ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики
Им. А.Н.Тихонова НИУ ВШЭ

Подчезерцев Алексей Евгеньевич, группа БИВ172
Солодянкин Андрей Александрович, группа БИВ172

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ЭЛЕКТРОННОЙ СИСТЕМЫ ДЛЯ ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ СО ШКОЛЬНИКАМИ

Междисциплинарная курсовая работа
по направлению 09.03.01 Информатика и вычислительная техника
студентов образовательной программы бакалавриата
«Информатика и вычислительная техника»

Студент _____ А.Е. Подчезерцев

Студент _____ А.А. Солодянкин

Руководитель
Ассистент
_____ А.Ю. Ролич

Москва 2021 г.

Аннотация

A Software Tool for Automatic Generation of a Graph of Conversation Using a Drama Corpus

Abstract

Содержание

Введение

В последние годы технологии машинного обучения стали неотъемлемой частью нашей жизни. Они представлены голосовыми помощниками, рекомендательными системами, умными домами, умными автомобилями и другими системами. Важной частью этих систем являются модули, которые помогают сделать понятным для компьютера то, что от него требуется. Для систем по обработке текста это модули обработки естественного языка или Natural Language Processing (NLP).

Компьютер без дополнительной помощи не способен обрабатывать естественный текст, зато компьютер хорошо работает с числами. Поэтому для того, чтобы «подружить» вычислительную машину с текстом, нужно представить текст в виде чисел или в виде многомерных векторов. Эти вектора также называют эмбеддингами.

Модели, использующие данный принцип, называются моделями векторного представления слов. В основе большей части данных моделей лежит гипотеза дистрибутивности [9]. Эта гипотеза заключается в том, что слова со схожим смыслом встречаются в похожих контекстах.

Прорывной и наиболее известной моделью векторного представления слов является выпущенная в 2013 году модель word2vec [6]. Было представлено две архитектуры модели нейронной сети word2vec: Continuous Bag-of-Words (CBOW) и Skip-gram. Continuous Bag-of-Words дословно переводится как «непрерывный мешок слов». Работает архитектура похожим образом, предсказывается вероятность появления слова по его контексту в виде окна фиксированного размера. Архитектура Skip-gram наоборот предсказывает вероятность появления контекста у заданного слова. Порядок слов в контексте не влияет на результат ни в одном из этих алгоритмов. В процессе обучения модель корректирует веса между входным и скрытым слоем, которые в дальнейшем станут эмбеддингами слов.

Оказалось, что полученные векторные представления слов скрывают в себе семантические отношения между словами. Это хорошо заметно на примере задачи по построению пропорциональной аналогии. Эту задачу можно сформулировать так: «какое слово *d* относится к слову *c* так, как слово *b* относится к слову *a*». В модели word2vec это отношение можно выразить в виде разницы векторов. Для слов *a* и *b* с соответствующими им векторами v_a и v_b вектор разности $v_a - v_b$ будет характеризовать семантическую связь между словами. Тогда для решением

задачи пропорциональной аналогии будет выражение $v_d - v_c = v_b - v_a$, где v_d и v_c эмбеддинги слов *d* и *c* соответственно.

Из полученного выражения получаем: $v_d = v_c + v_b - v_a$. Но вероятность того, что полученный вектор v_d совпадает с вектором какого-либо слова крайне мала, поэтому в качестве ответа берется слово с вектором наиболее близким к v_d, формула (1).

$$v_d = \underset{v'}{argmax} \cos(v', v_c + v_b - v_a) \quad (1)$$

Данная задача для модели word2vec исследовалась в работе []. Где пришли к выводу, что эта модель не всегда дает правильный ответ для задачи пропорциональной аналогии.

В настоящее время появляется все больше моделей для обработки естественного языка.

В 2018 году в исследовании [] была предложена модель контекстуализированной модели обработки естественного языка ELMo (Embeddings from Language Models). Если в модели word2vec векторное представление слов было одним и тем же независимо от контекста, то в ELMo решается эта проблема. Для каждого контекста будет свой эмбеддинг. В основе архитектуры ELMo лежат блоки долгой краткосрочной памяти (LSTM - Long Short-Term Memory). Данные блоки расположены в прямом и обратном направлениях для того, чтобы при создании эмбеддинга учитывался контекст до и после слова.

