Санкт-Петербургский государственный университет
Факультет Прикладной математики - Процессов управления

Кафедра Технологии программирования

Пекша Любовь Станиславовна

Выпускная квалификационная работа бакалавра

Сентимент-анализ упоминаний персоналий в СМИ

Направление СВ.5005.2015
Прикладная математика, фундаментальная информатика и программирование

Научный руководитель:
кандидат технических наук Блеканов И.С.

Рецензент:
старший преподаватель Давыденко А.А

Санкт-Петербург
2019

SAINT-PETERSBURG STATE UNIVERSITY
Факультет Прикладной математики - Процессов управления

Programming Technology

Peksha Liubov

Sentiment analysis of personalities mentioned in the media

Bachelor's Thesis

Scientific supervisor:
Candidate of Engineering Blekanov I.S.

Reviewer:
lecturer Davydenko A.A.

Saint-Petersburg
2019

Оглавление

Введение

Автоматический анализ интернет-медиа и блогосферы достаточно хорошо исследованная область, первые работы в которой можно отнести ещё к началу 2000-х годов[1, 2]. Однако, не смотря на долгую историю, данные исследования не теряют актуальности. Методы, применяемые для анализа новостного потока, часто наследуются из близких областей: информационного поиска, машинного обучения, анализа естественных языков (Natural Language Processing), следуя текущим трендам в соответствующих областях.

С другой стороны все большее количество специалистов из различных гуманитарных областей: социологии, политологии, филологии и лингвистики из-за развития и популяризации различных инструментов анализа данных, а в особенности инструментов NLP, начинают использовать эти методы в своих исследованиях, создавая на пересечении отдельных наук новую междисциплинарную область — цифровые гуманитарные науки. Digital Humanities[3] (DH, цифровые гуманитарные науки) — стремительно развивающееся направление. Основной причиной такого роста может служить значительное "снижение уровня входа" в NLP, а также появление и распространение прикладных пакетов, упрощающих обработку и анализ естественного языка (stylo[4], nltk[5], pymorphy2[6], deepPavlov[7]). Одна из проблем, стоящая перед филологами и лингвистами, заключается в том, что зачастую анализ какого-то конкретного корпуса классическими методами — например, выделение и подсчет различных частей речи — представляется крайне трудоемким, а количество необходимых высококвалифицированных и дорогостоящих специалистов линейно зависит от объема подобного корпуса.

Большую часть рутинной работы, однако, можно делегировать тому или иному алгоритму машинного обучения, оставив специали-

сту только необходимость делать высокоуровневые выводы на основе полученной информации. Такой задачей является выделение некоторого контекста вокруг персонажа художественных произведений и анализ эволюции этого контекста с течением времени. Этой задаче посвящен целый ряд исследований[8, 9], который показывают впечатляющие результаты. Идея данной выпускной квалификационной работы заключается в том, чтобы попробовать переложить имеющиеся модели, посвященные анализу художественных произведений на другую доменную область — новостной поток современных интернет-СМИ, осуществив некоторый синтез наработок специалистов в Digital Humanities и новейших достижений в области анализа данных.

Данная задача не только интересна с научной точки зрения, но и имеет практическое применение. Например, сравнение упоминаний персоны в различных СМИ или получение общего представления о личности путем анализа новостных текстов.

Постановка задачи

Особый интерес представляет собой анализ персоналии во времени, то есть изучение того, как изменяется новостная повестка относительно персоны с течением времени.

Данная тема предоставляет широкий спектр возможных направлений для работы такие как выделение траекторий персоналий (ранжированного по времени набора статей о персоналии), выявление долгосрочных трендов траектории, а также анализ эмоциональной окраски траекторий.

Отсутствие размеченных данных и способа формально измерить качество работы алгоритма усложняют работу в данной сфере. Вследствие этого возникает необходимость реализовать модель способную обучаться без учителя, а также провести эксперимент, позволяющий оценить качество реализованной модели.

Основной целью проекта является реализация существующей модели решающей задачу тематико-эмоционального анализа траекторий персоналий в долгосрочном периоде. А также применение модели для публицистических текстов, ее модернизация и анализ полученных результатов.

Для достижение цели ставятся следующие задачи:

1. Формирование обучающей выборки
2. Построение базовой модели, способной выделять долгосрочные тренды и тематики
3. Модернизация модели
4. Анализ полученных результатов

Обзор литературы

Имеющиеся модели, реализующие анализ траекторий персоналий, посвящены анализу художественной литературы[8, 9].

Особый интерес представляет собой работа Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships[8]. В этой работе описывается модель **RMN**, которая совместно обучает набор глобальных дескрипторов отношений между героями из художественных текстов, а также сопоставляет каждому отрывку текста веса дескрипторов, которые отражают насколько хорошо каждое из слов-дескрипторов описывает траекторию в текущий момент времени.

