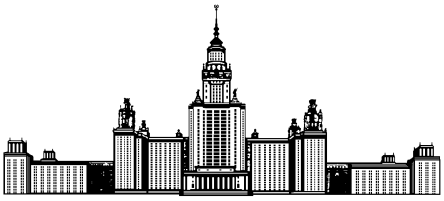

Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Полыковский Даниил Александрович

Механизмы внимания в нейронных сетях

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
к.ф-м.н.,
Д. П. Ветров

Москва, 2017

Содержание

Аннотация

В работе рассматривается применение механизмов внимания к задаче согласования слов на русском языке. Отличительная особенность предложенной модели — эффективная эксплуатация соответствия генерируемых и нормализованных слов. Вычислительные эксперименты показывают преимущество предложенной модели по сравнению с charRNN. Также в работе рассматривается применение предложенной модели к вопросно-ответным системам. Комбинация пословной модели, работающей с нормализованными текстами, и модели, согласовывающей текст, показывает лучшие результаты по сравнению с посимвольной моделью.

1 Введение

Seq2Seq модели — наиболее часто используемая архитектура в машинном переводе и нейросетевых вопросно-ответных системах. Наибольшее количество памяти в таких моделях расходуется на хранение матрицы представлений, содержащей представление каждого слова из словаря. Для пословной генерации согласованного ответа требуется иметь наряду со стандартной формой слова еще и все его словоформы. В некоторых языках у слов имеется лишь небольшое количество словоформ (например, единственное и множественное число). Тем не менее, в таких языках как русский, у многих слов присутствует большое число словоформ, получающихся изменением рода, числа, падежа и времени.

Модели со словарями, достаточно полно покрывающими множество всех словоформ, превышают разумные ограничения как по времени, так и по памяти. Во многих работах для обхода этой проблемы переходят к посимвольным моделям. В таких моделях размер словаря соответствует размеру используемого алфавита. Посимвольная генерация позволяет избегать хранения словоформ, однако из-за увеличения длины последовательности в несколько раз, модель быстро забывает начало предложения. Альтернативное решение — хранение в словаре лишь стандартной формы слов. Такая модель будет генерировать несогласованный текст и не может быть использована в рабочей системе.

В данной работе предлагается компромисс между посимвольной и пословной моделями. Предложенная модель состоит из двух компонент: генератора и согласователя. Генератор последовательно обрабатывает слова из нормализованного вопроса и генерирует нормализованный ответ. Согласователь преобразует нормализованный текст в согласованный, используя при этом слова вопроса в качестве контекста.

2 Описание используемых методов

2.1 LSTM сети

Long-short term memory (LSTM) [1] — популярная архитектура рекуррентной сети. Основная идея этой архитектуры — выделение ячейки памяти, ответственной за хранение информации, полученной в предыдущие моменты времени. Для управления этой ячейкой выделяются три шлюза: входной $\mathbf{i}^t$ (контролирует поступление данных в память), исходящий $\mathbf{o}^t$ (контролирует распространение данных из памяти) и шлюз памяти $\mathbf{f}^t$ (ответственен за сохранение/забывание предыдущего состояния ячейки).

Функция ячейки памяти схожа с состояниями в детерминированном конченом автомате, однако в данном случае состояние «распределенное» и позволяет работать с бесконечным числом состояний. LSTM сети пользуются большой популярностью, так как они способны обнаруживать долгосрочные зависимости в данных.

Следующие формулы [2] описывают процесс вычисления одной итерации использованной в работе архитектуры LSTM сети. В формулах поэлементное произведение обозначается символом $\odot$.

$$
\begin{aligned}
\mathbf{z}^t &= g(\mathbf{W}_z\mathbf{x}^t + \mathbf{R}_z\mathbf{y}^{t-1} + \mathbf{b}_z) && \textit{вход} \\
\mathbf{i}^t &= \sigma(\mathbf{W}_i\mathbf{x}^t + \mathbf{R}_i\mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i) && \textit{входной шлюз} \\
\mathbf{f}^t &= \sigma(\mathbf{W}_f\mathbf{x}^t + \mathbf{R}_f\mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f) && \textit{шлюз памяти} \\
\mathbf{c}^t &= \mathbf{i}^t \odot \mathbf{z}^t + \mathbf{f}^t \odot \mathbf{c}^{t-1} && \textit{состояние ячейки} \\
\mathbf{o}^t &= \sigma(\mathbf{W}_o\mathbf{x}^t + \mathbf{R}_o\mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^t + \mathbf{b}_o) && \textit{исходящий шлюз} \\
\mathbf{y}^t &= \mathbf{o}^t \odot h(\mathbf{c}^t) && \textit{выход}
\end{aligned}
$$

LSTM сети обучаются при помощи алгоритма обратного распространения ошибки сквозь время (backpropagation through time), идея которого состоит в разворачивании графа вычислений во времени. Из LSTM сетей можно выстраивать многослойные нейронные сети, передавая выходную последовательность очередного слоя на вход следующему.

