Министерство образования и науки Российской Федерации

Московский авиационный институт (национальный исследовательский университет)

ЖУРНАЛ

ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Наименование практики: *вычислительная*
Студенты: А. А. Довженко, И. Е. Найденов
Факультет №8, курс 2, группа 7

Практика с 29.06.18 по 12.07.18

Москва, 2018

ИНСТРУКЦИЯ

о заполнении журнала по производственной практике

Журнал по производственноий практике студентов имеет единую форму для всех видов практик.

Задание в журнал вписывается руководителем практики от института в первые три-пять дней пребывания студентов на практике в соответствии с тематикой, утверждённой на кафедре до начала практики. Журнал по производственной практике является основным документом для текущего и итогового контроля выполнения заданий, требований инструкции и программы практики.

Табель прохождения практики, задание, а также технический отчёт выполняются каждым студентом самостоятельно.

Журнал заполняется студентом непрерывно в процессе прохождения всей практики и регулярно представляется для просмотра руководителям практики. Все их замечания подлежат немедленному выполнению.

В разделе «Табель прохождения практики» ежедневно должно быть указано, на каких рабочих местах и в качестве кого работал студент. Эти записи проверяются и заверяются цеховыми руководителями практики, в том числе мастерами и бригадирами. График прохождения практики заполняется в соответствии с графиком распределения студентов по рабочим местам практики, утверждённым руководителем предприятия. В разделе «Рационализаторские предложения» должно быть приведено содержание поданных в цехе рационализаторских предложений со всеми необходимыми расчётами и эскизами. Рационализаторские предложения подаются индивидуально и коллективно.

Выполнение студентом задания по общественно-политической практике заносятся в раздел «Общественно-политическая практика». Выполнение работы по оказанию практической помощи предприятию (участие в выполнении спецзаданий, работа сверхурочно и т.п.) заносятся в раздел журнала «Работа в помощь предприятию» с последующим письменным подтверждением записанной работы соответствующими цеховыми руководителями. Раздел «Технический отчёт по практике» должен быть заполнен особо тщательно. Записи необходимо делать чернилами в сжатой, но вместе с тем чёткой и ясной форме и технически грамотно. Студент обязан ежедневно подробно излагать содержание работы, выполняемой за каждый день. Содержание этого раздела должно отвечать тем конкретным требованиям, которые предъявляются к техническому отчёту заданием и программой практики. Технический отчёт должен показать умение студента критически оценивать работу данного производственного участка и отразить, в какой степени студент способен применить теоретические знания для решения конкретных производственных задач.

Иллюстративный и другие материалы, использованные студентом в других разделах журнала, в техническом отчёте не должны повторяться, следует ограничиваться лишь ссылкой на него. Участие студентов в производственно-технической конференции, выступление с докладами, рационализаторские предложения и т.п. должны заноситься на свободные страницы журнала.

Примечание. Синьки, кальки и другие дополнения к журналу могут быть сделаны только с разрешения администрации предприятия и должны подшиваться в конце журнала.

Руководители практики от института обязаны следить за тем, чтобы каждый цеховой руководитель практики перед уходом студентов из данного цеха в другой цех вписывал в журнал студента отзывы об их работе в цехе.

Текущий контроль работы студентов осуществляется руководители практики от института и цеховыми руководителями практики заводов. Все замечания студентам руководители делают в письменном виде на страницах журнала, ставя при этом свою подпись и дату проверки.

Результаты защиты технического отчёта заносятся в протокол и одновременно заносятся в ведомость и зачётную книжку студента.

Примечание. Нумерация чистых страниц журнала проставляется каждым студентом в своём журнале до начала практики.

С инструкцией о заполнении журнала ознакомились:

« » _____ 2018 г.
(дата)

Студенты _____
(подпись)

ЗАДАНИЕ

кафедры 806 по вычислительной практике:
Разработать Android приложение, подбирающее релевантные картинку/картинки к статье.