Вскоре после выхода ELMo вышла модель BERT или Bidirectional Encoder Representations from Transformers. BERT – это модель, побившая несколько рекордов по успешности решения ряда NLP-задач, например BERT показала лучшее качество на тесте SQuAD 1.1 [5]. BERT также контекстуализированная модель. Архитектура модели BERT представляет из себя последовательность двунаправленных кодировщиков из Transformer. В основе обучения модели лежат две идеи:

1. Первая состоит в том, чтобы заменить 15% текста масками и заставить модель предсказывать пропущенные слова.
2. Вторая идея заключается в том, чтобы научить модель оценивать насколько одно предложение является логичным продолжением второго.

Успех модели, помимо хорошего качества, можно объяснить тем, что код модели был выложен в открытый доступ, а также были выложены различные

модели, предобученные на больших объемах данных. Это дало возможность всем разработчикам встроить модель BERT в свои модели машинного обучения для обработки естественного языка.

ELMo и BERT - контекстуализированные модели, это значит, что векторное представление одного и того же слова будет отличаться в зависимости от его контекста. Отсюда возникает вопрос, возможно ли провести аффинные преобразования в семантическом пространстве модели BERT?

Целью данной практики является исследование аффинных преобразований для модели BERT и определение точности этих преобразований.

Для достижения поставленной цели потребовалось решить следующие **задачи**:

1. Исследование моделей векторного представления слов;
2. Исследование методов оценки аффинных преобразований;
3. Разработка метода оценки точности параллельного переноса для контекстуализированных моделей;
4. Подготовка экспериментальных данных;
5. Проведение экспериментов;
6. Оценка полученных результатов.

1 Обзор литературы

1.1 Появления моделей векторных представлений слов

Векторное представление слов является одним из ключевых инструментов в обработке естественного языка. Основная идея заключается в том, чтобы сопоставить каждому слову вектор определенной величины.

Самой простой реализацией модели векторного представления слов является one-hot encoding. Идея этой модели заключается в том, что в наборе из *K* слов каждому слову k_i сопоставить вектор v_i длиной *K* со всеми нулями и одной единицей в позиции *i*, где *i* - это номер слова во всем наборе (2).

$$v_i^j = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases}, j = 1, ..., K \tag{2}$$

Недостатком этого метода является то, что по данным векторным представлениям нельзя судить о семантической схожести слов. Также для при обработке реальных текстов размер словаря будет очень большим, а значит длина векторных представлений будет очень большой, данные вектора неэффективно хранить в памяти. Также к многих алгоритмов машинного обучения могут возникнуть сложности с обработкой разреженных векторов.

Позднее появились более продвинутые модели векторного представления. В этих реализациях уже наблюдаются семантические отношения между эмбеддингами слов. Модели векторного представления делятся на две группы в зависимости от используемых методов.

Первая группа - статистические модели векторного представления. При работе этих моделей строится матрица совместной встречаемости слов, далее эта матрица подвергается сингулярному разложению. Одна из полученных после разложения матриц содержит вектора слов.

Вторая группа - предиктивными модели. Данные модели для создания эмбеддинга используют контекст, используются слова, попадающие в окно определенного размера вокруг интересующего нас слова. Для работы используются нейронные сети.

В данный момент большей популярностью пользуются предиктивные модели. Большая популярность пришла к этим моделям с появлением word2vec.

1.2 Word2vec

Идея создания векторов в word2vec основана на предположении о контекстной близости, а именно на том, что слова встречающиеся в одинаковых контекстах скорее всего имеют схожее значение.

Модель word2vec является простейшей нейронной сетью с обратным распространением ошибки. у нее один скрытый слой. В основе обучения модели лежит идея, что тренировать можно не только на контексте из предыдущих слов, но также использовать слова, после целевого слова. При этом порядок слов в контексте не учитывается.

Существует 2 метода обучения word2vec: CBOW (Continuous Bag of Words) и Skip-gram. Основная цель архитектуры Continuous Bag-of-Word состоит в том, чтобы предсказать пропущенное слово по его контексту. В Skip-gram предсказывается контекст по целевому слову.

На рисунке 1 представлены изображения архитектур CBOW и Skip-gram. Где V – мощность словаря, x – слова, подающиеся на вход нейронной сети, закодированные методом one-hot encoding (2) для словаря мощностью V, N – количество нейронов в скрытом слое и y – результат работы функции активации softmax.