2.2 Seq2Seq модели

Одной из наиболее популярных архитектур в машинном переводе являются модели sequence to sequence (Seq2Seq) [3]. Такие модели состоят из двух рекуррентных сетей: кодировщика и декодировщика. Кодировщик строит представление входной последовательности слов. Далее полученное представление (последние выход и значение ячейки сети) копируются в декодировщик. По полученному представлению декодировщик пытается восстановить целевую последовательность слов. В задачах машинного перевода входной и выходной последовательностями являются предложения на разных языках. В вопросно-ответных и диалоговых системах — вопрос и ответ.

Для преобразования слов во входные вектора используется так называемая матрица представлений (embedding matrix). Количество строк этой матрицы равно размеру словаря, а число столбцов — размеру ячейки LSTM. Каждая строка соответствует векторному представлению соответствующего слова. Каждое слово перед подачей на вход LSTM сети заменяется на соответствующую строку матрицы представлений.

На вход декодировщику на первом такте подается специальный символ `<GO>`, затем на каждом такте подается сгенерированное в предыдущую итерацию слово. Генерация ответа продолжается до тех пор, пока не будет сгенерировано специальное слово – маркер конца строки `<EOL>` (end of line). Во время обучения в качестве сгенерированного символа на следующий такт передается целевой символ, а распределение на предсказанных символах передается в функцию потерь.

Во время предсказания требуется найти наиболее вероятное предложение с точки зрения модели. Сделать это напрямую невозможно, так как модель позволяет вычислять только наилучшее слово при фиксированных предыдущих. Компромиссным решением между жадным выбором слов и полным перебором является Beam Search. При использовании этого метода на каждой итерации выбирается небольшое количество лучших кандидатов, а остальные гипотезы отбрасываются.

После применения Beam Search сеть часто начинает отвечать при помощи наиболее часто встречающихся в выборке ответов: «да», «нет», «не знаю». Для борьбы с этим можно обучить две модели[4]: предсказывающую ответ по вопросу и вопрос

по ответу. Предложения, сгенерированные beam search первой модели, переранжируются согласно выпуклой комбинации логарифмов правдоподобий двух моделей: $\lambda \log P(A|Q) + (1 - \lambda) \log P(Q|A)$, где Q — вопрос, A — ответ, $\lambda \in [0, 1]$ — гиперпараметр. Теперь частотные ответы будут иметь низкую вероятность, так как по ним редко возможно восстановить вопрос. Для генерации длинных предложений к функции ранжирования добавляется штраф, поощрающий генерацию большого числа слов. Обычно в качестве такой фукнции выбирается $\gamma|A|$, где $|A|$ — число сгенерированных слов, а γ — гиперпараметр.

2.3 Механизмы внимания

Механизмы внимания — это подход в машинном обучении, заключающийся в выделении части входных данных (регионов изображений, фрагментов текста) для более детальной обработки.

Часто для решения задачи классификации изображений не требуется обрабатывать все пиксели изображения: например, в задаче классификации фон часто играет незначительную роль. Тем не менее, сверточные сети, являющиеся наиболее популярным методом решения такой задачи, тратят одинаковое количество вычислительных ресурсов на все части изображения. Так Spatial Transformer Networks [5] используют преобразования изображений (например, афинные) чтобы выделять наиболее важные области. Другой подход — введение агента, исследующего изображение [6]. Такой агент в каждый момент времени обрабатывает небольшую часть изображения, а также решает в какую позицию переместить фокус внимания. Механизмы внимания могут быть использованы для повышения производительности нейронных сетей.

Опишем наиболее распространенный подход к реализации внимания в нейронных сетях. Изначально выделяется множество векторов $\{h_i\}_{i=1}^K$, над которыми будет осуществляться внимание. Например, в задаче генерации заголовков изображений[7] можно преобразовать исходное сообщение при помощи нескольких сверточных слоев и использовать набор выходных векторов для каждого пикселя в качестве объекта внимания. На втором шаге часть сети генерируется ключ k, который будет отвечать за то, какие вектора из $\{h_i\}_{i=1}^K$ будут использованы. При помощи ключа каждому

вектору h_i сопоставляется вес w_i, который может быть интерпретирован как вероятность того, что внимание должно быть сконцентрировано именно на объекте h_i. Чаще всего эти веса получаются по формуле $w_i = \exp(d(k, h_j)) \Big/ \left[\sum_{j=1}^{K} \exp(d(k, h_i))\right]$. В качестве меры близости $d(k, h)$ обычно используется косинусный коэффициент: $d(k, h) = \frac{k^T h}{\|k\| \cdot \|h\|}$. Таким образом внимание фокусируется в основном на объектах схожих с ключами. Например, если в качестве ключа будет использован вектор, отвечающий за текстуру коры, то наибольшим весом будут обладать части изображения, на которых изображена кора деревьев.

