admin?authSource=admin

Облако не работает постоянно, т. к. это личное облако Зюзя П. А. )))