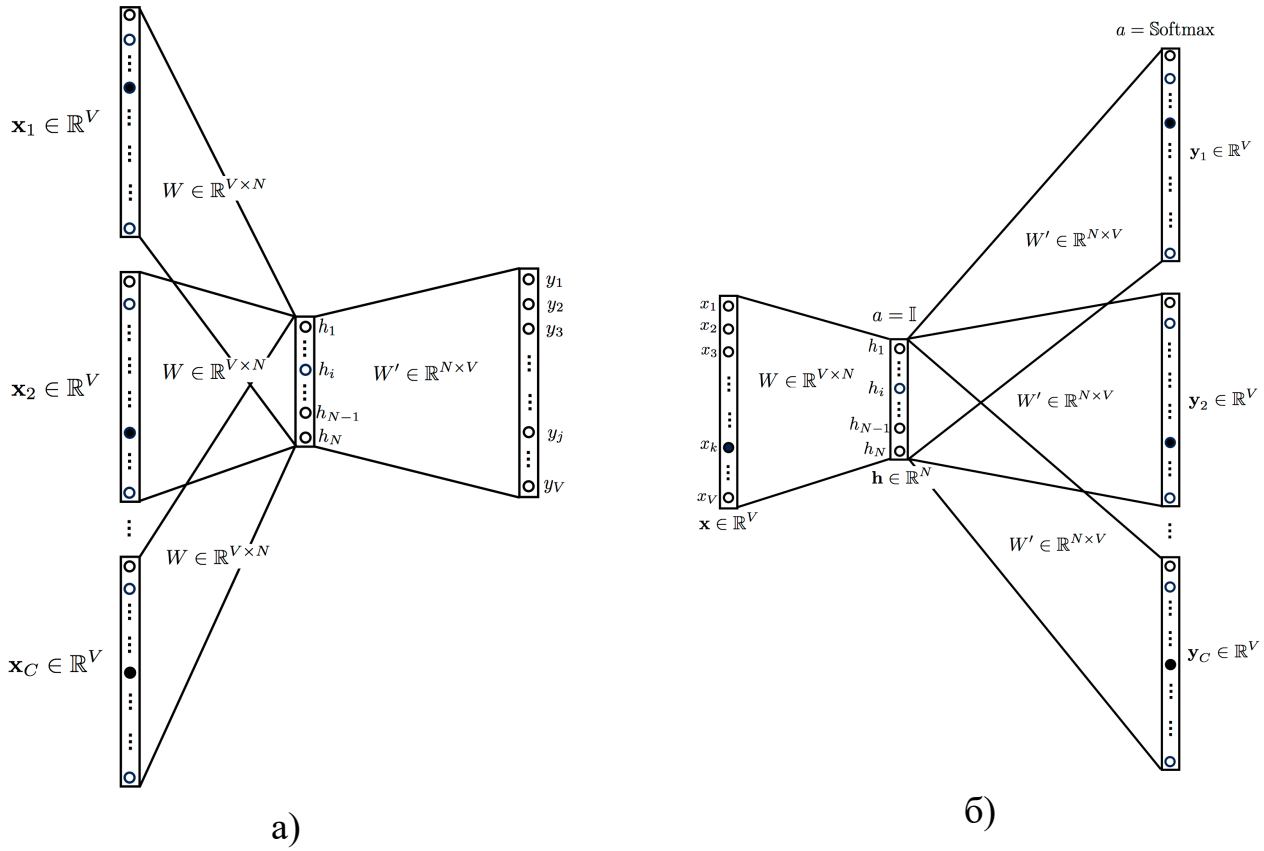

Рис. 1. Схемы архитектур CBOW (а), Skip-gram (б)

Архитектуры представляют из себя полносвязные нейронные сети с обратным распространением ошибки. Для больших словарей метод активации softmax накладывает серьезные вычислительные нагрузки. Поэтому для оптимизации было предложено использовать метод негативного семплирования (negative sampling).

При работе softmax каждое слово представляет из себя отдельный класс, авторы word2vec предложили использовать бинарную классификацию вместо многоклассовой. Предлагается научить модель отличать пары слов, которые встречаются в одном контексте, от тех, которые никогда не стоят в одном контексте.

Появление модели word2vec послужило мощным толчком для развития моделей обработки естественного языка.

1.3 FastText

FastText является продолжением развития модели word2vec. FastText имеет архитектуру Skip-gram. При этом в fastText отличается от word2vec тем, что у новой модели используются N-граммы символов. Например, для слова молоко 3-граммами являются <мо, мол, оло, лок, око, ко>, где символы « < » и « > » кодируют начало и конец слова соответственно. Векторные представления строятся именно для N-грамм, векторные представления слов - это сумма векторных представлений всех его N-грамм. При этом решается проблема того, что словарь модели word2vec был ограничен и не все формы слов вошли в словарь. Также, использование N-грамм позволяет получать векторные представления для редких слов.