На следующем шаге вычисляется вектор контекста $\widehat{h}$ — обобщенное представление объектов внимания с учетом ключа. Существует два подхода к его заданию: жесткое внимание (hard attention) и мягкое внимание (soft attention). При жестком внимании в качестве $\widehat{h}$ берется h_ξ, где $\xi \sim \text{Descrete}(w_1, w_2, \ldots, w_K)$. При обучении модели используется Монте-Карло оценка на градиент: такая оценка является несмещенной, однако на практике обладает большой дисперсией. Из-за этого модели с жестким вниманием приходится использовать различные методы снижения дисперсии, такие как REINFORCE[8]. При мягком внимании $\widehat{h}$ вычисляется как $\mathbb{E}_{i \sim \xi}[h_i] = \sum_{i=1}^{K} w_i h_i$. Такой подход позволяет обучать модель детерминистически end-to-end. В зависимости от весов внимания информация может как агрегироваться с наиболее важных частей данных, так и только с одного объекта. На сегодняшний день чаще используется именно механизм soft attention, поэтому далее будет рассматриваться только он [9][10][11].

Механизмы внимания также часто используются в нейросетевом машинном переводе. Проблемой обычных Seq2Seq моделей (см. раздел 2.2) является необходимость сжать всю информацию в векторе представления. Эта проблема становится особенно существенной при переводе длинных последовательностей. Было показано[9][10], что Seq2Seq с вниманием значительно улучшает качество работы на длинных последовательностях. В качестве объекта внимания в таких моделях используются выходы последнего слоя кодирующей части для каждого слова. В качестве ключа выбирается выход последнего слоя декодирующей части. Для генерации слов вектор контекста конкатенируется с ключом и пропускается через еще один рекуррентный слой.

Важное применение механизмы внимания находят в создании нейрокомпьютера[12][13] — нейронной сети, моделирующей структуру обычных компьютеров. Их отличительная особенность заключается в наличии у них памяти и контроллера, позволяющего осуществлять операции чтения и записи в нее. Также внимание используется в дифференцируемых версиях таких структур данных как список, стек, дек, очередь[14][15].

3 Обзор литературы

Ранее было предложено несколько моделей, работающих с текстом на разных уровнях: символах, n-граммах и словах. Бояновски и др. [16] предлагают использовать последовательность из двух рекуррентных сетей: работающей со словами, но обладающей маленьким словарем и с посимвольной моделью, использующей скрытое состояние первой сети в качестве дополнительного входа. Луонг и др. [11] предлагают генерировать и считывать слова, не входящие в словарь, при помощи посимвольной сети. Миколов и др. [17] предлагают использовать небольшое количество наиболее частотных слов, а остальные слова представлять как последовательность слогов. Йохансен и др. [18] предлагают схожий подход: в их модели все слова преобразуются в вектор представлений, который затем подается на вход LSTM сети. Выходная последовательность в такой модели генерируется посимвольно.

4 Описание предложенной модели (Concorde)

Модель для согласования слов получает на вход вопрос и нормализованный ответ и по этим данным генерирует согласованный ответ, содержащий такое же количество слов, как и нормализованный ответ. Это преобразование проходит в четыре этапа: генерация представления вопроса, генерация представления слов нормализованного ответа, преобразование представлений, генерация слов согласованного ответа (рис. 1).

Изначально при помощи посимвольной рекуррентной сети каждое слово вопроса преобразуется в вектор представления q_i. В этих векторах закодирована информация, которая может быть полезна для согласования слов в ответе. Предположительно этот вектор может кодировать такие признаки как время, род, число, падеж и так далее. Также этот вектор может быть полезен для переноса имен собственных из вопроса в ответ.

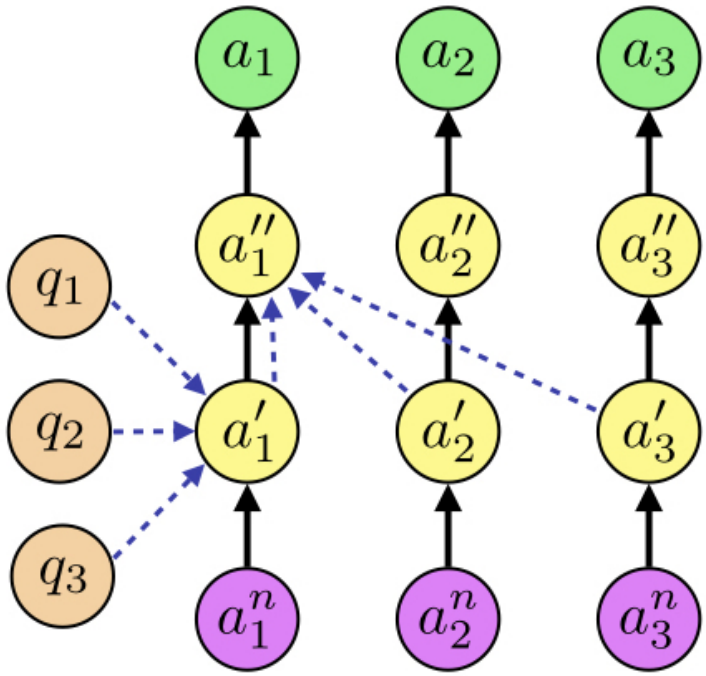

Рис. 1: Модель Concorde. q_i — представления слов вопроса, a^n_i — представления нормализованных слов, a_i — согласованные слова, a'_i и a''_i — модифицированные представления.

