Министерство образования Республики Беларусь
Учреждение образования Белорусский государственный университет информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Машинное обучение (МО)

РЕФЕРАТ

на тему: «Представления слов в задачах обработки естественного языка (Word2Vec, negative sampling, GloVe).»

Выполнил: студент гр. 858601 Губский М.Д.

Проверил: Стержанов М.В.

Минск 2020

Содержание

ВВЕДЕНИЕ

Человечество с давних времен начало обмениваться информацией для решения всякого рода задач и удовлетворения своих потребностей. Однако, если отправитель не мог преобразовать информацию в понятное для адресата преставление - донести свою мысль, - она могла быть воспринята неверно. Именно для этого давным давно люди придумали способ формализовать информацию и свои мысли — они придумали языки.

Язык — один из основных способов представления информации, будь то текст, аудио, изображение, мысль или даже чувство.

Человечество испокон веков пыталось облегчить себе жизнь, создавая всевозможные приспособления для удовлетворения нужд. Всё начиналось с упрощения физического труда, однако с развитием вычислительных приборов стало возможным облегчение и умственного. Со временем люди смогли переложить не только решение прикладных вычислительных задач, но и обмен информацией.

За последние десятилетия стало создаваться и передаваться так много информации, что человек стал не справляется с ее обработкой. По этой причине земляне решили переложить и эту задачу на плечи машин. С появлением больших вычислительных мощностей это стало возможно. Одной из проблем в реализации данной задумки оказалось представление собранных данных в понятный и удобный для компьютера язык.

Существенная часть аккумулированной информации имеет текстовое представление, однако, исторически сложилось так, что компьютеры лучше работают с числами, а не с текстом, поэтому и возникла задача преобразования текстовых данных в числовой материал.

О методах решения этой задачи и пойдет речь в этом реферате.

1. Обработка естественного языка

Естественный язык — это язык, на котором общаются люди в повседневной жизни (английский, французский и т.д.).

Под обработкой естественного языка (Natural Language Processing, NLP) понимают область искусственного интеллекта, которая направлена на разработку компьютерных систем для анализа человеческого языка.

Сегодня к НЛП существует сильный интерес благодаря огромным объемам данных, размещенных в общем доступе, и увеличению вычислительных мощностей, которые позволяют, к примеру, практикующим врачам быстрее и точнее собрать анамнез у пациентов. Также НЛП применяется не только в здравоохранение, но и в таких отраслях как СМИ, финансы и человеческие ресурсы и другие. Круг задач по обработке естественного языка стал намного шире: задача распознавания речи, перевод речи в текст, с которым можно работать, анализ тональности текста, классификация именованных объектов и др.

Несколько задач, для которых используется НЛП рассмотрим подробнее.

1.1. Анализ тональности текста

Анализ тональности текста подразумевает автоматическое определение эмоциональной окраски текста и выявление отношения человека, написавшего текст, к объекту обсуждения. Данный тип анализа может быть использован, например, в маркетинге для того, чтобы лучше понять, какой из продаваемых товаров пользуется большим успехом среди покупателей, анализируя отзывы. Также его могут использовать власти для выявления отношения граждан страны к поднимаемым вопросам и к предлагаемым решениями т.д. В наше время наиболее часто используются методы на основе машинного обучения с учителем. Сутью таких методов является то, что на первом этапе обучается машинный классификатор на заранее размеченных текстах, а затем используют полученную модель при анализе новых документов. [1]

1.2. Машинный перевод

Также одним из наиболее известных и часто используемых направлений обработки естественного текста является его перевод с одного естественного языка на другой. Одной из самых продвинутых методик для достижения перевода, максимально приближенному к человеческому, является использование нейронных сетей типа “Seq2Seq”[2] с “Attention”[3], что расшифровывается, как “sequence to sequence”, или “из последовательности в последовательность”.

