Векторные представления документов, полученные путем преобразований словаря

Шундеев Александр Сергеевич
кандидат физико-математических наук
МГУ имени М.В. Ломоносова
119192 Москва, Мичуринский проспект, дом 1
alex.shundeev@gmail.com

Аннотация: Методы интеллектуального анализа данных все чаще становятся основой математического обеспечения прикладных информационных систем. Популярным подходом к интеллектуальному анализу текстовых данных является использование так называемых векторных представлений слов и текстов. В докладе будет рассмотрен оригинальный метод формирования векторного представления набора текстов, согласованного с заданным векторным представлением слов.

Ключевые слова: векторное представление слов, векторное представление документов, классификация текстов, регрессия, Word2Vec, Doc2Vec.

Towards document embeddings using dictionary transformation

Alexandr S. Shundeev
Candidate of Physical and Mathematical Sciences
Lomonosov Moscow State University
Moscow, ul. Michurinskiy prospekt, d. 1, Russia 119192
alex.shundeev@gmail.com

Abstract: Data mining methods are increasingly becoming the basis of software for applied information systems. A popular approach to mining textual data is the use of word and document embeddings. The report considers a method of forming a vector representation of a set of documents that is consistent with a given vector representation of words.

Keywords: word embeddings, document embeddings, text classification, regression, Word2Vec, Doc2Vec.

Введение

Сфера приложений интеллектуального анализа текстовых данных включает целый ряд взаимосвязанных задач, для решения которых часто используются общие принципы и подходы. К числу таких задач относятся задача определения смысловой близости слов (word similarity), задача поиска аналогий (word analogies), задача классификации текстов (document classification, text categorization). Приведенный список не является исчерпывающимся.

Общие для выделенных задач методы их решения будут представлять объект исследования в контексте настоящей работы.

Одним из популярных и показавших свою практическую эффективность подходов к решению первых двух из числа перечисленных выше задач стало использование так называемых векторных представлений слов (word embeddings). Здесь можно отметить модели Word2Vec [1], [2] и модель GloVe [3]. В рамках этих моделей каждому слову ставится в соответствие вещественный вектор фиксированной размерности. При таком подходе предоставляется возможность сопоставить близким по смыслу словам близкие вектора.

Под упомянутыми выше аналогиями понимаются высказывания следующего вида: слово A связано по смыслу со словом B, как слово C связано по смыслу со словом D. В задаче поиска аналогий слова A, B, C заданы, а слово D необходимо подобрать. Например, слово *женщина* относится к слову *королева*, как слово *мужчина* относится к словам *король*, *принц*, *царь*, *герцог*. Слова *король*, *принц*, *царь*, *герцог* являются найденными решениями.

На языке векторных представлений задача поиска аналогий имеет элегантную формулировку и решение. Предположим, что словам A, B, C, D сопоставлены соответственно вектора v_A, v_B, v_C, v_D. Тогда делается предположение, что аналогия может быть записана в виде выражения $v_B - v_A \approx v_D - v_C$. Поэтому в качестве вариантов искомого слова D нужно подбирать слова, векторные представления которых близки вектору $v_B - v_A + v_C$.

Задача классификации текстов [4] состоит в том, чтобы отнести каждый документ из рассматриваемого набора к определенному классу. Число классов при этом фиксировано. В некоторых постановках этой задачи допускается возможность отнести документ сразу к нескольким классам. Однако в дальнейшем этот случай рассматриваться не будет. Предположение о том, что в один класс должны попадать тематически близкие документы, позволяет эффективно использовать векторные представления слов для решения этой задачи. Например, текст может быть представлен как последовательность, составленная из векторов входящих в него слов. Такое представление в частности используется при построении классификаторов на основе сверточных нейронных сетей (convolutional neural networks) [5].

Недостатком такого подхода является то обстоятельство, что разным текстам могут соответствовать вектора разной размерности. Модели векторного представления текстов Doc2Vec [6] позволяют обойти подобное ограничение. Оказывается возможным одновременно построить векторные представления одной фиксированной размерности, как для текстов, так и для слов, встречающихся в этих текстах. В результате предоставляется возможность определять смысловую близость и аналогии не только между словами, но и между текстами, а также между текстами и наборами ключевых слов. В дальнейшем, подобные векторные представления текстов и слов будем называть согласованными.

Создание векторных представлений является трудоемкой вычислительной задачей. В связи с этим обстоятельством появились подходы [7], реализующие так называемую пост-обработку готовых векторных представлений с целью повышения их качества. В некоторых случаях побочным эффектом пост-обработки является уменьшение размерности векторного представления.

