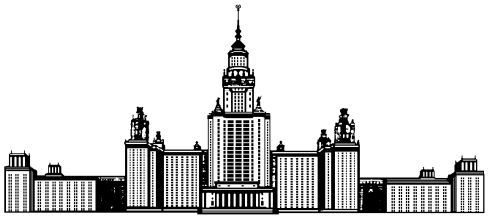


Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Попов Артём Сергеевич

Выделение множества тематик в неразмеченной коллекции диалогов

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Научный руководитель:
д.ф.-м.н., доцент, профессор РАН
К.В. Воронцов

Москва, 2019

Содержание

1 Введение

В последнее время широкое распространение получили диалоговые системы. Такие системы используются, например, в контактных-центрах для автоматизации выдачи ответов на типичные вопросы пользователя. Для обучения таких систем нужна большая размеченная выборка реплик пользователя по категориям запросов.

При работе с новой коллекцией обычно неизвестны категории запросов, более того, обычно даже неизвестно их количество. Так как над созданием размеченной выборки обычно работает сразу несколько человек, велика вероятность того, что созданная ими разметка будет в итоге неудобна для машинного обучения. Например, классы будут сильно пересекаться между собой, для вопросов из разных областей будет сильно различаться степень детализации классов.

Для упрощения этого процесса, можно произвести предварительное выделение устойчивых тем в коллекции, не используя размеченную выборку. Под темой будем понимать набор документов и ключевых токенов (слов, словосочетаний и т.д.). В дальнейшем, асессорам будет необходимо только лишь подкорректировать выделенные темы (например, удалив некорректные документы). Предварительная тематизация позволит получить первичный набор классов, позволит в начале разметки синхронизировать и упростить работу асессоров.

В «идеальной» модели, все документы, соответствующие конкретной теме, должны соответствовать одному конкретному запросу пользователя. Понятие «запроса пользователя» очень сложно формализовать, поэтому почти невозможно корректно сравнивать качество работы разных моделей выделения тем, используя только неразмеченные данные. В данной работе предлагается использовать для оценки моделей выборку мер близости пар диалогов. Такую разметку легко и быстро составить, даже не зная структуру классов, один элемент выборки — оценка близости двух диалогов. Так как эта разметка будет использовать только для оценивания, но не для обучения коллекции, количество примеров в ней не обязано быть слишком большим.

В данной работе предложены способы построения моделей, решающие задачу максимизации качества моделей выделения тематик по множеству размеченных пар диалогов. В работе показано преимущество иерархического тематического моделирования над другими подходами. В работе показано как можно улучшить качество модели, комбинируя различные техники анализа текстов, такие как выделение именованных сущностей, исправление опечаток, выделение коллокаций, выделение частей речи. Для ис-

правления опечаток и выделения коллокаций предложены модификации, повышающие качество решения задачи. В работе предложен способ повышения качества решения задачи, использующий разделение токенов коллекции по группам на разных уровнях иерархической тематической модели.

2 Способы кластеризации документов

Задача выделения тем в коллекции тесно связана с задачей кластеризации. В этом разделе будут рассмотрены, использующиеся в данной работе, способы построения кластеризации на коллекции текстовых документов.

2.1 Кластеризация на моделях векторных представлений

Самый простой способ построения модели кластеризации на коллекции текстовых документов является двухэтапным. На первом этапе каждому документу ставится в соответствие вещественный вектор в некотором пространстве R^m. На втором этапе к построенным представлениям применяется один из стандартных алгоритмов кластеризации.

Существует много способов построить векторное представление документа. Наиболее простой — использовать tf-idf представление. Каждому документу ставится в соответствие вектор длины $|W|$, каждая компонента w вектора вычисляется по формуле $\text{tf-idf}(w, d, D) = \text{tf}(w, d) \times \text{idf}(w, D)$ [1].

В задаче классификации текстов логистическая регрессия на tf-idf представлении является достаточно сильным алгоритмом, качество которого зачастую сложно побить даже с помощью использования глубоких нейронных сетей. В то же время, в задаче кластеризации использование tf-idf представления напрямую приводит к плохим результатам. В большинстве моделей кластеризации используются функции расстояния/близости между объектами в пространстве. Из-за «проклятия размерности» качество вычисления близости на основе tf-idf представления будет недостаточно хорошим.

Для повышения качества кластеризации можно воспользоваться методами понижения размерности, чтобы отобразить tf-idf представление в плотное пространство более

[1] определение tf-idf (википедия)

низкой размерности. Например, можно использовать классический алгоритм PCA[2] или Uniform Manifold Approximation and Projection (UMAP) [11].