1.3. Классификация текста

Классификация текста (так называемая текстовая категоризация или тегирование текста) — это задача назначения тексту набора предопределенных категорий. Классификаторы текста могут использоваться для организации, структурирования и категоризации практически чего угодно. Например, новые статьи могут быть организованы по темам, заявки в службу поддержки могут быть организованы по срочности, чаты могут быть организованы по языку, упоминания бренда могут быть организованы по настроению и так далее.

Возвращаясь к НЛП, стоит отметить, что работа с текстовыми данными является непростой, поскольку компьютеры, сценарии и модели машинного обучения не могут читать и понимать текст в человеческом смысле. Они лучше работают с числовыми представлениями.

На самом деле, не так важен алгоритм преобразование, сколько охват языковых и смысловых характеристик слов.

Самым распространенным подходом к этой проблеме является вложение слова (word embedding).

2. Вложение слова

В традиционном NLP слова рассматриваются как дискретные символы, которые далее представляются в виде one-hot векторов. Проблема со словами — дискретными символами — отсутствие определения параметров схожести между векторами. Поэтому альтернативой выступает включение этих параметров в сами векторы.

Вложение слов (встраивание слов, word embeding) — это тип представления слов, который позволяет алгоритмам машинного обучения понимать слова со схожим значением. Технически говоря, это отображение слов в векторы действительных чисел с использованием нейронной сети, вероятностной модели или сокращения размеров в матрице совместного использования слов. Вложение слова также называется распределенной семантической моделью или распределенным представленным или семантическим векторным пространством или моделью векторного пространства. Всё это подразумевает категоризацию похожих слов. Например, фрукты, такие как яблоко, манго, банан, должны быть расположены близко, тогда как книги будут находиться далеко от этих слов. В более широком смысле, встраивание слов создаст вектор фруктов, который будет расположен далеко от векторного представления книг.

Доступны различные модели встраивания слов, такие как word2vec (Google), Glove (Stanford) и т.д.

Использование слов в качестве подготовленных признаков в задачах обработки естественного языка – интуитивное и естественное решение, потому что слова являются основными значащими единицами для естественных языков. И не удивительно, что представлению слов в виде числовых векторов для последующего применения методов машинного обучения посвящено большое количество исследований.

На рисунке 1. представлен простейший вариант векторного представления слов с помощью one-hot кодирования .

Все методы векторного представления слов плохо обобщаются на морфологически богатые языки. Скорее всего, это происходит по причине проблемы «словарного взрыва». Также одной из проблем может стать тот