RMN превзошла модель **HTMM**[9], решающую сходную задачу, а также модели тематического моделирования **LDA**[10] и **NUBBI**[11], которые способны выделять тематики в неразмеченном тексте, но не предназначены для построения траекторий в долгосрочном периоде. Авторами модели провели два эксперимента с помощью краудсорсинговой площадки. Участники первого эксперимента оценивали интерпретируемость и когерентность получаемых дескрипторов. Участники второго оценивали соответствие получаемых траекторий краткому пересказу художественного текста.

Данная модель решает следующие подзадачи:

1. Выделение интерпретируемых дескрипторов — тематического базиса для составления траекторий

2. Создание модели, способной сопоставлять траектории распределение на дескрипторах (веса дескрипторов), которое отражает долгосрочные тренды в текстах статей траектории а также их изменение

RMN хорошо справляется со своей задачей, выделяя интерпре-

тируемые дескрипторы, среди которых достаточно четко можно выделить эмоциональные (love, sadness, violence, etc.).

Несмотря на свои достоинства, архитектура данной модели имеет ряд недостатков, которые будут рассмотрены далее.

1. Данные

Для формирование обучающей выборки необходимо выполнить следующие шаги

1. Сбор данных с сайтов СМИ
2. Выделение траекторий персоналий - ранжированного по времени набора статей о персоналии
3. Обработка текста статей с целью выделить релевантную информацию

Для дальнейшей работы необходимо собрать данные с сайтов СМИ. В открытом доступе есть датасеты со статьями различных СМИ. Но в силу того, что такие датасеты не предоставляют информацию о дате выхода статьи, возникла необходимость собрать необходимые данные непосредственно с новостных сайтов.

Для работы были выбраны следующие новостные интернет-издания: Lenta.ru[15], Дождь[16], Meduza[17], РИА Новости[18]. Данные СМИ имеют внушительный архив данных, а также ведут свою деятельность продолжительный период времени.

1.1. Сбор данных

Сбор данных осуществлялся с помощью Scrapy[19]. Это написанная на Python платформа, которая нацелена на простой, быстрый и автоматизированный обход (краулинг) веб-страниц, имеющий большую популярность. Одним из главных преимуществ Scrapy является то, что он построен поверх Twisted, асинхронного сетевого фреймворка. Асинхронность означает, что не нужно ждать завершения запроса, прежде чем сделать еще один, это позволяет добиться высокого уровня производительности. Тот факт, что Scrapy реализован с

использованием неблокирующего (асинхронного) кода для параллелизма, делает его одним из самых эффективных фреймворков для краулинга.

В ходе работы по сбору данных возникло несколько проблем.

Первая из них — отсутствие архива статей на сайте СМИ Meduza[17]. Данная проблема была решена с помощью группы ВКонтакте данного интернет-издания. Ссылки на новостные статьи были собраны при помощи VK API — интерфейса, который позволяет получать информацию из базы данных vk.com. Для удобства работы с VK API была использована библиотека vk для Python. Эта библиотека предоставляет удобный интерфейс для работы с VK API, а также не требует авторизации.

Еще одна проблема возникла из-за того, что сайт телеканала Дождь блокирует запросы от краулера (программы, осуществляющей сбор данных), если эти запросы поступают от него слишком часто. Из-за этого краулер без дополнительных изменений будет скачивать лишь часть доступной информации. Выход из этой ситуации – использовать разные useragent для сетевых запросов. Useragent — это клиентское приложение, использующее определённый сетевой протокол. При посещении веб-сайта клиентское приложение обычно посылает веб-серверу информацию о себе. Это текстовая строка, являющаяся частью HTTP запроса, обычно включающая такую информацию, как название и версию приложения, операционную систему компьютера и язык. Библиотека fake_useragent для Python позволяет создавать случайные useragent для каждого запроса на сайт.

Для хранения полученных данных используется база данных на основе SQLite. Для работы с базой данных используется SQLAlchemy. Это набор инструментов SQL с открытым исходным кодом и ORM (технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования)

для языка программирования Python.

1.2. Формирование траекторий

Для формирования траекторий необходимо уметь извлекать из текста имена упоминаемых в нем людей. Для этого требуется использование программы, решающей задачу извлечения именованных сущностей. Извлечение именованных сущностей — это класс подзадач извлечения информации, цель которой найти и классифицировать упоминания именованных сущностей в неструктурированном тексте по заранее определенным категориям, таким как имена людей, организации, адреса, даты и т. д. Библиотека DeepPavlov предоставляет модель, которая решает задачу извлечения имен для текстов на русском языке.

Далее необходимо сопоставить извлеченные имена из разных статей друг с другом чтобы определять, что разные статьи относятся к одному и тому же человеку. Для этого необходимо привести имя человека к нормальной форме. Для этой цели используется библиотека pymorphy2. Затем полученные слова сортируются в алфавитном порядке. Разделение на имена и фамилии не используется, так как задача определения фамилий работает недостаточно хорошо, особенно для иностранных фамилий. К тому же, благодаря деловому стилю написания новостных статей, при первом упоминании человека как правило используется его полные имя и фамилия. Более того, упоминание в новостной статье фамилии без имени обычно используется в устойчивых словосочетаниях, например ”пакет Яровой”. Таким образом появляется возможность в траекториях отделить упоминания самого человека от упоминания связанного с ним устойчивого словосочетания.