Основной задачей исследования, результаты которого представлены в настоящей работе, является выработка подхода к созданию векторных представлений текстов, согласованных с заданным векторным представлением слов. Предполагается, что изначально у рассматриваемого векторного представления слов не было согласованного с ним векторного представления текстов. Побочным результатом решения этой задачи оказалась возможность существенно снижать размер исходных векторных представлений без потери их качества. В данном случае качество оценивается с точки зрения возможности эффективного использования векторного представления для решения задачи классификации текстов. В этой связи предложенный подход можно отнести к области пост-обработки векторных представлений.

Дальнейшее изложение структурировано следующим образом. В разделе 1 в качестве примеров векторных представлений слов и текстов описываются модели Word2Vec и Doc2Vec. В разделе 2 приводятся основные понятия и описываются используемые в данной работе методы машинного обучения [8]. В разделе 3 описывается алгоритм построения векторного представления текстов, согласованный с заданным векторным представлением слов. Приводятся результаты тестирования этого алгоритма. В заключении обсуждаются полученные результаты и описываются возможные направления для дальнейших исследований.

1 Векторные представления

В качестве примеров векторных представлений слов и документов рассмотрим модели Word2Vec [1], [2] и Doc2Vec [6].

1.1 Модель Word2Vec

В основе построения многих моделей векторных представлений слов лежит так называемая дистрибутивная гипотеза (distributional hypothesis) [9]. Согласно этой гипотезе смысл слова определяется распределением слов, в окружении которых оно встречается в текстах. Покажем, как эта гипотеза реализуется в моделях Word2Vec. В качестве примера рассмотрим следующее предложение:

«*Лиса схватила зайца, пока он ел морковку*».

После предварительной обработки текста, включающей запись слов в нижнем регистре, приведение слов к их словарной форме (лемматизация), удаление часто встречающихся слов и знаков пунктуации, получим последовательность:

«*лиса хватать заяц есть морковь*».

Предположим, что из этой последовательности было удалено третье слово, и был оставлен только контекст (окружение), в котором оно присутствовало:

«*лиса хватать* ____ *есть морковь*».

Видя этот контекст, содержащий такие слова как *лиса* и *морковь*, можно с высокой долей уверенности предположить, что было пропущено слово *заяц*.

Подобные рассуждения были положены в основу варианта модели Word2Vec под названием continuous bag-of-words (CBOW). В рамках модели CBOW производится анализ всех контекстов заданного фиксированного размера, встречающихся в рассматриваемом наборе текстов. Целевой установкой является оценка условных вероятностей появления разных слов в окружении рассматриваемых контекстов.

В рамках варианта под названием skip-gram (SG) модели Word2Vec решается обратная задача. По заданному слову требуется предугадать слова, которые могут встречаться в его контексте. Для слова *заяц* можно с высокой долей уверенности предположить, что в тексте его соседями могут быть такие слова, как *лиса* и *морковь*.

С математической точки зрения, наиболее корректное описание моделей CBOW и SG приводится в работе [10]. Следуя этой работе, опишем формальную постановку и решение задачи для модели CBOW в случае, когда контекст состоит из одного слова. Например, для заданного слова (контекста) требуется предсказать, какие слова могут следовать за ним в рассматриваемом наборе текстов.

Пусть D – это словарь, содержащий все рассматриваемые слова. Каждому слову $w \in D$ ставятся в соответствие два вектора $v_w, \acute{v}_w \in \mathbb{R}^n$. Первый вектор интерпретируется как векторное представление слова w, а второй вектор носит вспомогательный характер. Фиксированная размерность векторов n выбирается заранее.

Условная вероятность появления слова $w \in D$ в контексте слова $c \in D$ моделируется с помощью выражения вида

$$p(w|c) = \frac{\exp\langle \acute{v}_w, v_c \rangle}{\sum_{s \in D} \exp\langle \acute{v}_s, v_c \rangle}.$$

Приведенное выражение является эвристикой. Правомочность использования этой эвристики подтверждается практикой. Заметим, что вектора, фигурирующие в правой части выражения, изначально не известны и должны быть вычислены. Для этого используются методы математической статистики. Для условных вероятностей записывается функция правдоподобия

$$L(\Theta) = \prod_{(w,c) \in T} p\,(w|c;\Theta).$$

Оцениваемыми статистическими параметрами являются вектора, которые сопоставляются словам из словаря

$$\Theta = \{(v_w, \acute{v}_w) | w \in D\}.$$

В определении функции правдоподобия фигурирует множество T. Оно составлено из всевозможных пар вида (w, c), где c – контекст, w - слово, появившееся в контексте c, извлеченных из заданного набора текстов. В рассмотренном ранее примере множество T состоит из пар слов (*хватать*, *лиса*), (*заяц*, *хватать*), (*есть*, *заяц*), (*морковь*, *есть*).