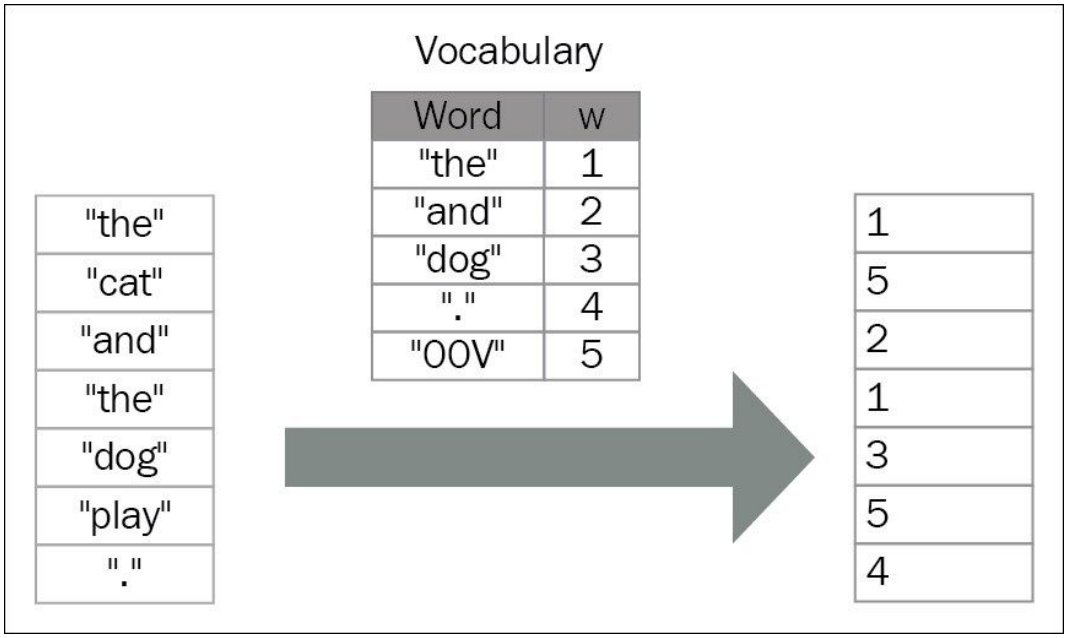

Рисунок 1. Представление слов с помощью one hot кодирования

факт, что даже если word embedding словаря большой, некоторые слова по-прежнему будут в нем отсутствовать. Эту проблему можно частично решить с помощью добавления этапа лемматизации слов перед поиском словаря векторных представлений. Лемматизация, выполненная с достаточной точностью, может полностью устранить данную проблему для некоторых морфологически богатых языков, но существуют и такие языки, в которых структура слов слишком сложная. Обработка таких слов, не вошедших в словарь, вызывает затруднения, поскольку представление слов не может быть аппроксимировано из письменной формы. Письменная форма слова содержит лишь малую часть семантического значения слова, поэтому методы локальных окон обрабатывают семантическую информацию исключительно из контекста, в котором появляется слово.

3. Методы представления слов

3.1. Word2Vec

Word2Vec — эта модель для вложения слов для лучшего их представления. Она фиксирует большое количество точных синтаксических и семантических словосочетаний. Данный метод был разработан группой исследователей во главе с Томасом Миколовым из Google.

Данная модель представляет слова в виде векторов, и их размещение выполняется таким образом, что сходные по значению слова появляются вместе, а разнородные слова расположены далеко. Это также называется семантическими отношениями. Нейронные сети не понимают текст, а понимают только цифры. Вложение слов обеспечивает способ преобразования текста в числовой вектор.

Word2Vec реализует две основные архитектуры — Continuous Bag of Words (CBOW) и Skip-gram. На вход подается корпус текста, а на выходе получается набор векторов слов (рисунок 2).