Для более точного работы будущей модели, учитываются имена, извлеченные из первых двух предложений и только при условии, что

	Количество траекторий	Количество статей	Средняя длина траектории
Lenta.ru	450	31645	69.4
Дождь	405	14136	34.9
Meduza	63	1618	25.7
РИА Новости	189	37058	196.1
Суммарно	1113	84457	75.9

Таблица 1. Статистика по разным СМИ

в этом же предложении нет других упоминаний имен. Это практически гарантирует то, что главным фигурантом новостной статьи будет именно тот человек, имя которого извлечено.

В дальнейшем для работы модели используется текст нескольких первых предложений, если в них не упоминается какая-либо другая личность. Суммарная длина этих предложений не должна превышать 200 слов (в среднем это 12 предложений). Это обоснованно тем, что суть новости обычно укладывается в эти 200 слов. Последующий текст чаще всего является уточнением или справкой о каких-то событиях или организациях. Также к этим предложениям добавляются те, в которых упоминается извлеченная в начале личность, если такие предложения есть в дальнейшем тексте.

Итоговый датасет представляет собой набор новостных статей сгруппированных по траекториям. Представленные в датасете новостные интернет-издания: Lenta.ru[15], Дождь[16], Meduza[17], РИА Новости[18]. Подробная статистика по данным представлена в Таблице 1.

2. Базовая модель

Базовая модель создана на основе модели **RMN**, представленной в статье Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships[8]

Для программной реализации используется библиотека TensorFlow для языка программирования Python, которая предназначена для решения задач построения и тренировки нейронных сетей.

2.1. Описание модели

RMN решает следующие формализованные задачи.

1. Построить матрицу R размерности $K \times dim$, где K – задаваемое количество дескрипторов, dim – размерность векторного представления слов. R должна состоять из строк близким к векторным представлениям слов, которые и будут дескрипторами модели.

2. Сопоставить каждой статье с номером t из траектории веса дескрипторов (распределение на дескрипторах), представленные вектором d_t размерности K.

Модель получает на вход векторное представление $\hat{u}_t$ новостной статьи

$$\hat{u}_t = \frac{1}{|\hat{S}_t|} \sum_{w \in \hat{S}_t} u_w \tag{1}$$

где t — номер текущей статьи в траектории, S_t — множество всех слов траектории, $\hat{S}_t$ — множество слов w текущей статьи, в которое каждое слово попадает с вероятностью p — параметр Word Dropout[12], u_w — векторное представление слова.

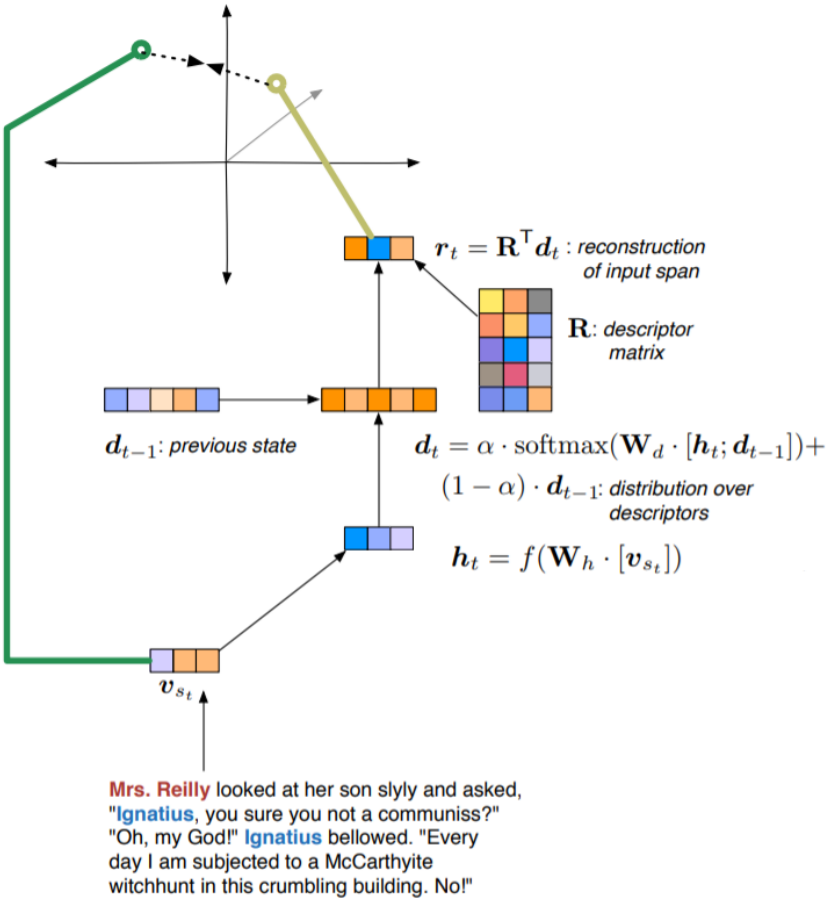

Рис. 1. Схематичное изображение базовой модели из статьи[8]

Далее модель вычисляет вектор скрытого состояния

$$h_t = \mathrm{ReLU}(W_h \cdot u_t) \tag{2}$$

где $\mathrm{ReLU}(x) = \max(0; x)$ — функция активации, W_h — матрица весов слоя.