Далее, следуют стандартные шаги по переходу к логарифмической функции правдоподобия (переход от произведения множителей к сумме слагаемых) и решению задачи ее максимизации

$$l(\Theta) = \sum_{(w,c) \in T} \ln p(w|c;\Theta) \longrightarrow \max_{\Theta}.$$

На практике решается эквивалентная задача минимизации функции $-l(\Theta)$. Для этого используется метод стохастического градиентного спуска, идея которого будет обсуждаться позже. Следует отметить, что целевая функция $-l(\Theta)$ не является выпуклой. Поэтому говорить о гарантированном нахождении глобального экстремума не приходится. Использование стохастического алгоритма означает, что результат решения задачи будет зависеть от выбора начального значения генератора псевдослучайных чисел. Проведение серии повторных вычислений, при которых варьируется это начальное значение, позволит получить разные варианты решения задачи. Из этих вариантов может быть выбран наилучший.

Итоговое векторное представление образуют вектора v_w ($w \in D$), а вектора $\acute{v}_w$ отбрасываются.

1.2 Модель Doc2Vec

Существует два варианта модели Doc2Vec. Вариант distributed memory (DM), соответствующий варианту CBOW модели Word2Vec, а также вариант distributed bag-of-words (DBOW), который соответствует варианту SG модели Word2Vec.

В обоих упомянутых выше вариантах каждому тексту сопоставляется новое уникальное слово, которое интерпретируется как его идентификатор. Для этих идентификаторов и обычных слов, встречающихся в рассматриваемых текстах, совместно строятся векторные представления. Векторное представление идентификатора интерпретируется как векторное представление соответствующего этому идентификатору текста.

В рамках модели DM моделируются условные вероятности появления различных слов в окружении рассматриваемых контекстов. При этом каждый контекст расширяется за счет добавления к нему идентификатора текста, в котором он был обнаружен.

В рамках модели DBOW моделируются условные вероятности появления различных контекстов внутри заданного текста. При этом контексты интерпретируются как неупорядоченные наборы слов.

2 Методы машинного обучения

Приведем основные понятия и методы из области машинного обучения [8], которые будут использоваться в дальнейшем.

Постановка задачи машинного обучение с учителем (supervised learning) предполагает наличие множества объектов $\mathcal{X}$, множества целевых значений $\mathcal{Y}$, а также неизвестной функциональной зависимости между объектами и целевыми значениями. Об этой функциональной зависимости можно судить только по конечному множеству обучающих примеров

$$\mathrm{T} = \{(\mathrm{x_i}, \mathrm{y_i}) \mid \mathrm{i} = 1, \dots, \mathrm{m}\} \subset \mathcal{X} \times \mathcal{Y}.$$

Модель машинного обучения фиксирует множество гипотез $\mathcal{H}$ (функций вида $\mathrm{h}\colon \mathcal{X} \to \mathcal{Y}$), среди которых выбирается «приближение» к неизвестной функциональной зависимости. Если $|\mathcal{Y}| < \infty$, то говорят о задаче классификации. В этом случае элементы множества $\mathcal{Y}$ называются классами. Если $\mathcal{Y} = \mathbb{R}$, то говорят о задаче восстановления регрессии. Модель машинного обучения включает алгоритм, который по заданному множеству обучающих примеров T строит гипотезу $\mathrm{h_T} \in \mathcal{H}$, которая рассматривается как результат решения задачи машинного обучения.

Должны быть зафиксированы числовые метрики вида $\mathrm{est}(\mathrm{T}, \mathrm{h})$, позволяющие судить о том, насколько хорошо гипотеза $\mathrm{h} \in \mathcal{H}$ соответствует множеству T. Подобные метрики используются для оценки качества построенного решения $\mathrm{h_T}$, а также могут использоваться в процессе работы алгоритма машинного обучения, в некоторых случаях являясь настраиваемым параметром.

В процессе решения задачи машинного обучения часто приходится сталкиваться с явлениями недообучения (underfitting) и переобучения (overfitting). Если оценка $\mathrm{est}(\mathrm{T}, \mathrm{h_T})$ признается неудовлетворительной, то констатируют случай недообучения. Если построенная гипотеза $\mathrm{h_T}$ показывает хорошие результаты только на обучающих примерах из T, то констатируют случай переобучения. Эти две ситуации являются взаимосвязанными. Меры, направленные на улучшение одной из них, могут приводить к усугублению другой.

Наличие явления переобучения свидетельствует о том, что одной оценки $\mathrm{est}(\mathrm{T}, \mathrm{h_T})$ недостаточно, чтобы судить о качестве решения задачи машинного обучения. Чтобы обойти это ограничение, были разработаны методы, получившие общее название скользящий контроль (cross validation). Один из вариантов скользящего контроля подразумевает случайное разбиение обучающего множества на два непересекающихся подмножества $\mathrm{T} = \mathrm{T_{train}} \cup \mathrm{T_{test}}$. Обычно тренировочная выборка $\mathrm{T_{train}}$ содержит 70% обучающих примеров, а тестовая выборка $\mathrm{T_{test}}$ содержит оставшиеся 30%.