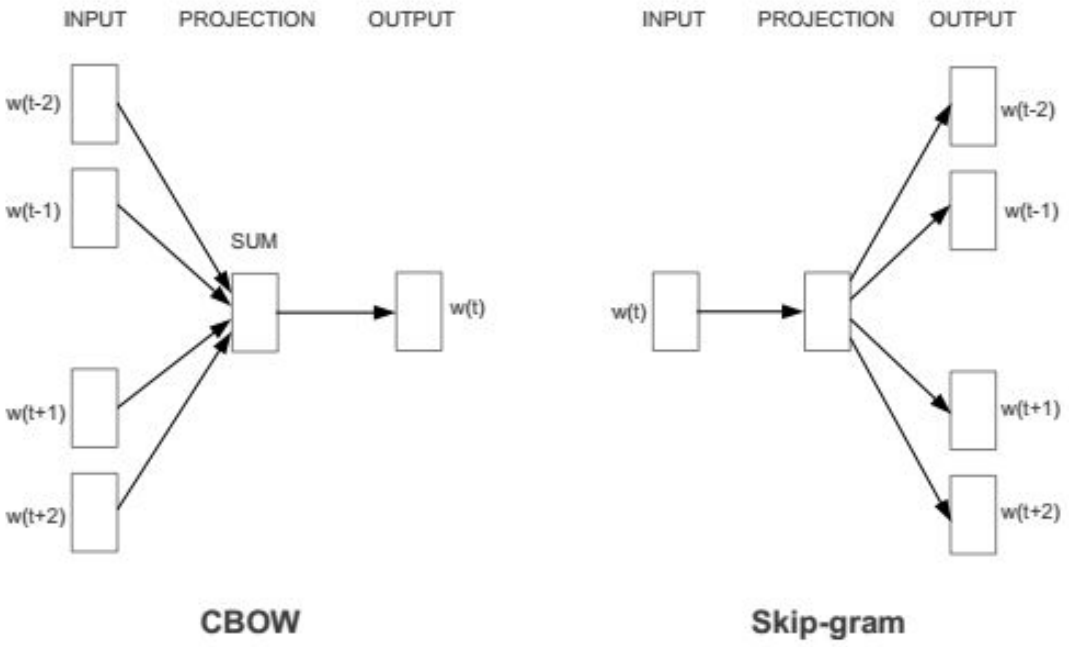

Рисунок 2. Continuous Bag of Words (CBOW) и Skip-gram

Обе архитектуры являются трехслойными нейронными сетями (один скрытый слой). Оба эти метода изучают веса, которые действуют как представления слов в векторе.

Более формально задачу Word2Vec можно поставить следующим образом: максимизировать косинусную близость между векторами слов (скалярное произведение векторов), которые появляются рядом друг с другом, и минимизировать косинусную близость между векторами слов, которые не появляются друг рядом с другом. Рядом друг с другом в данном случае значит в близких контекстах.

3.1.1. CBOW

CBOW (Continuous Bag of Words) или непрерывный мешок слов позволяет предсказывать текущее слово, основываясь на его контексте, который определяется соседними словами в окне. В CBOW используются три слоя. Входной слой соответствует контексту. Скрытый слой – проекции каждого слова из входного слоя в весовую матрицу, которая проецируется в третий выходной уровень. Последним этапом модели является сравнение ее вывода с самим словом, чтобы скорректировать его представление, основанное на обратном распространении градиента. Таким образом, целью нейронной сети CBOW является максимизация следующего выражения:

$$J = \frac{1}{V}\sum_{t=1}^{V} log(P(m_t|m_{t-w/2}..m_{t+w/2}) \rightarrow max, \quad (1)$$

$$P(c|t) = \frac{exp(u_c^T \overline{v}_t)}{\sum_{i \in V} exp(u_i^T \overline{v}_t)}, \quad (2)$$

$$\overline{v}_t = (v_{01} + ... + v_{0w}) / 2w, \quad (3)$$

где V – размер словаря, w – размер окна для каждого слова, u_i - вектор контекстного слова, v_i - вектор центрального целевого слова.

Согласно работе Миколова [5] CBOW работает быстрее, чем skip-gram и лучше обрабатывает часто встречающиеся слова.

3.1.2. Skip-gram

Метод skip-gram решает обратную задачу: на основании одного слова предсказывается контекст. Последний шаг алгоритма – сравнение вывода с каждым словом в контексте с целью корректирования представления, основанного на обратном распространении градиента. Данный метод выполняет максимизацию следующего выражения:

$$J = \frac{1}{V} \sum_{t=1}^{V} \sum_{j=t-w,\, j \neq w}^{t+w} log(P(m_j|m_t)) \rightarrow max, \quad (4)$$

$$P(t|c) = \frac{exp(u_t^T v_c)}{\sum_{i \in V} exp(u_i^T v_c)}, \quad (5)$$

где V – размер словаря, c – размер окна для каждого слова, u_i - вектор контекстного слова, v_i - вектор центрального целевого слова.

Согласно работе Миколова[5] skip-gram наиболее эффективен при обработке небольшого корпуса обучающих данных. Более того, с помощью него хорошо описываются редко встречающиеся слова.

Таким образом Word2Vec ищет векторные представления различных слов, максимизируя логарифмическую вероятность встречаемости слов контекста для данного центрального слова и преобразуя векторы методом стохастического градиентного спуска.