На втором шаге генерируется набор представлений — по одному представлению a^n_i на каждое слово нормализованного ответа. Далее этот набор модифицируется так, чтобы учитывать другие нормализованные слова и слова из вопроса. Обе эти модификации происходят по одинаковой схеме, описанной ниже. Основная ее идея — при помощи механизма внимания выбрать важные для согласования слова (сначала из вопроса, затем — из нормализованного ответа) и учесть полученную информацию в представлении.

Представления a_i^n не способны генерировать согласованный ответ, так как не используют информацию о других словах в нормализованном ответе, поэтому принятие решения каким словам стоит уделять больший вес критично в данной модели. Чтобы дать модели возможность уточнить веса внимания, модификация словами вопроса применяется раньше. Из нее модель получает важную информацию о времени и числе, что позволяет на следующем этапе более точно сгенерировать веса.

Опишем как происходит модификация векторами q_i. Чтобы объединить a_i^n с q_i, применяется механизм внимания над всеми модифицированными представлениями. Изначально вычисляются вектора $\widehat{q}_i = \sum_{k=1}^{K} \widehat{w}_{ik} q_i$, $\widehat{w}_{ik} = \frac{exp(w_{ik})}{\sum_{j=1}^{K} exp(w_{ij})}$, а веса вычисляются по формуле $w_{ij} = \cos(W a_i^n + b, q_j)$. Например, при преобразовании фразы «я завтра купить молоко» в «я завтра куплю молоко» для генерации слова «куплю» сеть будет выделять наибольший вес слову «завтра». Добавление смещения b позволяет выучивать глобальную направленность внимания: например, на артикли или слова, определяющие время, в котором формулируется предложение.

Модифицированное представление a'_i получается из старого представления a_i^n и вектора $\widehat{q}_i$ по следующей формуле: $a'_i = A \cdot (a_i^n || \widehat{q}_i) + d$, где символ $||$ означает конкатенацию векторов. Эта формула позволяет учесть полученные из механизма внимания данные, а также преобразовать представление в удобный для дальнейшей обработки вид. Такая модификация, вносит информацию о времени, числе и других лингвистических параметрах, необходимых для согласования.

Схема для модификации при помощи a'_i определяется по аналогии. Пусть $\widehat{a'}_i$ — полученный вектор внимания над a'_i. Тогда $a''_i = A' \cdot (a'_i || \widehat{a'}_i) + d'$. Такая модификация позволяет сети учесть другие слова, присутствующие в предложении.

После преобразования представлений вектора a''_i используются в качестве начального состояния декодирующей рекуррентной сети, которая по каждому представлению посимвольно генерирует одно слово. При генерации возможно использование внимания над исходными символами. Сгенерированный в результате этого шага набор слов является выходом модели. Модель обучается градиентным спуском так же, как это делается для обычных Seq2Seq моделей.

5 Вопросно-ответная система с Concorde

Модель для согласования слов Concorde может может быть использована в вопросно-ответных системах. Основная проблема Seq2Seq моделей для языков с богатой морфологией — необходимость хранить словоформы большинства слов. Для решения этой проблемы предлагается разбить задачу на две: генерацию нормализованного ответа и дальнейшее согласование сгенерированного текста. Работа на уровне слов с нормализованным словарем в несколько раз уменьшает объем словаря (примерно в 2.5 раза для русского языка: с 220402 до 86415 слов). Помимо выигрыша по памяти работа с меньшим словарем ускоряет обучение представлений слов, что снижает время сходимости в разы.

Генерируемые такой моделью тексты редко допускают ошибки в согласовании. В то же время получаемые тексты довольно точно отвечают на задаваемые вопросы.

6 Эксперименты

Во всех экспериментах буквы слов из согласованного ответа предсказываются в обратном порядке[3], так как это позволяет изначально сгенерировать окончание — наиболее вариабельную часть. Чтобы модель могла учитывать порядок слов, в начало каждого нормализованного слова дописывалась последовательность из специальных символов <x> и <.>. Длина этой последовательности соответствовала количеству слов в предложении. В такой последовательности содержался лишь один символ <x> — в позиции, соответствующей положению рассматриваемого слова. Например, последовательность <.><.><x><.> дописывалась в конец третьего слова предложения, состоящего из четырех символов. Чтобы ускорить обучение, контекст обрезался по 300 символам а согласуемое предложение обрезалось по 10 словам.