Для построения решения используется только множество $\mathrm{T_{train}}$. Далее, анализируются две оценки $\mathrm{est}\left(\mathrm{T_{train}}, \mathrm{h_{T_{train}}}\right)$ и $\mathrm{est}\left(\mathrm{T_{test}}, \mathrm{h_{T_{train}}}\right)$. По первой оценке судят о том, имело ли место недообучение. Сравнивая обе оценки, судят о том, имело ли место переобучение.

В дальнейшем, для решения задач классификации будет использоваться метрика точность (accuracy)

$$\mathrm{acc}(\mathrm{T}, \mathrm{h}) = \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} 1\{\mathrm{h}(\mathrm{x_i}) = \mathrm{y_i}\}.$$

Чем выше точность (максимальное значение равно 1), тем лучше гипотеза h соответствует обучающим примерам.

Для решения задач восстановления регрессии будет использоваться метрика средний квадрат отклонения (mean squared error)

$$\mathrm{mse}(\mathrm{T}, \mathrm{h}) = \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} |\mathrm{h}(\mathrm{x_i}) - \mathrm{y_i}|^2.$$

Чем ближе средний квадрат отклонения к нулю, тем лучше гипотеза h соответствует обучающим примерам.

2.1 Линейная регрессия

Линейная регрессия представляет собой наиболее популярный и теоретически исследованный метод решения задачи восстановления регрессии. Этот метод предполагает, что $\mathcal{X} = \mathbb{R}^{\mathrm{n}}$ $(\mathrm{n} \geq 1)$, а гипотезами являются линейные функции вида $\mathrm{h_{\theta,\theta_0}}(\mathrm{x}) = \langle \theta, \mathrm{x} \rangle + \theta_0$. Нахождение требуемой гипотезы осуществляется путем решения задачи минимизации

$$\mathrm{mse}\left(\mathrm{T}, \mathrm{h_{\theta,\theta_0}}\right) \to \min_{\theta,\theta_0}.$$

Эта задача имеет точное аналитическое решение. Однако на практике чаще используется приближенный численный алгоритм под названием метод градиентного спуска. Этот метод вычисляет последовательные приближения к точке минимума целевой функции J. В текущей точке вычисляется направление наибольшего убывания целе-

вой функции, которое совпадает с направлением отрицательного градиента $-\nabla J$. В качестве очередного приближения к точке минимума выбирается точка, лежащая на этом направлении.

Для минимизации среднего квадрата отклонения mse можно напрямую использовать метод градиентного спуска. Однако, если размерность n пространства объектов велика, вычисление градиента будет обладать большой погрешностью. Чтобы устранить это ограничение, используется модификация этого метода, которая называется методом стохастического градиентного спуска.

Идея метода стохастического градиентного спуска состоит в следующем. Целевая функция mse представляет собой сумму неотрицательных слагаемых, каждому из которых соответствует свой обучающий пример. На каждом шаге вычислительного процесса можно случайным образом выбирать подмножество обучающих примеров. Для очередного приближения будет строиться направление убывания не всей целевой функции, а только суммы слагаемых, соответствующих выбранным обучающим примерам.

Целевая функция mse является выпуклой, поэтому применение метода стохастического градиентного спуска будет порождать последовательность приближений, сходящихся к глобальной точке минимума.

Задача восстановления регрессии может быть обобщена. В качестве множества целевых значений может выступать множество $\mathbb{R}^s$ $(s > 1)$. Такую задачу будем называть задачей восстановления многомерной регрессии. Простейший подход к ее решению сводится к решению s отдельных задач восстановления регрессии. Каждой координате целевых значений соответствует своя отдельная задача.

2.2 Логистическая регрессия

Одним из популярных способов решения задачи классификации является метод логистической регрессии. В базовой постановке решается задача бинарной классификации. Предполагается, что $\mathcal{X} = \mathbb{R}^n$ $(n \geq 1)$, $\mathcal{Y} = \{0,1\}$. В качестве гипотез выступают функции вида

$$h_{\theta,\theta_0}(x) = \frac{1}{1 + \exp(-\langle\theta, x\rangle - \theta_0)}.$$

Гипотеза интерпретируется как условная вероятность принадлежности объекта x классу 1, а именно $p(y = 1|x; \theta, \theta_0) = h_{\theta,\theta_0}(x)$ и $p(y = 0|x; \theta, \theta_0) = 1 - h_{\theta,\theta_0}(x)$. Если $p(y = 1|x; \theta, \theta_0) > 0.5$, то считается, что объект x принадлежит классу 1, иначе он принадлежит классу 0.