Пожалуй, самым интересным вкладом Word2Vec в развитие NLP стало появление линейных отношений между разными векторами слов. После обучения векторы отражают различные грамматические и семантические концепции.

$$x_{\text{рубашка}} - x_{\text{одежда}} \approx x_{\text{стук}} - x_{\text{мебель}}$$

$$x_{\text{король}} - x_{\text{мужчина}} \approx x_{\text{королева}} - x_{\text{женщина}}$$

Однако обучение выходного вектора алгоритмов CBOW и skip-gram представляет собой одно из самых больших ограничений этих моделей, так

как оно может быть тяжелой задачей, требующей больших вычислительных затрат. Для решения этой проблемы можно использовать два алгоритма. Первый – негативное семплирование, о котором описано далее. Второй алгоритм – иерархический softmax. Он основан на дереве Хаффмана. Фактически, это двоичное дерево, которое представляет все термины на основе их частоты появления в корпусе текста. Затем каждый шаг от корня до цели нормализуется. По мнению Миколова, каждый алгоритм обладает преимуществами по сравнению с другим в зависимости от обучающих данных. Например, негативное семплирование более эффективно работает с векторами малой размерности и часто употребляемыми словами. Тем не менее, иерархический softmax показывает себя лучше в работе с редко употребляемыми словами. [4]

3.2. Negative sampling

Основная идея алгоритма заключается в ограничении количества выходных векторов, которые требуется обновлять. Таким образом, только некоторые векторы обновляются случайным образом. Образцы называются негативными, потому что это слова, которые не относятся к контексту c_i (и поэтому модели в идеале следует присваивать вероятность, равную нулю). Это распределение шума является вероятностным и используется в процессе семплирования.

Алгоритм был предложенная Миколовым [5], чтобы решить проблемы высокой вычислительной нагрузки при вычислении выходных параметров в Word2Vec модели.

Более формально идея подразумевает преобразование проблемы многочленной классификации в проблему двоичной классификации: для каждой обучающей выборки модель снабжается положительной парой (центральное слово и следующее из контекста) и небольшим количеством отрицательных пар (центральное слово и случайно выбранное слово из словаря). Модель учится отличать истинные пары от отрицательных.

Вероятность попадания слов в выборку можно записать следующим образом:

$$P(c_i) = \frac{f(c_i)}{\sum_{j=0}^{V} f(c_j)}, \quad (6)$$

где c_i - слово из контекста, $f(c_i)$ - вероятность появления c_i слова (рассчитывается равновероятностно между всеми словами контекста)

Авторы утверждают в своей статье [5], что они пробовали несколько вариантов этого уравнения, и наиболее эффективным оказалось возвести вероятности в степень 3/4:

$$P(c_i) = \frac{f(c_i)^{3/4}}{\sum_{j=0}^{V} f(c_j)^{3/4}}, \quad (7)$$

Таким образом, отрицательной выборке целевой функцией становятся:

$$J = log(P(t \mid c_i)) = log(\delta(v_t v_{c_i}^T)) + \sum_{j=1}^{V} log(\sigma(- v_{c_j} v_{c_i}^T)), \quad (8)$$

где σ - логистическая (сигмоидальная) функция:

$$\delta(x) = \frac{1}{1+exp(-x)}, \quad (9)$$

В отличие от SoftMax , сигмовидной является функцией одного аргумента, и поэтому не требуют , чтобы все выходные значения должны были бвть оценены в то же самое время.

Также в документе [5] говорится, что выбор 5-20 слов хорошо подходит для небольших наборов данных, и вы можете выбрать только 2-5 слов для больших наборов данных.

3.3. GloVe

GloVe (Global Vectors) – один из наиболее популярных методов векторного представления слов, был предложен Пеннингтоном в 2014 году [6]. В его основе лежит способ подсчета частоты появления слов в текстовом корпусе. Фактически он состоит из двух основных этапов.

Первый этап – построение матрицы смежности X из обучающего корпуса, где X_{ij} – частота появления слова i вблизи слова j. Тогда общее количество слов i в корпусе (V соответствует размеру корпуса) можно рассчитать по формуле:

$$X_{ij} = \sum_{k}^{V} X_{ik}, \quad (10)$$

Второй этап – факторизация матрицы X для получения векторов. В работе Пеннингтона [6] показано, что информативны отношения вероятностей совпадения двух слов, а не сами вероятности совпадения, и они используются для кодирования этой информации в виде разности векторов:

$$F(w_i - w_j, \overline{w_k}) = \frac{P_{ik}}{P_{jk}}, \quad (11)$$

$$P_{ik} = \frac{X_{ik}}{X_i}, \quad (12)$$

где w_i, w_j и w_k – три вектора слов, P_{ik} – вероятность появления слова k в контексте слова i, w – векторы слов и $\overline{w_k}$ – вектор контекстного слова.