ТАБЕЛЬ ПРОХОЖДЕНИЯ ПРАКТИКИ

Дата	**Содержание или наименование проделанной работы**	**Место работы**	**Время работы**		**Подпись цехового руководителя**
			Начало	Конец	
29.06.2018	Получение задания	МАИ	9:00	18:00	
02.07.2018	1. Завести репозиторий на github; 2. Разработать Use Case диаграмму поведения пользователя при работе с приложением (frontend); 3. Найти новостные источники или размеченный data-set картинок с тегами (backend); 4. Изучить алгоритмы и методы соответствия картинки тексту; 5. Обработать полученный data-set (парсинг); 6. Создать прототип интерфейса (форму);	МАИ	18:00	20:00	
05.07.2018	1. flickr. Причесать существующий код – написать API, оформить библиотеку; 2. Написать сервер; 3. Связать API с сервером; 4. Познакомиться с elasticsearch или кластеризацией;	МАИ	16:00	18:00	

Продолжение на следующей странице

Дата	**Содержание или наименование проделанной работы**	**Место работы**	**Время работы**		**Подпись цехового руководителя**
			Начало	Конец	
08.07.2018	1. Поднять клиента; 2. Доустановить elasticsearch и искать в датасете; 3. Дозапрос доп. картинки, если первая не понравилась пользователю; 4. Разобраться с андроидом;	МАИ	18:00	20:00	
10.07.2018	1. Индексация в эластике; 2. Добавить/перевести русские теги; 3. Удалять дубли; 4. Карусель результатов на клиенте и улучшение дизайна приложения;	МАИ	17:00	20:00	
12.07.2018	Сдача журнала	МАИ	9:00	18:00	

Отзывы цеховых руководителей практики

ПРОТОКОЛ

ЗАЩИТЫ ТЕХНИЧЕСКОГО ОТЧЁТА

по *производственной практике*

студентами:
Довженко Анастасия Александровна
Найденов Иван Евгеньевич

Слушали:	**Постановили:**
Отчёт практикантов	считать практику выполненной и защищённой на Общая оценка: _____

Руководители: Зайцев В. Е. _____

Кухтичев А. А. _____

Дата: 12 июля 2018 г.

МАТЕРИАЛЫ ПО РАЦИОНАЛИЗАТОРСКИМ ПРЕДЛОЖЕНИЯМ

1. Автоматическая сборка приложения и логирование;
2. Избавиться от ngrok'а, захоститься;
3. Использовать датасет вместо обращений к flickr'у;
4. Улучшить дизайн;
5. Ускорить работу сервера, сейчас он работает непозволительно долго;

ТЕХНИЧЕСКИЙ ОТЧЁТ ПО ПРАКТИКЕ

Архитектура

Frontend: Java

- Предоставляет пользователю поле для ввода статьи
- Передаёт текст на сервер
- Принимает список ссылок на картинки, отображает картинки пользователю в галереи

Backend: Python

- Принимает текст, выделяет из него теги по алгоритму TextRank
- Обращается к flickr'у по API, ищет картинки, помеченные выделенными тегами
- Передает список ссылок на картинку клиенту

Описание

При запуске приложения открывается окно с полем для ввода статьи. После ввода текста пользователь нажимает на кнопку «Send». Далее текст отсылается на сервер. Там происходит обработка текста: перевод (если текст не на английском языке), лемматизация, приведение к нижнему регистру, удаление стоп-слов. После обработки из текста выделяются 5 тегов с помощью алгоритима TextRank. Далее находятся все подмножества множества тегов, и происходит попытка найти картинки, соответствующие каждому подмножеству. Когда 5 картинок найдены, сервер отправляет клиенту ссылки на них. Клиент отображает пользователю найденные картинки в виде галереи.