Для нахождения подходящей гипотезы по множеству обучающих примеров строится функция правдоподобия, для которой в свою очередь решается задача максимизации

$$L(\theta, \theta_0) = \prod_{i=1}^{m} p(y_i|x_i; \theta, \theta_i) \longrightarrow \max_{\theta,\theta_0}.$$

Метод логистической регрессии естественным образом обобщается на случай $|\mathcal{Y}| > 2$.

3 Изменение модели и размерности векторного представления

В данном разделе будет описан алгоритм построения векторного представления текстов, согласованный с заданным векторным представлением слов. Будет проведен анализ результатов тестирования этого алгоритма.

3.1 Построение векторного представления набора текстов

Пусть $\mathcal{T}$ - набор текстов, а $\mathcal{D}$ – словарь, состоящий из всех слов, встречающихся в текстах из набора $\mathcal{T}$. Отображение вида $\tau: \mathcal{T} \longrightarrow \mathbb{R}^n$ будем называть векторным представлением текстов, а отображение вида $\delta: \mathcal{D} \longrightarrow \mathbb{R}^n$ будем называть векторным представлением слов. При этом число n будем называть размерностью векторного представления.

Алгоритм.

Вход: δ_1, δ_2 - векторные представления слов, соответственно размерности n и m; τ_1 - векторное представление текстов размерности n; $\mathcal{R}$ - модель решения задачи восстановления многомерной регрессии (не обязательно линейной).

Выход: τ_2 - векторное представление текстов размерности m.

Начало.

1. Построим $T := \{(\delta_1(w), \delta_2(w))|w \in \mathcal{D}\}$.
2. С помощью модели $\mathcal{R}$ и множества обучающих примеров T построим многомерную регрессию $\rho: \mathbb{R}^n \longrightarrow \mathbb{R}^m$.
3. Положим $\tau_2 := \rho \circ \tau_1$.

Конец.

В основе описанного алгоритма лежит простая идея, согласно которой каждому слову из рассматриваемого словаря сопоставлено по два вектора. Первый вектор получается с помощью исходного векторного представления τ_1, а второй вектор – с помощью целевого векторного представления τ_2. Подобные пары векторов рассматриваются как обучающие примеры, на основе которых строится многомерная регрессия. Полученная регрессия применяется к векторному представлению текстов, согласованному с исходным векторным представлением слов. Можно сделать эвристически обоснованное предположение, что получившееся векторное представление текстов будет согласова-

но с целевым векторным представлением слов. Требуется выработать способ проверки степени обоснованности выдвинутого предположения. Точнее, необходимо выработать подход к оценке качества получившегося в результате применения алгоритма векторного представления текстов.

В рамках предлагаемого подхода качество векторного представления текстов оценивается с точки зрения решения задачи классификации. Поэтому будем предполагать, что набор текстов $\mathcal{T}$ разбит на конечное число попарно непересекающихся классов, а также сформированы тренировочная $\mathrm{T_{train}}$ и тестовая $\mathrm{T_{test}}$ выборки.

Через $\tau(\mathrm{T})$ будем обозначать следующую модификацию множества обучающих примеров T с помощью векторного представления текстов τ. В каждом обучающем примере текст t заменяется на его векторное представление $\tau(\mathrm{t})$.

Зафиксируем некоторую модель $\mathcal{C}$ решения задачи классификации. Тогда векторное представление текстов τ можно оценивать через точность решения задачи классификации текстов в рамках модели $\mathcal{C}$. При этом предполагается, что объектами обучающих примеров являются вектора текстов, полученные с помощью представления τ.

Метод скользящего контроля предписывает одновременно оценивать две величины

$$\mathrm{a_{train}}(\tau) = \mathrm{acc}\big(\tau(\mathrm{T_{train}}), \mathrm{h_{\tau(T_{train})}}\big) \text{ и } \mathrm{a_{test}}(\tau) = \mathrm{acc}\big(\tau(\mathrm{T_{test}}), \mathrm{h_{\tau(T_{train})}}\big),$$

с целью выявления случаев недообучения и переобучения.

Поэтому, если требуется определить, насколько построенное целевое представление текстов τ_2 «лучше» («хуже») исходного представления τ_1, то разумным выглядит подход, когда в качестве соответствующей оценки берутся значения двух величин

$$\mathrm{e_{train}}(\tau_1, \tau_2) = \frac{\mathrm{a_{train}}(\tau_2)}{\mathrm{a_{train}}(\tau_1)} \text{ и } \mathrm{e_{test}}(\tau_1, \tau_2) = \frac{\mathrm{a_{test}}(\tau_2)}{\mathrm{a_{test}}(\tau_1)}.$$

Если построенные величины приблизительно равны единице, то можно считать, что качество исходного и целевого векторного представления одинаково. Если эти величины строго больше (меньше) единицы, то можно считать, что построенное целевое векторное представление лучше (хуже), чем исходное.