Другой популярный способ построения представлений — использование векторных представлений слов (word embeddings). В этом подходе сначала каждому слову ставится в соответствие вектор из пространства $\mathbb{R}^m$ (где $m \ll |W|$), затем представление документа вычисляется как взвешенная сумма представлений входящих в него слов.

Самыми популярными моделями векторных представлений слов являются модели семейства word2vec [3]: CBOW, Skip-gram и их модификации [4].

Все эти модели основаны на гипотезе дистрибутивности [6] — слова, встречающиеся в схожих контекстах, являются близкими по смыслу. Модель CBOW обучает вектора в ходе решения задачи предсказания слов по их контекстам (под контекстом понимается окрестность слова из k слов). Модель Skip-gram обучается предсказывать каждое из слов контекста на основе центрального слова. Модели обучаются с помощью стохастического градиентного спуска, на практике для ускорения обучения применяются модификации моделей, использующие иерархический softmax (hierarchical softmax) или негативное семплирование (negative sampling).

Для обучения моделей представлений необходимо использовать большую коллекцию документов (например, википедию). Для улучшения качества можно дополнительно дообучить полученные представления на коллекции, которую необходимо кластеризовать. Как и в случае tf-idf представления, использование методов снижения размерности на практике увеличивает качество решения задач кластеризации.

В качестве функции агрегации представлений слов для получения представлений документа можно использовать среднее (равномерный вклад всех слов в документ) или взвешенную сумму представлений слов в документе с нормированными idf весами (более редкие слова имеют больший вклад чем частые).

$$v_d = \frac{1}{n_d} \sum_{w \in d} v_w$$

$$v_d = \frac{1}{\sum_{w \in \text{set}(d)} \text{idf}(w)} \sum_{w \in \text{set}(d)} \text{idf}(w) v_w$$

[2]определение PCA (википедия)

[3]название программного продукта, в котором были реализованы упомянутые выше модели

2.2 Тематическое моделирование

Тематическое моделирование — подход к обработке текстовых коллекций, позволяющий одновременно получить представления для сущностей внутри коллекции (например, слов) и мягкую кластеризацию документов.

Предполагается, что появление каждого термина w в документе d связано с некоторой скрытой переменной t из конечного множества тем T. Тогда коллекция D представляет собой выборку троек (d, w, t), взятых независимо из дискретного распределения $p(d, w, t)$ на множестве $D \times W \times T$. С учётом гипотезы условной независимости [7] и формулы полной вероятности, а также обозначений $p(w|t) = \phi_{wt}$ и $p(t|d) = \theta_{td}$, вероятность появления слова в документе в тематической модели записывается в виде:

$$p(w|d) = \sum_{t \in T} p(w|t)p(t|d) = \sum_{t \in T} \phi_{wt}\theta_{td} \tag{1}$$

Простейшей вероятностной тематической моделью является модель PLSA [7]. В PLSA для построения модели (1) максимизируется логарифм правдоподобия при ограничениях нормировки и неотрицательности:

$$L(\Phi, \Theta) = \sum_{d \in D} \sum_{w \in W} n_{dw} \ln \sum_{t \in T} \phi_{wt}\theta_{td} \;\to\; \max_{\Phi, \Theta} \tag{2}$$

$$\sum_{w \in W} \phi_{wt} = 1,\ \phi_{wt} \geq 0; \quad \sum_{t \in T} \theta_{td} = 1,\ \theta_{td} \geq 0 \tag{3}$$

Стандартным методом решения этой задачи является EM-алгоритм — итерационный процесс, состоящий из двух шагов: E-шаг (expectation) и M-шаг (maximization). На E-шаге по текущим параметрам ϕ_{wt} и θ_{td} (начальное приближение — нормированные неотрицательные случайные вектора) вычисляются вероятности $p(t|d, w)$ для всех $t \in T$, $w \in W$, $d \in D$. На M-шаге при фиксированных вероятностях $p(t|d, w)$ вычисляются новые приближения для параметров ϕ_{wt} и θ_{td}.

Задача тематического моделирование является некорректно поставленной (имеет бесконечное множество решений). Общим методом решения таких задач является регуляризация — введение разумных дополнительных ограничений на решение задачи.

Большинство способов обобщения модели PLSA основано на использовании байесовских методов. Самой известной модификацией является модель LDA [2], в основе которой лежит предположение, что векторы документов $\theta_d \in \mathbb{R}^{|T|}$ и векторы тем $\phi_t \in \mathbb{R}^{|T|}$ порождаются из распределения Дирихле. Обучение модели заключается в поиске апостериорных распределений на параметры Φ и Θ.