Затем с помощью рекуррентного слоя RNN с использованием результата предыдущего шага вычисляется текущее распределение на дескрипторах d_t

$$\begin{aligned} d_t &= \alpha \cdot \mathrm{softmax}(W_d \cdot [h_t; d_{t-1}]) + (1-\alpha) \cdot d_{t-1} \quad t > 1 \\ d_1 &= \mathrm{softmax}(W_d \cdot [h_1; \vec{0}]) \end{aligned} \tag{3}$$

где $\alpha \in (0, 1]$ — параметр сглаживания,

$$\text{softmax}(X)_i = \frac{\exp(x_i)}{\sum\limits_{x_j \in X} \exp(x_j)} \tag{4}$$

Таким образом сумма всех компонент дескриптора равна единице.

Далее модель вычисляет вектор-реконструктор

$$r_t = R^T d_t \tag{5}$$

Матрица R нормированная и обучаемая. Задача вектора-реконструктора — приближать начальный вектор $u_t = \frac{1}{|S_t|} \sum\limits_{w \in S_t} u_w$ новостной статьи, который был вычислен без использования word dropout.

Таким образом матрица R состоит из строк, которые можно интерпретировать как векторные представления слов. Эти слова можно выявить, найдя ближайшие к строкам матрицы векторные представления слов используя косинусное расстояние. Вектор d_t отражает на сколько каждое из этих слов описывает исходную новостную статью. За счет параметра α обеспечивается гладкость распределения на дескрипторах во времени

Функция ошибки

$$L(\Theta) = J(\Theta) + \lambda X(\Theta) \tag{6}$$

состоит из двух слагаемых.

Первое слагаемое

$$J(\Theta) = \sum_{n \in N} \max(0; 1 - r_t \cdot u_t + r_t \cdot u_n) \tag{7}$$

минимизирует косинусное расстояние между вектором новостной статьи и вектором-реконструктором. Также эта функция максимизирует это расстояние между вектором-реконструктором и случайно

выбранными N векторами u_n (negative sampling[13]). Таким образом модель принуждается обучать более уникальные дескрипторы для каждой статьи.

Второе слагаемое

$$X(\Theta) = ||RR^T - I|| \tag{8}$$

отвечает за ортогональность матрицы дескрипторов R. Это означает, что вектора дескрипторов должны быть как можно более отдаленными друг от друга в смысле косинусного расстояния. λ – параметр ортогональности.

В базовой конфигурации используется значение Word Dropout $p = 0.75$, $\alpha = 0.5$, количество случайных векторов для Negative Sampling N = 50, коэффициент скорости обучения lr = 10e-3, параметр ортогональности λ = 10e-4. Оптимизация модели осуществляется методом Adam[14].

2.2. Достоинства и недостатки

Достоинством модели является то, что она не только выделяет тематики траекторий во времени, но и находит дескрипторы, которые должным образом кластеризируют блоки траекторий. Более того, эта модель не нуждается в обучающем множестве.

Рассмотрим примеры полученных дескрипторов (Таблица 1). Данные дескрипторы в большинстве своем отсылают к интуитивно понятным темам (стагфляция — экономические показатели, переизбраться — выборы, мессенджер — интернет), что несомненно является хорошим результатом.

Несмотря на свои достоинства, модель имеет недостатки, связанные с ее архитектурой.

Задача параметра α - сглаживать траекторию, то есть обеспе-

демократический	стагфляция	незастроенный	невиновный
правоотношение	пленум	педагогический	выплачивать
подражательница	очевидец	печалиться	переизбраться
бомбардирование	визит	импортный	мессенджер
самофинансирование	акционер	чемпионат	начальник

Таблица 2. Примеры дескрипторов

чивать незначительное отклонение текущего распределения на дескрипторах от предыдущего. Это обеспечивает выделение в качестве дескрипторов более долгосрочных тематик. Данный способ слишком "жестко" задает . К тому же, есть все основания полагать, что функция ошибки будет меньше при α близком к 1, но в этом случае распределение на дескрипторах будет недостаточно гладким. Возникает желание сконструировать такую сеть, в которой уменьшение значения функции ошибки будет однозначно отражать то, что модель лучше справляется с поставленной задачей.