Если имеется набор различных моделей решения задачи классификации, то величины $\mathrm{e_{train}}$ и $\mathrm{e_{test}}$ могут быть вычислены для каждой из них. В этом случае будем рассматривать максимальное и среднее значение этих величин.

Далее представим результаты экспериментов над тестовыми наборами данных, которые получены с целью анализа эффективности построенного алгоритма.

3.2 Эксперименты

В ходе проведения экспериментов использовалось два набора тестовых данных[1]. Каждый набор был предварительно разделен на тренировочную и тестовую выборку. Набор movies содержит рецензии к кинофильмам. Каждая рецензия отнесена к одному из шести жанров. Набор состоит из 44012 элементов. Набор twitter состоит из сообщений одноименной социальной сети. Каждое сообщение оценено как позитивное или негативное. Набор состоит из 1596753 элементов.

Для каждого набора данных всегда можно построить тривиальный классификатор. Этот классификатор относит все объекты к одному классу, содержащему наибольшее количество элементов. Точность тривиального классификатора равна отношению размера класса с наибольшим количеством элементов к размеру всего набора данных. Точность тривиального классификатора задает своеобразную нижнюю границу. Классификаторы с меньшей точностью следует рассматривать как неадекватные. Для набора movies точность тривиального классификатора равна 0.48, а для набора twitter равна 0.50.

Для генерации векторных представлений типа Word2Vec и Doc2Vec использовалась библиотека Gensim[2]. Для генерации векторных представлений типа Glove использовалась разработанная авторами этой модели программа[3].

Для набора movies были созданы исходные векторные представления типа DBOW размерностей 50, 100, 200 и 300. При этом создавались векторные представления текстов и согласованные с ними векторные представления слов. В качестве целевых были созданы векторные представления слов типа CBOW, SG и GloVe размерностей 50, 100 и 200. Для набора twitter были созданы векторные представления аналогичных типов, но только размерностей 50 и 100.

В ходе проведения экспериментов использовались модели машинного обучения, реализованные в библиотеке Scikit-Learn[4]. Использовался класс LinearRegression, реализующий модель линейной регрессии и класс MultiOutputRegressor, реализующий модель многомерной регрессии. В качестве модели решения задачи классификации была использована логистическая регрессия, реализованная в классе LogisticRegression. Этот класс имеет набор конфигурационных параметров. В частности, может быть выбран алгоритм решения соответствующей оптимизационной задачи (newton-cg, lbfgs, liblinear, sag, saga), задан генератор псевдослучайных чисел, а также установлен параметр регуляризации. Выбор различных комбинаций значений этих параметров будет приводить к созданию разных классификаторов. В ходе проведения экспериментов рассматривались все предустановленные алго-

[1] Наборы тестовых данных находятся в сети Интернет по адресу https://github.com/group112/se2019
[2] https://radimrehurek.com/gensim
[3] https://nlp.stanford.edu/projects/glove
[4] https://scikit-learn.org

ритмы решения оптимизационной задачи, пять различных начальных значений генератора псевдослучайных чисел и девять различных значений параметра регуляризации в диапазоне $[10^{-5}, 1000]$.

В таблице 1 собраны результаты тестирования алгоритма на наборе данных movies. Строка таблицы соответствует исходному векторному представлению, а столбец - целевому. Таким образом, ячейка соответствует регрессии, отображающей исходное векторное представление в целевое векторное представление. Каждая ячейка содержит пять чисел. Первые четыре числа являются характеристиками построенных классификаторов. Это, соответственно, максимальное и среднее значение величины e_{train}, а также максимальное и среднее значение величины e_{test}. Пятое число – это средний квадрат отклонения построенной регрессии.

Таблица 1. Результаты тестирования алгоритма на наборе данных movies.