Реализация

Frontend:
Клиентское приложение состоит из двух лайаутов (окон). В первом происходит получение данных от пользователя, во втором отображение изображений найденных сервером. Исходя из этого, всё о чем пришлось думать при написании клиента – это выполнение запросов и отображение изображений в виде галереи.В текущей версии Android запросы осуществлять можно только асинхроно, наследуясь от класса AsyncTask, однако, на первый вгляд тривиальная задача оказалась нетривиальной и мы решили, что лучше использовать AsyncHttpClient.
После получения ссылок на изображения их, очевидно, нужно загрузить и отобразить. С однрой стороны можно сделать это, опять же, вручную выполнив http запросы, однако, был способ получше, и мы им воспользовались – это библиотека Picasso, разработанная специально для работы с изображениями. К примеру простая загрузка и одновременно отображение картинки может выглядить следующим образом:

```
Picasso.get().load("http://url.com/test.jpg").into(imageView);
```

Непосредственно отображение списка изображений в виде галереи так же задача не такая уж и тривиальная, ведь решаться может очень многими способами. Создать линейный лайаут, помещенный в HorizontalScrollView и динамически создавать в нем imageView как нам показалось, не самая худшая идея, и мы сделали именно так. Выглядит, конечно, просто однако в процессе разработки мы пробовали и другие подходы, а так же сторонние библиотеки, реализующие галереи изображений,

которые, зачастую, предоставляли плохо работающий функционал и, в конце концов, этот способ показался нам оптимальным.

Backend:
Работа сервера логичным образом разбивается на два этапа: первый – извлечь ключевые слова из текста, второй – найти картинки по этим словам.
На первом этапе осуществляется препроцессинг данного текста. Если текст написан на отличном от английского языка, то он переводится на английский с помощью библиотеки googletrans. Текст приводится к нижнему регистру, чтобы исключить дублирование тегов. Текст очищается от знаков припинания и стоп-слов, которые могут повлиять на конечный результат. Далее текст разбивается на токены, токены лемматизируются, чтобы слова разных частей речи не считались как разные по смыслу. Потом применяем алгоритм TextRank к полученному набору слов. Есть и другие алгоритмы извлечения ключевых слов, например TF-IDF и RAKE. Статистические алгоритмы не подошли нам, потому что у нас есть только один текст, а не коллекция, алгоритмы машинного обучения показались нам слишком сложными в реализации. Тем более, что TextRank сразу показал хорошие результаты на тестовых данных. В основе этого алгоритма лежит алгоритм PageRank, который используется для ранжирования страниц в поисковых системах. Суть его в том, что строится граф, вершинами которого являются слова из текста, а ребра между ними имеют вес, равный похожести вершин, которым оно инцидентно. Потом считаются значения PageRank для построенного графа. По связям между вершинами графа PageRank назначает каждой из них рейтинг, учитывая две вещи: количество ребер, которые идут из других вершин и рейтинг этих ребер. Получается, что одни вершины «рекомендуют» другие, а сила рекомендации вычисляется рекурсивно на основе рейтингов вершин. Связь между двумя предложениями будет определяться по количеству в них одинаковых слов. В отличие от оригинального PageRank, здесь надо учитывать вес ребер, который будет зависеть от этого количества. Тогда всё получается так же логично, как и для веб-страниц: одно слово рекомендует другое, а самыми важными являются те, которые содержат информацию из нескольких других предложений.
Поиск картинок осуществляется через Flickr API, клиент которого реализован в библиотеке flickrapi. Чтобы гарантировано найти картинки, мы генерируем все подмножества нашего множества тегов и поочередно отправляем запросы Flickr'у, пока найденных картинок не будет ровно 5 или пока множество подмножеств тегов не станет пустым. При каждом запросе проверяем, что картинка по адресу действительно существует и что её еще нет в найденных нами картинках.

Тестирование

При разработке производилось исключительно ручное тестирование, что впоследствии сыграло с разработчиками злую шутку. Авторы сделали выводы из допущенных ошибок и впредь будут больше внимания уделять тестированию.

Ссылка на GitHub

https://github.com/tutkarma/AutoPicturer