В качестве рекуррентных сетей использовались LSTM блоки. Размерность представления символа была равна 32, размеры скрытых слоев всех рекуррентных сетей равны 512 и состояли из двух слоев. Обучение проводилось при помощи метода Adam[19] с темпом обучения 0.0002, убывающим в два раза после каждых 50000 обновлений весов. Размер батча был равен 16. Модели обучались 200 прогонов по 1000

батчей. Чтобы не столкнуться с проблемой взрывающихся градиентов, элементы вектора градиента урезались по порогу[20] 100. В качестве алфавита модели использовались строчные и заглавные буквы русского и английского языка, а также цифры, таким образом задача расстановки пунктуации не рассматривалась.

Модель сравнивалась с посимвольным Seq2Seq с двумя слоями по 1024 нейрона и вниманием над выходами кодирующей части. Эта модель далее обозначается как charRNN. На вход такой модели подается строка, содержащая вопрос и ответ, разделенные специальной последовательностью.

6.1 Сравнение качества с charRNN

В этом разделе приводятся эксперименты на записях двух корпусов: OpenSubtitles* и Ответы Mail.ru. Примеры данных показаны в таблице 1.

OpenSubtitles	
Q	Я люблю тебя
nA	умолять ты не делать это
A	Умоляю тебя не делай этого
Ответы Mail.ru	
Q	спортивная статистика вещь обманчивая
nA	с статистика не поспорить но бывать и исключение
A	со статистикой не поспоришь но бывают и исключения

Таблица 1: Примеры использованных данных. Q — вопрос, nA — нормализованный ответ, A — согласованный ответ.

Корпус OpenSubtitles состоит из субтитров фильмов, переведенных на русский язык. В качестве контекста согласования (вопроса) и согласуемого предложения (ответ) использовались пары соседних предложений.

В качестве второго набора данных использовались записи с сайта otvet.mail.ru — службы вопросов и ответов компании Mail.Ru Group. Для работы было скачано 5

*http://opus.lingfil.uu.se/OpenSubtitles2016.php

миллионов вопросов и ответов. В качестве ответа, где это было возможно, брался ответ с пометкой «лучший». Вопросом считался заголовок темы — тело вопроса не учитывалось. Чтобы избежать потери большого количества информации о вопросе, оставлялись только те темы, в которых заголовок заканчивался на символ вопроса.

Для оценки качества были использованы четыре метрики: перплексия (perplexity), доля правильно предсказанных символов (character accuracy), доля правильно предсказанных слов (word accuracy) и доля правильно предсказанных предложений (sentence accuracy). Метрика word accuracy не считается для модели charRNN, так как модель не оперирует со словами как отдельными объектами.

Графики обучения показаны на рисунках 2 и 3. В таблице 2 показаны метрики качества для обоих корпусов.

		OpenSubtitles		**Ответы Mail.ru**	
		Concorde	**CharRNN**	**Concorde**	**CharRNN**
Перплексия	**train**	1.12	1.16	1.1	1.16
	test	1.10	1.17	1.1	1.15
Character accuracy	**train**	96.38%	95.21%	96.47%	95.05%
	test	96.62%	95.04 %	96.51%	95.15%
Word accuracy	**train**	81.5%	78.38%	80.38%	74.48%
	test	82.17%	77.91%	80.62%	74.85%
Sentence accuracy	**train**	46.91%	43.38%	42.03%	32.4%
	test	49.5%	45.35%	43.17%	33.07%

Таблица 2: Качество работы Concorde и charRNN. Перплексия минимизируется, точность максимизируется.

По всем метрикам модель Concorde продемонстрировала лучший результат по сравнению с charRNN. При этом отличие более заметно на корпусе Ответы Mail.ru, так как данные этого корпуса представляют собой именно вопросы и ответы, а не просто пары соседних предложений.

Предолженная модель обучается значительно быстрее charRNN. Это связано с тем, что архитектура сети позволяет без труда выучить копирования слова за первые

несколько эпох, а потом уже выучивать необходимые для согласования преобразования.

6.2 Существенность компонент

В данном разделе показывается существенность компонент предложенной модели. Были обучены модели с параметрами, аналогичными предыдущим, но без некоторых частей модели: без слов вопроса, без внимания над символами и без внимания над соседними словами. В таблице 3 показано качество таких моделей. Удаление любой из этих компонент приводит к ухудшению качества для данных Ответ Mail.ru. Для набора данных OpenSubtitles наблюдается лишь незначительное падение в качестве при удалении лингвистического контекста. Это связано с тем, что в субтитрах контекст не играет существенной роли для предсказания: например, почти все предложения написаны в настоящем времени.

	OpenSubtitles	**Ответы Mail.ru**
Полная модель	46.91%/49.5%	42.03%/43.17%
Без слов вопроса	46.89%/49.34%	38.86%/39.88%
Без внимания над символами	45.84%/48.76%	41.04%/42.09%
Без внимания над словами	25.8%/27.8%	26.17%/26.66%

Таблица 3: Качество работы Concorde после удаления различных компонент. Метрика sentence accuracy, результаты записаны в формате train/test.