Рекуррентный слой RNN является устаревшим и имеет ряд недостатков, которые будут рассмотрены далее. К настоящему времени существуют архитектуры, которые решают проблемы RNN и работают сравнительно лучше.

3. Модернизация базовой модели

В данной главе описываются изменения базовой модели

3.1. Изменение рекуррентного слоя

Рекуррентный слой RNN имеет ряд недостатков, к примеру затухающие и "взрывающиеся" градиенты, проблемы с обработкой долговременных зависимостей.

Рекуррентные сети GRU[21] и LSTM[20] используют систему юнитов и вентилей, которая обеспечивает более плавное распространение градиента.

Более того, сеть LSTM имеет два внутренних состояния. Одно из них отвечает за краткосрочную память, другое — за долговременную. Благодаря этому LSTM эффективно решает задачи, в которых возникает необходимость обрабатывать долговременные зависимости.

Формула распределения на дескрипторах без использования рекуррентного слоя:

$$\begin{aligned} d_t &= \alpha \cdot \text{softmax}(W_d \cdot h_t) + (1 - \alpha) \cdot d_{t-1} \quad t > 1 \\ d_1 &= \text{softmax}(W_d \cdot h_1) \end{aligned} \tag{9}$$

Для сравнения работы различных вариантов модели используются следующие конфигурации сети.

Векторное представление (embeddings) слов:

1. Word2vec обученный на датасете Russian National Corpus, размерность 300
2. FastText обученный на текстах Википедии и Lenta.ru, размерность 300
3. FastText обученный на текстах Twitter, размерность 100

embedding	alpha	RNN		GRU		LSTM	
		error	α	error	α	error	α
word2vec	0.5	0.7384		0.7396		0.7575	
	trained	**0.7269**	0.93	0.7287	0.93	0.7306	0.99
fastText wiki+lenta	0.5	0.7107		0.7133		0.7292	
	trained	**0.7002**	0.91	0.7013	0.91	0.7027	0.99
fastText twitter	0.5	0.7470		0.7476		0.7651	
	trained	**0.7345**	0.9	0.7352	0.9	0.7371	0.99

Таблица 3. Результаты работы модели с различными рекуррентными слоями

Рекуррентный слой:

1. RNN
2. GRU
3. LSTM

Параметр сглаживания α:

1. 0.5
2. Обучаемый параметр (trained)

При анализе работы моделей с различными рекуррентными слоями стоит учитывать что более сложные рекуррентные слои требуют больше эпох для обучения, поэтому при фиксированном количестве эпох значение функции ошибки вероятно будет выше у сложного слоя, при условии что этот слой не улучшает работу модели.

Из полученных данных (Таблица 3) можно сделать следующие выводы:

1) Улучшенные рекуррентные слои не улучшают работу модели. При обучении значение α приближается к 1, то есть исходный вектор намного лучше приближается за счет текущего вектора даже несмотря на word dropout 0.75.

2) Чем сложнее рекуррентный слой, тем больше α. Это можно объяснить то, что необходимая для работы модели информация лучше передается через внутреннее состояние сети чем через смешение распределений текущего и предыдущего дескриптора.

3) Значение функции ошибки значительно зависит от способа векторного представления слов.

Главная задача параметра α — сглаживать траекторию, то есть обеспечивать незначительное отклонение текущего распределения на дескрипторах от предыдущего. Это обеспечивает выделение в качестве дескрипторов более долгосрочных тематик. Но исходя из полученных результатов, можно выделить проблему касающуюся того, что "поведение" этого параметра желательно как-то зафиксировать, но при этом иметь больше свободы при обучении сети. Также возникает желание сконструировать такую сеть, в которой уменьшение значения функции ошибки будет однозначно отражать то, что модель лучше справляется с поставленной задачей.

3.2. Изменение функции ошибки

Логично переместить роль параметра α на функцию ошибки. Для этого изменим в исходной функции(6) первое слагаемое (7). Заменим вектор u_t, являющийся векторным представлением статьи на $\hat{u}_t$

$$\begin{aligned} &\hat{u}_t = \beta \cdot u_t + (1-\beta) \cdot \hat{u}_{t-1} \quad t > 1 \\ &\hat{u}_1 = u_1 \end{aligned} \tag{10}$$

Таким образом за гладкость траекторий отвечает параметр β, поэтому теперь функция ошибки отражает то, какой результат от нее требуется. Это означает, что мы можем обучать параметр не переживая, что реальный результат модели от этого ухудшится.

embedding	alpha	RNN		GRU		LSTM	
		error	α	error	α	error	α
word2vec	1	0.7803		0.7831		**0.7673**	
	trained	0.7712	0.61	0.7735	0.61	0.7727	0.98
fastText wiki+lenta	1	0.7520		0.7543		**0.7439**	
	trained	0.7439	0.45	0.7445	0.45	0.7444	0.97
fastText twitter	1	0.789		0.79		**0.7784**	
	trained	0.7793	0.44	0.7794	0.44	0.7785	0.98

Таблица 4. Результаты работы модели

Зафиксируем $\beta = 0.5$

Используем ту же сеть конфигураций что и в предыдущем разделе, в котором анализировалась работу предшествующей модели (Таблица 3). Только теперь зафиксируем $\alpha = 1$, при этом значении выходной вектор рекуррентного слоя не смешивается с вектором предыдущего распределения на дескрипторах. Это необходимо для того, чтобы оценить, осталась ли необходимость использовать жесткое смешение весов дескрипторов для достижения гладкости.