В русском языке существуют слова омонимы и омографы, слова, которые совпадают в написании, но имеют разный смысл. Предыдущие методы обработки естественного языка никак не решали эту проблему. Одной из первых эту проблему постаралась решить модель ELMo.

1.4 ELMo

ELMo – модель обработки естественного языка, которая представляет собой двунаправленную рекуррентную нейронную сеть с LSTM. модель была предложена в работах []. Модель учитывает семантическую неоднозначность слов в предложениях, и эмбеддинги, присваиваемые словам, зависят не только от самого слова, но и от контекста. Основная идея получения эмбеддингов – использование скрытых состояний LSTM. Разберемся по подробнее с LSTM блоками.

1.4.1 LSTM

LSTM или Long short-term memory дословно переводится как долгая краткосрочная память, были предложены в статье []. Данные блоки являются разновидностью архитектуры рекуррентных нейронных сетях и предназначены для того, чтобы хранить информацию на длинные и на короткие промежутки времени.

Данные блоки имеют одну особенность, в них нет функции активации. За счет этого хранимая информация не размываются по времени и во время обучения при использовании метода обратного распространения ошибки вычисляемый градиент не исчезает.

В LSTM модуле есть 2 основных компонента: состояние ячейки и различные фильтры. Состояние ячейки – это память сети, которая передается по всей цепочке.

Во время обучения состояние ячейки постоянно меняется. Происходит добавление и удаление информации. Все это контролируют фильтры. Фильтры состоят из сигмоидальной нейронной сети и операции поточечного умножения. Сигмоидальный модуль возвращает числа в диапазоне [0;1], которые обозначают долю блока информации, которую следует пропустить дальше по сети. Фильтры бывают трех типов:

1. Забывания;
2. Входные;
3. Выходные.

На рисунке 2.а изображен фильтр забывания. На данном этапе решается какую информацию можно забыть или оставить. h_{t-1} – значения выхода из предыдущего блока, x_t – вход данного блока. Данные значения проходят обработку в сигмоидальном блоке. Результаты находятся в диапазоне [0;1] то, что ближе к 0 будет забыто, что ближе к 1 оставлено (3).

$$f_t = \sigma(W_f \cdot [h_{t-1},x_t] + b_f) \tag{3}$$

На рисунке 2.б решается какая информация будет храниться в состоянии ячейки. Сигмоидальный блок решает какую информацию необходимо обновить (4), tanh-слой строит вектор со значениями, которые могут быть добавлены в состояние ячейки (5).

$$i_t = \sigma(W_i \cdot [h_{t-1},x_t] + b_i) \tag{4}$$

$$\tilde{C_t} = \text{tanh}(W_C \cdot [h_{t-1},x_t] + b_C) \tag{5}$$

На рисунке 2.в проиллюстрирован процесс изменения состояния ячейки. Ненужная информация в C_{t-1} забывается после умножения на f_t. Затем к состоянию ячейки добавляются текущие изменения $i_t * \tilde{C_t}$. Все вместе можно записать формулой (6).

$$\tilde{C_t} = f - t * C_{t-1} + i_t * \tilde{C_t} \tag{6}$$

На рисунке 2.г изображен финальный формирования выходов. Сначала в сигмоидальном слое поступает информация предыдущего выхода h_{t-1} и текущего входа x_t, где определятся какая информация из состояния ячейки будет отправлена на выход (7). Далее значения из состояния ячейки обрабатываются tanh-слоем и перемножаются со значениями с сигмоидального слоя (8).

$$o_t = \sigma(W_o \cdot [h_{t-1},x_t]) \tag{7}$$

$$\tilde{h}_t = o_t * \text{tanh}(C_t) \tag{8}$$

После преобразований h_t и C_t передаются на следующий блок по цепочке.