6.3 Примеры работы

Рассмотрим примеры работы обученной сети. В левом столбце таблиц показан нормализованный ответ, а в правом — выход модели. Интересно, что при переходе к множественному числу модель не допустила ошибки в чередующейся согласной в корне.

Множественное число	
один заяц	Один заяц
два заяц	Два зайца
один боец	Один боец
два боец	Два бойца

Род	
один дом	Один дом
один тарелка	Одна тарелка
один растение	Одно растение

Числительные	
одна груша	Одна груша
два груша	Две груши
три груша	Три груши
четыре груша	Четыре груши
пять груша	Пять груш

Предлоги	
у дом	У дома
в дом	В доме
над дом	Над домом
к дом	К дому

Рассмотрим какие слова используются для согласования предложений на примере пары «Кто такая Элис?» и «девочка элиса жить в соседний подъезд» (рис. 4). Видно, что сеть разделила предложение на две части и согласовала их независимо. Например, для согласования слова «соседний» сеть посмотрела на слово «подъезд» и использовала его род. Также сеть использовала наличие рядом предлога «в», что позволило ей узнать падеж.

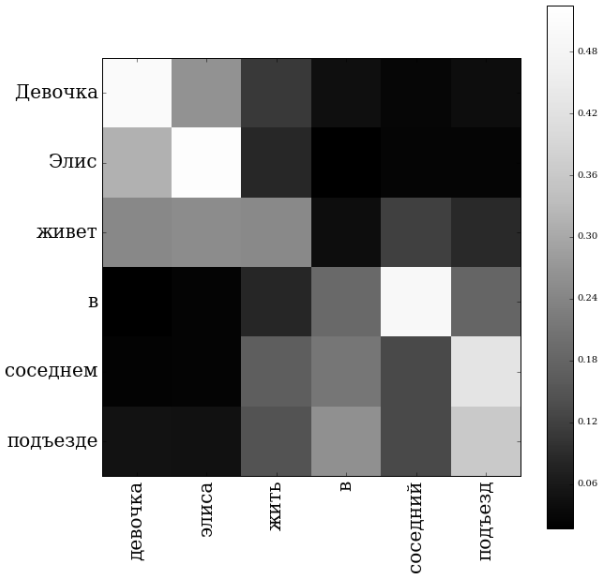

Рис. 4: Веса внимания, сгенерированные моделью.

6.4 Вопросно-ответная система с Concorde

Модель Concorde была применена для вопросно-ответной системы. Была обучена модель для пословной генерации нормализованного ответа. Ответ, сгенерированный моделью, согласовывался при помощи Concorde, обученной на данных Ответы Mail.ru.

Для сравнения такой модели и модели, обученной на символах, была использована асессорская оценка. Асессорам показывались вопросы из данных контроля и два варианта ответа в случайном порядке: сгенерированный посимвольной моделью и связкой пословной с Concorde.

По итогам эксперимента было размечено 682 вопроса. В 62.1% случаях асессоры выбрали предложенную модель, в 37.9% — посимвольную.

7 Обсуждение

Полученная модель может быть использована в связке не только с нейросетевыми алгоритмами, что делает её независимой компонентой в системе. Данные для рассматриваемой задачи могут генерироваться синтетически из любых текстов, что позволяет сколь угодно расширять обучающую выборку. Ожидается, что можно добиться прироста в качестве при обучении модели на большей выборке.

Модель Concorde может быть также использована для постобработки текстов. Например, при генерации новостного сообщения из различных источников можно запустить Concorde для согласования слов между предложениями, взятыми из разных источников.

Отличительная особенность модели — эффективная эксплуатация соответствия генерируемых слов и нормализованных слов. Это позволяет избежать нагрузки на вектор представления, как это происходит в посимвольной модели.

Предложенная модель допускает распараллеливание, так как вектора представлений этой модели могут генерироваться независимо для каждого слова. Также параллельно на последнем шаге могут генерироваться слова из представлений. При-

менение внимания эффективно вычисляется при помощи матричного умножения, которое так же быстро вычисляется на GPU.

8 Заключение

В работе предложена модель для согласования слов для языков с богатой морфологией. Построенная модель может быть использвана в большом числе приложений: построении сниппетов, генерации новостей и вопросно-ответных системах. Предложенный подход к вопросно-ответным системам представляет собой компромиссное решение между посимвольной генерацией текста и нормализованным словарем. Рассмотренная модель демонстрирует значимое преводсходство над посимвольными моделями.

В данной работе получены следующие результаты:

- Предложен метод для согласования нормализованных предложений, центральную роль в котором играют механизмы внимания;
- Было исследовано применение предложенной модели к нейросетевым вопросно-ответным системам;
- Была исследована существенность компонент предложенной системы.

Список литературы

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997.

[2] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. arXiv preprint arXiv:1503.04069, 2015.