Исходное представление DBOW (размерность)	Целевое представление (модель, размерность)								
	CBOW 50	CBOW 100	CBOW 200	SG 50	SG 100	SG 200	GloVe 50	GloVe 100	GloVe 200
50	1.030 1.005 1.033 1.005 0.390	1.031 1.005 1.034 1.005 0.232	1.031 1.006 1.035 1.005 0.129	1.019 1.003 1.021 1.003 0.101	1.023 1.004 1.026 1.004 0.069	1.027 1.005 1.030 1.004 0.046	1.029 1.005 1.032 1.005 0.282	1.030 1.005 1.034 1.005 0.175	1.030 1.005 1.033 1.005 0.099
100	1.033 1.006 1.035 1.005 0.390	1.033 1.006 1.036 1.005 0.232	1.033 1.006 1.036 1.006 0.129	1.030 1.005 1.032 1.005 0.100	1.031 1.006 1.034 1.005 0.069	1.032 1.006 1.035 1.005 0.046	1.032 1.006 1.035 1.005 0.282	1.033 1.006 1.036 1.005 0.175	1.034 1.006 1.036 1.006 0.099
200	1.034 1.006 1.038 1.005 0.389	1.035 1.006 1.037 1.005 0.231	1.035 1.007 1.037 1.005 0.128	1.033 1.006 1.036 1.005 0.100	1.034 1.006 1.037 1.005 0.068	1.034 1.006 1.037 1.005 0.046	1.034 1.006 1.037 1.005 0.281	1.034 1.006 1.037 1.005 0.175	1.034 1.006 1.037 1.005 0.098
300	1.036 1.006 1.039 1.006 0.388	1.036 1.006 1.039 1.005 0.231	1.036 1.007 1.038 1.005 0.128	1.035 1.006 1.038 1.005 0.100	1.035 1.006 1.038 1.005 0.068	1.035 1.007 1.038 1.005 0.046	1.035 1.006 1.037 1.005 0.280	1.035 1.006 1.037 1.005 0.174	1.035 1.007 1.038 1.005 0.098

Построенные классификаторы обладали следующими характеристиками. На тренировочной (тестовой) выборке минимальная, максимальная и средняя точности равны соответственно 0.48, 0.96 и 0.86 (0.48, 0.87 и 0.79). Все эти числа не меньше точности тривиального классификатора. Было принято решение для последующего анализа оставить результаты только «сильных» классификаторов, точность которых превосходит средние значения. Показатели каждой из ячеек со столбцами типа CBOW и SG были сформированы на основе обобщения результатов 300 пар классификаторов, а показатели ячеек со столбцами типа GloVe – 150 пар классификаторов.

Следует обратить внимание на то, что классификаторы, построенные для целевых векторных представлений, обладают строго большей точностью, чем классификаторы, построенные для исходных векторных представлений. Более того, показатели точности зависят только от типа целевого векторного представления и практически не зависят от размерностей исходного и целевого векторных представлений. Этот результат сложно назвать ожидаемым. Построенные целевые векторные представления текстов оказались с точки зрения решения задачи классификации лучше, чем исходные представления.

Неожиданное поведение продемонстрировал также показатель точности регрессии. В каждом столбце он одинаковый. Этот факт означает, что точность регрессии никак не зависит от размерности исходного векторного представления. Если анализировать точность регрессии построчно, то можно заметить, что в границах одного типа целевого векторного представления с ростом его размерности эта точность улучшается.

Результаты тестирования алгоритма на наборе данных twitter, приведенные в таблице 2, в общем, подтверждают результаты, полученные на наборе данных movies. Некоторое отличие состоит в том, что целевые векторные представления обладают приблизительно теми же самыми характеристиками, что и исходные представления. Нет особых улучшений, но и ухудшения незначительны.

Во всех рассмотренных случаях нельзя сделать вывод, что построенные регрессии обладают большой точностью. Оценки их точности лежат в диапазоне [0.046,0.390] для набора movies и в диапазоне [0.052,0.656] для набора twitter. Можно предположить, что с точки зрения решения задачи классификации какую-либо роль играют только старшие разряды (после запятой) значений координат векторных представлений текстов. Поэтому младшие

разряды можно просто отбросить, проведя округления значений координат. Для проверки этой гипотезы была проведена дополнительная серия экспериментов.

Таблица 2. Результаты тестирования алгоритма на наборе данных twitter.

Исходное представление DBOW (размерность)	Целевое представление (модель, размерность)					
	CBOW 50	CBOW 100	SG 50	SG 100	GloVe 50	GloVe 100
50	1.002 1.000 1.003 1.000 0.657	1.003 1.000 1.003 1.000 0.394	1.000 0.999 1.000 0.999 0.076	1.001 1.000 1.001 1.000 0.052	1.001 1.000 1.001 1.000 0.204	1.003 1.000 1.003 1.000 0.118
100	1.004 1.000 1.004 1.000 0.656	1.004 1.001 1.004 1.001 0.394	1.003 1.000 1.003 1.000 0.076	1.004 1.000 1.004 1.000 0.052	1.003 1.000 1.003 1.000 0.204	1.004 1.000 1.004 1.001 0.118

Если τ - векторное представление текстов, а n - натуральное число, то через τ_n будем обозначать векторное представление текстов, полученное из τ путем округления значений его координат. В результате округления остаются только n первых цифр после запятой. Качество получившегося векторного представления можно оценить с помощью двух параметров $e_{train}(\tau, \tau_n)$ и $e_{test}(\tau, \tau_n)$.