Из полученных результатов (Таблица 4) можно сделать следующий важный вывод. При использовании рекуррентного слоя LSTM получены достаточно хорошие результаты, особенно учитывая то, что это самая долгообучаемая сеть из представленных. Более того, эти результаты были лучше тогда, когда параметр α был зафиксирован и равнялся единице, то есть распределение предыдущего дескриптора не смешивалось с текущим. Это означает, что мы можем отказаться от этого компонента, который ранее отвечал за гладкость траектории. Внутренних состояний слоя LSTM оказалось достаточно, чтобы должным образом приблизить вектор $\hat{u}_t$.

Таким образом оптимальная конфигурация модели — LSTM без использования жесткого смешивания (эквивалентно $\alpha = 1$).

$$d_t = softmax(\text{LSTM}(h_t; d_{t-1})) \tag{11}$$

В дальнейшем используется векторное представление слов fastText wiki+lenta.

3.3. Дополнительные эксперименты

В данном разделе приведены дополнительные эксперименты с моделью. Происходит подбор гиперпараметров и дополнительно исследуются некоторые особенности модели.

3.3.1. Word Dropout

В конфигурации базовой модели используется параметр Word Dropout = 0.75. Такое высокое значение обосновывается тем, что это обеспечивает стабильность модели. На практике данный параметр вынуждает работать рекуррентный слой. Так как согласно архитектуре сети, базовый вариант модели должен приближать исходный вектор статьи, для модели с низким значением Word Dropout не возникает необходимость активировать рекуррентный слой — входной вектор сети в достаточной мере отражает вектор исходной статьи. Подтверждение этих слов в какой-то мере можно увидеть в Таблице 3. Модели, которые могли обучать значение альфа и в которых использовались рекуррентные слои, недостаточно хорошо отслеживающие долговременные зависимости, обучили данный параметр не слишком близко к единице. Это означает что при высоком Word Dropout жесткое смешение весов дескрипторов необходимо (хоть и с меньшим коэффициентом), даже несмотря на то, что предыдущее значение весов дескрипторов никак не используется в функции ошибки.

В модернизированной модели не возникает необходимости форсировать активацию рекуррентного слоя. Попробуем сравнить модели с различными значениями Word Dropout и оценить их на данных, полученных без использования Word Dropout (значения агрегированы

	0.75	0.5	0.25	0
mean error	0.7531	0.7517	0.7501	0.7497
std error	0.01845	0.01904	0.02380	0.02478

Таблица 5. Результаты работы модели

	without	20	30	40	50
mean error	0.7517	0.7490	0.7478	0.7474	0.7483

Таблица 6. Результаты работы модели

по результатам работы трех запусков моделей).

Согласно Таблице 5 средняя ошибка, посчитанная на "чистых" данных меньше при небольших значениях Word Dropout, но при приближении к нулю скорость падения ошибки падает. Противоположную ситуацию можно наблюдать со значениями стандартного отклонения.

Исходя из полученных результатов оптимальным выглядит использование значение параметра Word Dropout $p = 0.5$, так как при сравнительно небольшом увеличении стандартного отклонения, среднее значение ошибки уменьшается.

3.3.2. Скрытый слой

Как было показано ранее, LSTM улучшает работу модели (Таблица 4), но так как выходной вектор представляет собой произведение сигмоиды и гиперболического тангенса, значение каждого элемента выходного вектора лежит в промежутке (-1, 1). Веса дескрипторов в модифицированной модели вычисляются с помощью softmax(4). Рассмотрим крайний случай когда значение одного элемента вектора X размерности 20 равняется 1, а остальных -1.

$$\text{softmax}(X)_{x_i=1} = \frac{\exp(1)}{\exp(1) + 19 \cdot \exp(-1)} \approx 0.28 \tag{12}$$

Таким образом максимальное значение веса не превышает 0.28. Эта деталь сильно ограничивает возможности модели. Возможным решением является домножение выходного вектора LSTM на какую-нибудь константу, но стоит учитывать, что и сигмоида, и гиперболический тангенс слишком чувствительны к шуму входных значений около нуля, а домножение на константу больше единицы лишь увеличит влияние шума на результат. Хорошее решение в данном случае — добавить линейный слой между LSTM и softmax. Тогда формула распределения на дескрипторах будет выглядеть следующим образом

$$d_t = softmax(W_{LSTM} \cdot \text{LSTM}(h_t; d_{t-1})) \tag{13}$$

Скрытый слой дает возможность выбрать размерность h_{LSTM} выходного вектора LSTM, так как теперь количеству дескрипторов должна соответствовать первая размерность матрицы W_{LSTM}. Согласно проведенному эксперименту (Таблица 6), скрытый слой действительно улучшает работу модели, при этом его размерность не сильно влияет на значение ошибки. В дальнейшем будем использовать размерность скрытого слоя 40.