[3] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Advances in neural information processing systems, pages 3104–3112, 2014.

[4] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. arXiv preprint arXiv:1510.03055, 2015.

[5] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In Advances in Neural Information Processing Systems, pages 2017–2025, 2015.

[6] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In Advances in neural information processing systems, pages 2204–2212, 2014.

[7] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In ICML, volume 14, pages 77–81, 2015.

[8] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3-4):229–256, 1992.

[9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473, 2014.

[10] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025, 2015.

[11] Minh-Thang Luong and Christopher D Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. arXiv preprint arXiv:1604.00788, 2016.

[12] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. arXiv preprint arXiv:1410.5401, 2014.

[13] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette,

Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. Nature, 538(7626):471–476, 2016.

[14] Edward Grefenstette, Karl Moritz Hermann, Mustafa Suleyman, and Phil Blunsom. Learning to transduce with unbounded memory. In Advances in Neural Information Processing Systems, pages 1828–1836, 2015.

[15] Armand Joulin and Tomas Mikolov. Inferring algorithmic patterns with stack-augmented recurrent nets. In Advances in neural information processing systems, pages 190–198, 2015.

[16] Piotr Bojanowski, Armand Joulin, and Tomas Mikolov. Alternative structures for character-level rnns. arXiv preprint arXiv:1511.06303, 2015.

[17] Tomáš Mikolov, Ilya Sutskever, Anoop Deoras, Hai-Son Le, Stefan Kombrink, and Jan Cernocky. Subword language modeling with neural networks. preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf), 2012.

[18] Alexander Rosenberg Johansen, Jonas Meinertz Hansen, Elias Khazen Obeid, Casper Kaae Sønderby, and Ole Winther. Neural machine translation with characters and hierarchical encoding. arXiv preprint arXiv:1610.06550, 2016.

[19] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[20] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. ICML (3), 28:1310–1318, 2013.