Однако для сохранения линейности и предотвращения смешивания измерений, Пеннингтон использовал разности векторов и точечный результат аргументов:

$$F((w_i - w_j)^E, \overline{w_k}) = \frac{P_{ik}}{P_{jk}}, \quad (13)$$

Но окончательная модель не должна изменяться после замены $w \rightarrow \overline{w_k}$ и $X \rightarrow X^T$. Чтобы работало свойство симметрии, уравнение следует преобразовать следующим образом:

$$w_i^T \overline{w_k} + b_i + \overline{b_k} = log(X_{ik}), \quad (14)$$

В итоге Пеннингтон предложил использовать метод наименьших квадратов, как показано в уравнении,

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij})(w_i^T \overline{w_k} + b_i + \overline{b_k} - log(X_{ik}))^2, \quad (15)$$

где $f(x)$ – весовая функция.

В работе [6] показано, что выдаваемые системой GloVe векторные представления показывают себя в задачах нахождения синонимов, гипонимов и гиперонимов лучше или не хуже, чем модель Word2Vec. Также стоит отметить, что для обучения системы, дающей высокое качество в упомянутых задачах требуется меньший, чем системе Word2Vec, корпус данных. Однако, в этой системе присутствуют те же проблемы, что и в модели Word2vec, связанные с обработкой слов не из словаря и неучетом локального контекста.

ВЫВОДЫ

В ходе проведения исследования было установлено, что в нейролингвистическом программировании является важной сферой в оптимизации работы человека и удовлетворении его нужд. В свою очередь векторное представление слов является одной из основных проблем в НЛП.

На данный момент существует большое количество методов векторного представления слов со своими плюсами и минусами. К сожалению, универсального “лекарства” не существует. Поэтому стоит исходить из конкретной задачи.

Рассмотренные системы считаются мощными инструментами в представлении слов в векторном виде. Однако основным их недостатком является сложность и, как следствие, необходимость в высоких вычислительных мощностях.

Хотелось бы отметить, что, если выбирать между двумя системами (Word2Vec и GloVec) в конкретной задаче, стоит обратить внимание на объем имеющихся данных. Так Word2Vec требует больший корпус и может находить довольно глубокие связи. В свою очередь GloVec позволяет достичь существенных результатов с меньшим объемом данных.

ЛИТЕРАТУРА

[1] Анализ_тональности_текста – Режим доступа: https://ru.wikipedia.org/wiki/Анализ_тональности_текста

[2] Neural Machine Translation (seq2seq) Tutorial – Режим доступа: https://www.tensorflow.org/tutorials/seq2seq

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin Attention Is All You Need – ARXIV, Электронная версия печ. публикации arXiv:1706.03762, 06/2017 – PDF формат, версия 5 – Режим доступа: https://arxiv.org/pdf/1706.03762.pdf

[4] И.В. Бондарева, Д.Г. Лагерев: Исследование методов векторного представления текстовой информации для решения задачи анализа тональности, Электронная версия печ. публикации 2018 — Режим доступа: http://itids.ugatu.su/index.php/itids/itids2018/paper/download/2/55

[5] Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space // Proc. of Workshop at ICLR. 2013. P. 1301-3781.

[6] Pennington, J., Socher R., Manning C.D. Global Vectors for Word Representation. // Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing, P. 1532–1543.

[7] McCormick, C. (2017, Jan 11): Word2Vec Tutorial Part 2 - Negative Sampling. Режим доступа: http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/

[8] Weng, L. (2017, Oct 15): Learning Word Embedding. Режим доступа: https://lilianweng.github.io/lil-log/2017/10/15/learning-word-embedding.html