3.3.3. Коэффициент скорости обучения

В стандартной конфигурации используется коэффициент скорости обучения lr = 10e-3. У использования фиксированного значения коэффициента обучения есть несколько недостатков. При больших значениях модель рискует в начале обучения переобучиться на данных, которые на которых производиться первые шаги обучения, так как начальные значения градиента обычно очень высокие. Более того, при высоких значениях градиента может "перепрыгивать" значения минимума. При слишком маленьких значениях скорости обучения, модель рискует застревать в локальных минимумах. Оптимальное решение — использовать динамический коэффициент обучения,

	fixed lr	dynamic lr
mean error	0.7474	0.7462
std error	0.01896	0.01648

Таблица 7. Результаты работы модели

который зависит от номера шага обучения. Вариант функции, вычисляющий коэффициент скорости обучения в зависимости от шага приведен в статье[22].

$$lr = lr\_coef \cdot \min(step^{-0.5}; step \cdot warmup\_steps^{-1.5}) \tag{14}$$

Это соответствует линейному увеличению скорости обучения для первых этапов обучения $warmup\_steps$, и уменьшению его после этого пропорционально обратному квадратному корню от номера шага. Далее используется $warmup\_steps = 500$ и $lr\_coef = 0.03$.

Рассмотрим результаты моделей с разными конфигурациями скорости обучения (значения агрегированы по результатам трех запусков моделей). Как можно увидеть в Таблице 7, среднее значение и стандартное отклонение ошибки меньше при динамическом коэффициенте обучения. Это означает, что модель работает стабильнее и точнее при динамическом коэффициенте обучения.

3.3.4. Negative Sampling

Интересной деталью модели является Negative Sampling[13]. Согласно первому слагаемому функции ошибки(7), вектор-реконструктор должен в равной степени быть близким к вектору, который он приближает(u_t или $\hat{u}_t$ в зависимости от модели) и далеким от N случайных векторов из обучающей выборки. Данная деталь не кажется интуитивно понятной. Проведем эксперимент с функцией ошибки. Изменим исходную функцию ошибки модернизированной модели так, чтобы вес расстояния между вектором-реконструктором и случайными векто-

пятидесятилетие	фертильность	переадресовка	извечный
арестовать	февраль	говорить	субсидия
откровенничать	следователь	мальчишеский	выдвиженец
принадлежащий	боец	рефинансировать	площадь
транспортировочный	матчевый	благодарность	организатор

Таблица 8. Примеры дескрипторов

рами обучающей выборки был меньше.

$$\tilde{J}(\Theta) = \sum_{n \in N} \max(0; 1 - r_t \cdot u_t + 0.9(r_t \cdot u_n)) \tag{15}$$

При оценивании работы моделей с разными функциями ошибки очевидно нельзя смотреть на значения этих самых функций. Рассмотрим дескрипторы измененной модели (Таблица 8). Несмотря на то, что некоторым дескрипторам все еще можно поставить в соответствие определенную тематику (выдвиженец — выборы), многие из полученных дескрипторов представляют более фоновые слова, подобрать в соответствие тему которым практически невозможно (говорить, принадлежащий, и так далее).

Модель с измененной функцией ошибки в качестве дескрипторов выделяет более общие, неинформативные слова. Таким образом есть все основания полагать, что именно Negative Sampling способствует выделению дескрипторов, должным образом отражающих суть траекторий.

4. Сравнение моделей

Оценивание результатов работы моделей — отдельная сложная задача. Использовать какие-либо формальные функционалы качества не представляется возможным в связи с тем, что трудно определить требования к результатам работы модели.

Для сравнения результатов базовой модели и модели, полученной в данной работы был создан telegram-бот. Пользователю для сравнения поступает изображение с графическими результатами работы базовой и итоговой модели для одной и той же траектории. На графике для каждой модели изображены блоки тематик траектории, определяемые как несколько статей, идущих подряд во времени, у которых дескриптор с максимальным весом совпадает.