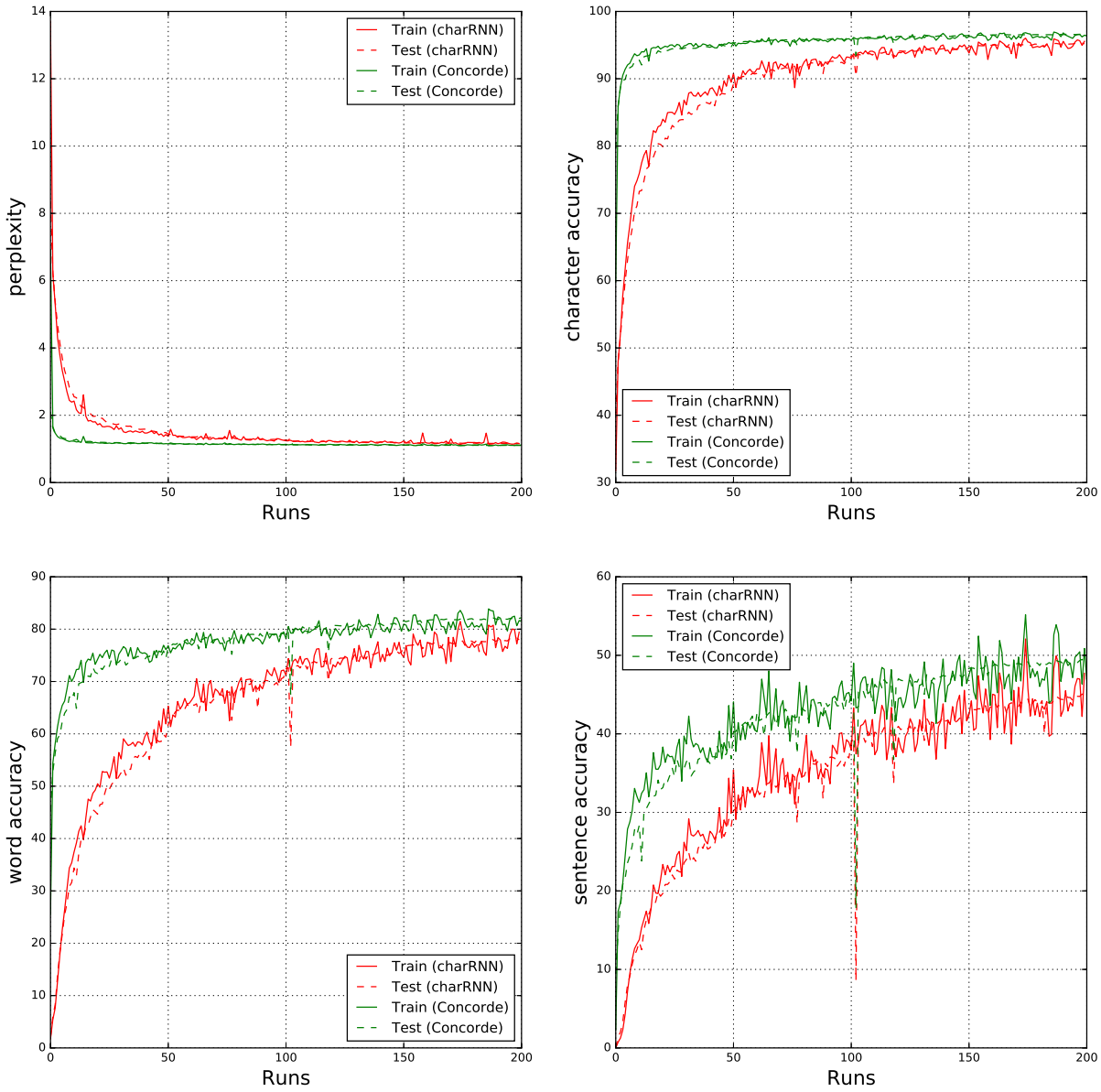

Рис. 2: Обучение Concorde и charRNN на данных OpenSubtitles

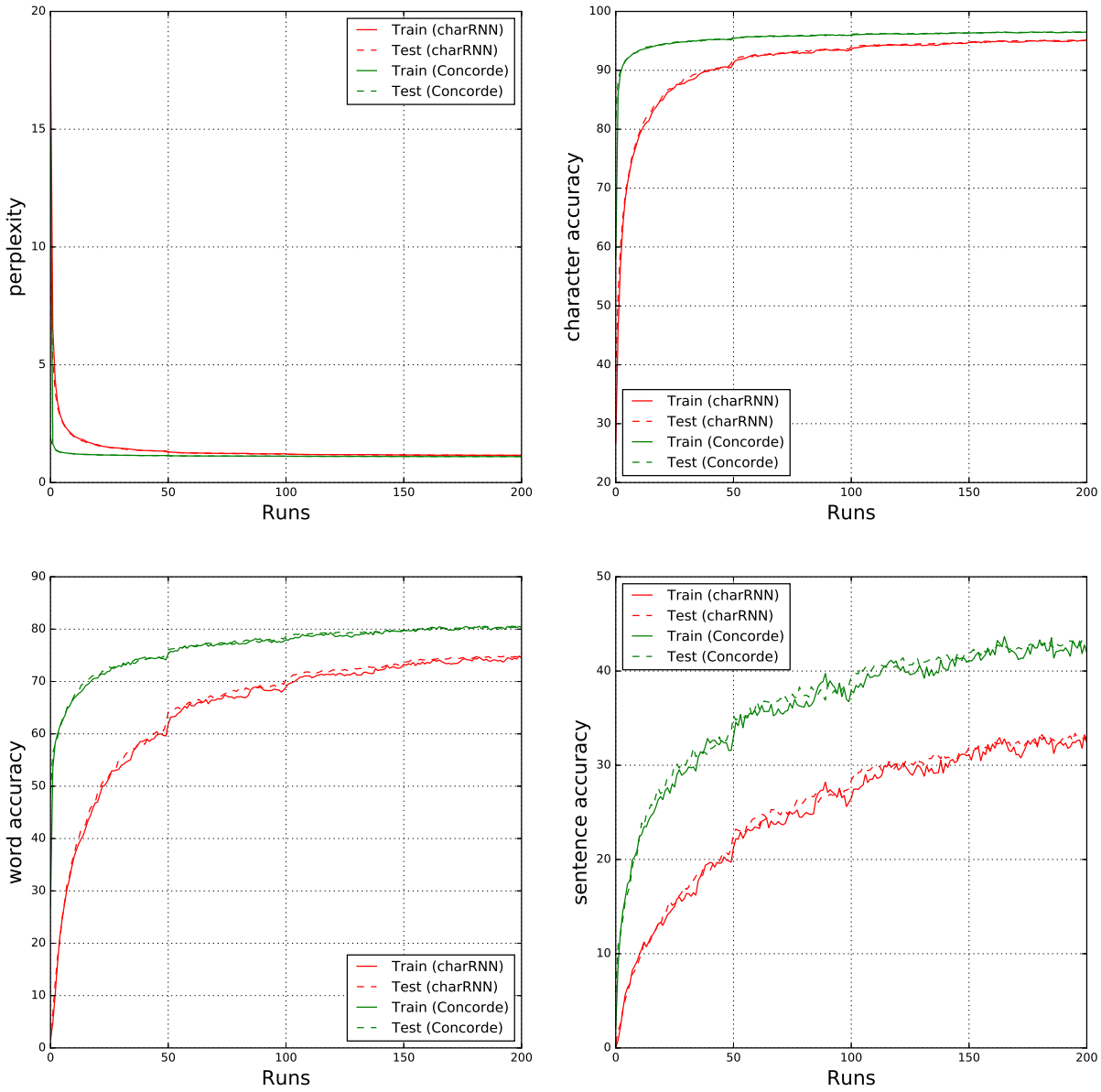

Рис. 3: Обучение Concorde и charRNN на данных Ответы Mail.ru