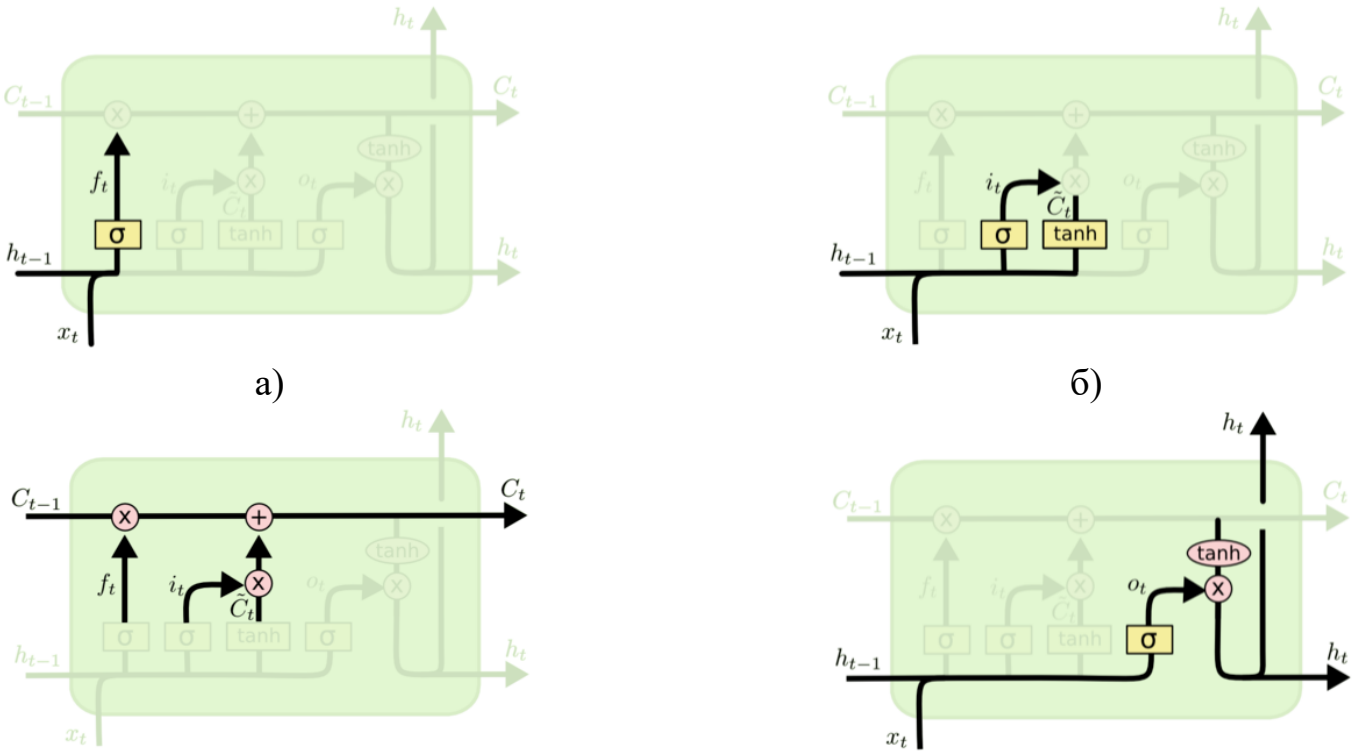

Рис. 2. Виды фильтров: a) слой фильтра забывания, b) слой входного фильтра, c) применение текущих изменений, d) слой выходного фильтра.

1.4.2 Структура ELMo

Структура ELMo представлена на рисунке 3. ELMo состит из двух двухслойных разнонаправленных рекуррентных LSTM сетей.

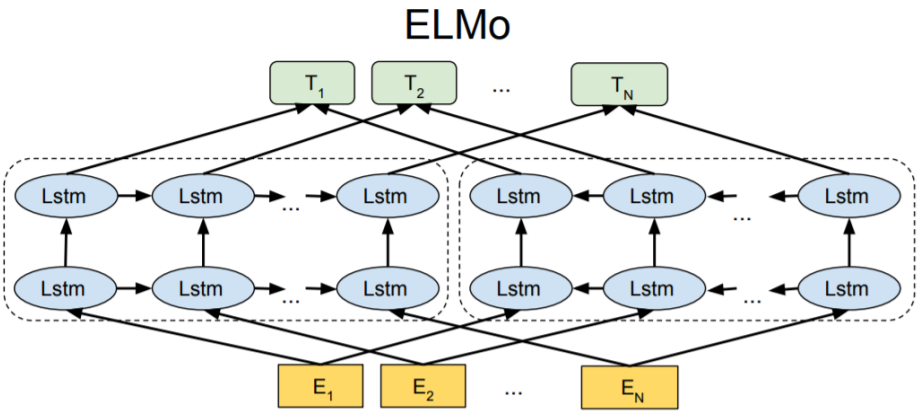

Рис. 3. Структура модели ELMo

1.5 Выводы

2 Теоретическая часть

2.1 Выводы

1.

3 Практическая часть

3.1 Выводы

1.

4 Заключение

Список использованных источников

1.