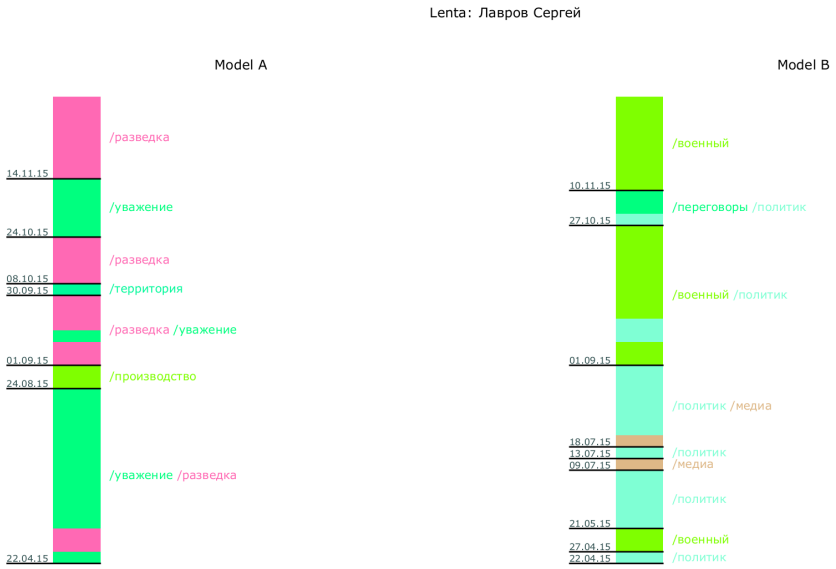

Рис. 2. Пример изображения

Выводы

Базовая модель достаточно качественно работает с данными СМИ, а также выделяет интерпретируемые дескрипторы благодаря использованию Negative Sampling.

У данной модели есть недостаток, связанный с тем, что функция ошибки не соответствует требуемому от модели результату. Из-за этого возникает необходимость использовать детерминированное смешение внутреннего состояния сети с предыдущим значением вектора весов дескрипторов(3). К тому же данная функция ошибки практически сводит на нет преимущества рекуррентного слоя, из-за чего улучшенные рекуррентные слои практически не влияют на результаты модели (Таблица 3). Новая функция ошибки решает эти проблемы. Благодаря ей при использовании улучшенного рекуррентного слоя LSTM результаты модели улучшаются и отпадает какая-либо необходимость в жестком, детерминированном смешении.

Добавление скрытого слоя между LSTM и softmax улучшает работу модели. Динамический коэффициент обучения также положительно влияет на работу модели, уменьшая среднее значение и стандартное отклонение ошибки.

Эксперимент поставленный с помощью telegram-бота доказал что-то там.

Заключение

В ходе выполнения данной работы были достигнуты следующие результаты:

1. При помощи автоматизированного обхода новостных сайтов СМИ, а также последующей предобработкой полученного материала
2. Реализована базовая модель, решающая задачи выделения дескрипторов и сопоставляющая каждой траектории
3. Модернизация модели
4. Анализ полученных результатов

Список литературы

[1] Nallapati, Ramesh, Ao Feng, Fuchun Peng and James Allan. “Event threading within news topics.” CIKM (2004).

[2] Mei, Qiaozhu and ChengXiang Zhai. “Discovering evolutionary theme patterns from text: an exploration of temporal text mining.” KDD (2005).

[3] Beagle, Donald, (2014). Digital Humanities in the Research Commons: Precedents & Prospects, Association of College & Research Libraries

[4] https://pypi.org/project/stylo/

[5] https://www.nltk.org/

[6] https://pymorphy2.readthedocs.io/en/latest/

[7] https://deeppavlov.ai/ [Электронный ресурс]: URL: `https://deeppavlov.ai/` (дата обращения: 20.05.2019).

[8] Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships / Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi [и др.]

[9] Amit Gruber, Yair Weiss, and Michal Rosen-Zvi. 2007. Hidden topic markov models. In Proceedings of Artificial Intelligence and Statistics

[10] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. / Latent dirichlet allocation. Journal of Machine Learning Research, 3

[11] Jonathan Chang, Jordan Boyd-Graber, and David M Blei. 2009a. / Connections between the lines: augmenting social networks with text. In Knowledge Discovery and Data Mining

[12] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daume III. 2015. / Deep unordered composition rivals syntactic methods for text classification. In Proceedings of the Association for Computational Linguistics.

[13] Richard Socher, Quoc V Le, Christopher D Manning, and Andrew Y Ng. 2014. / Grounded compositional semantics for finding and describing images with sentences. Transactions of the Association for Computational Linguistics.

[14] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In Proceedings of the International Conference on Learning Representations.

[15] Lenta.ru [Электронный ресурс]: URL: `https://lenta.ru/` (дата обращения:).

[16] Tvrain [Электронный ресурс]: URL: `https://tvrain.ru/` (дата обращения:).

[17] Meduza [Электронный ресурс]: URL: `https://meduza.io/` (дата обращения:).

[18] РИА Новости [Электронный ресурс]: URL: `https://ria.ru/` (дата обращения:).

[19] scrapy [Электронный ресурс]: URL: `https://scrapy.org/` (дата обращения:).

[20] Sepp Hochreiter; Jürgen Schmidhuber (1997). “Long short-term memory”

[21] Chung, Junyoung, Çaglar Gülçehre, Kyunghyun Cho and Yoshua Bengio. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.” CoRR abs/1412.3555 (2014)

[22] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. "Attention Is All You Need." NIPS (2017).