В таблице 3 приведены результаты округления целевых векторных представлений типа GloVe, созданных для набора данных twitter. Округления проводились до 1, 3 и 5 цифр после запятой. Как и прежде, строка таблицы соответствует исходному векторному представлению, а столбец соответствует целевому векторному представлению и параметру округления. Каждая ячейка содержит четыре числа: максимальное и среднее значение величины $e_{train}(\tau, \tau_n)$, а также максимальное и среднее значение величины $e_{test}(\tau, \tau_n)$. Как видно, округление до 3 и 5 цифр после запятой не изменило качество векторного представления текстов, а округление до 1 цифры слегка его ухудшило.

Отметим, что результаты округления других типов целевых векторных представлений для набора данных twitter, а также целевых векторных представлений для набора данных movies дали такие же результаты.

Таблица 3. Результаты округления координат целевого векторного представления текстов для набора twitter.

Исходное представление DBOW (размерность)	Целевое представление GloVe (размерность, округление)					
	50, 1	50, 3	50, 5	100, 1	100, 3	100, 5
50	0.988 0.986 0.988 0.986	1.000 1.000 1.000 1.000	1.000 1.000 1.000 1.000	0.993 0.991 0.993 0.992	1.000 1.000 1.000 1.000	1.000 1.000 1.000 1.000
100	0.997 0.996 0.997 0.996	1.000 1.000 1.000 1.000	1.000 1.000 1.000 1.000	0.998 0.996 0.997 0.995	1.000 1.000 1.000 1.000	1.000 1.000 1.000 1.000

Заключение

В настоящей статье был представлен подход к формированию векторного представления текстов, согласованного с заданным векторным представлением слов. Качество векторного представления текстов определяется точностью решения задачи классификации текстов, которую можно достигнуть, используя это векторное представление.

В основу подхода была положена идея трансформации векторного представления слов, согласованного с заданным векторным представлением текстов. Подобная трансформация включает изменение модели и размерности векторного представления. Ее целью может служить желание использовать векторные представления, более адекватно описывающие предметную область, а также желание уменьшить их размерность, которая напрямую влияет на затраты по их хранению и обработке. Рассматриваемые трансформации векторных представлений реализуются в виде решения задачи восстановления многомерной регрессии.

Полученные в ходе проведение экспериментов результаты показали эффективность предложенного подхода. Качество построенных целевых векторных представлений текстов оказалось не хуже, а в большинстве случаев лучше исходных.

Был получен ряд экспериментальных результатов, касающихся точности построенных регрессий векторных представлений. Некоторые из них оказались неожиданными и требующими дальнейшего осмысления. Например, оказалось, что точность регрессии не зависит от размерности исходного векторного представления, а определяется только типом и размерностью целевого векторного представления.

Относительная невысокая точность регрессий навела на мысль, что координаты векторных представлений могут быть округлены. Это не приводит к ухудшению качества векторных представлений с точки зрения решения задачи классификации, однако позволяет значительно уменьшить их размер.

Полученные результаты свидетельствуют о том, что методы решения задачи восстановления многомерной регрессии могут успешно использоваться для формирования векторных представлений текстов с улучшенными характеристиками. В настоящей работе предполагалось, что множество текстов фиксировано. Дальнейшее развитие предложенного подхода будет связано со случаем, когда исходное множество текстов может быть расширено. Тем самым, будет рассматриваться задача интерполяции векторного представления текстов.

Список использованной литературы

[1] Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector, Computing Research Repository (CoRR), 2013, pp. 1-12, available at: https://arxiv.org/abs/1301.3781.

[2] Mikolov T., Sutskever I., Chen K., Corrado G., Dean J. Distributed representations of words and phrases and their compositionality, Proceedings of the 26th International Conference on Neural Information Processing Systems, 2013, vol. 2, pp. 3111-3119.

[3] Pennington J., Socher R., Manning C.D. GloVe: Global Vectors for Word Representation, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532-1544.

[4] Sebastiani F. Machine learning in automated text categorization, ACM Computing Surveys, 2002, vol. 34, no. 1, pp. 1-47.

[5] Kim Y. Convolutional Neural Networks for Sentence Classification, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1746-1751.

[6] Le Q., Mikolov T. Distributed Representations of Sentences and Documents, Proceedings of the 31st International Conference on Machine Learning, 2014, vol. 32, no. 2, pp. 1188-1196.

[7] Mu J., Bhat S., Viswanath P. All-but-the-Top: Simple and Effective Post-processing for Word Representations, Computing Research Repository (CoRR), 2018, pp. 1-25, available at: https://arxiv.org/abs/1702.01417.

[8] Bishop C.M. Pattern Recognition and Machine Learning, Springer, Science+Business Media LLC, 2006, 738 p.

[9] Harris Z. Distributional structure, Word, 1954, vol. 10, no. 23, pp. 146-162.

[10] Rong X. word2vec Parameter Learning Explained, Computing Research Repository (CoRR), 2016, pp. 1-21, available at: https://arxiv.org/abs/1411.2738.