Модель ARTM [13] является небайесовским обобщением модели PLSA. Для построения модели (1) максимизируется сумма логарифма правдоподобия и r дополнительных критериев $R_i(\Phi, \Theta)$, называемых регуляризаторами, с коээфицентами регуляризации τ_i.

Ещё более общей моделью является мультимодальная ARTM[14], позволяющая учитывать различные модальности, встречающиеся в документе (например, время написания или автора). Пусть M — множество модальностей. Для каждой модальности строится своя матрица Φ^m, $\Phi = \bigcup\limits_{m \in M} \Phi^m$. Итоговый оптимизируемый функционал строится как сумма взвешенных с коеффициентами α_m логарифмов правдоподобий для каждой модальности и регуляризаторов:

$$L(\Phi, \Theta) = \sum_{m \in M} \alpha_m \sum_{d \in D} \sum_{w \in W^m} n_{dw} \ln \sum_{t \in T} \phi_{wt} \theta_{td} + \sum_{i=1}^{r} \tau_i R_i(\Phi, \Theta) \to \max_{\Phi, \Theta} \tag{4}$$

$$\sum_{w \in W} \phi_{wt} = 1,\ \phi_{wt} \geq 0; \quad \sum_{t \in T} \theta_{td} = 1,\ \theta_{td} \geq 0 \tag{5}$$

Эта задача также решается с помощью EM-алгоритма, изменения по сравнению с PLSA претерпевают формулы M-шага.

3 Различные аспекты моделей анализа текстов

В этом разделе будут рассмотрены различные задачи анализа текстов, решение которых предположительно может влиять на качество решения задачи выделения тематик в коллекции диалогов. Будут рассмотрены модификации различных алгоритмов, предложенные в данной работе.

3.1 Модели выделения n-грамм

Стандартный способ повышения качества тематической модели — использование в модели помимо слов n-грамм или коллокаций[4]. Существую различные способы выделения коллокаций, основанные на подсчёте различных статистик по коллекции.

В данной работе для выделения коллокаций использовался алгоритм TopMine [12]. Алгоритм работает двухэтапно: на первом этапе (обучение) подсчитываются статистики совстречаемости слов в коллекции, на втором этапе (выделение коллокаций) для каждого предложения производится итеративное слияние слов в коллокации.

[4]словосочетания, имеющее признаки синтаксически и семантически целостной единицы

У алгоритма TopMine есть несколько недостатков, которые могут существенно сказываться при работе с не самыми длинными документами (а могут легко встретиться в коллекции диалогов).

Во-первых, исходный алгоритм рассматривает как уникальную единицу последовательность слов $(w_1, w_2, \ldots, w_k)$. Последовательности (w_1, w_2) и (w_2, w_1) будут для алгоритма абсолютно разными. Особенность русского языка — недетерминированный порядок слов, поэтому для русского языка имеет смысл использовать в качестве уникальной единицы алгоритма не последовательность слов, а мультимножество слов.

Во-вторых, TopMine позволяет выделить только лишь непересекающиеся коллокации и только жадным способом. Например, встретив в документе последовательность «метод опорных векторов», алгоритм выделить коллокацию из трёх слов, но никак не учтёт коллокацию опорные вектора». Например, предложения «получение паспорта РФ» (выделится коллокация из всех трёх слов) и «паспорт РФ был утерян» (выделится коллокация из первых двух слов) могут вообще не содержать общих коллокаций.

Чтобы обойти это ограничение, небходимо изменить процедуру жадного поиска. Вывод в алгоритме TopMine предполагает итеративное слияние фраз: изначально все фразы — слова предложения, затем на каждой итерации две ближайшие фразы сливаются в одну. Вместо этого предлагается другая процедура: на каждой итерации в множество известных фраз алгоритм пытается добавить новое слово, если слово и фраза удовлетворяют критерию, новая фраза добавляется в множество, однако старая не удаляется.

В данной секции приведены псевдокоды для каждого из этапов алгоритма с учётом сделанных модификаций (алгоритмы 1 и 2). Функция is_token при обучении алгоритма возвращает `True`, если поданный в функцию токеном является словом. Это необходимо для того, чтобы не объединять в n-граммы знаки препинания. Функция Score в алгоритме вывода оценивает необходимость слияния n-граммы и следующего за ней слова и явно использует собранную статистику совстречаемостей слов. Способы задания этой функции описаны в статье [12].

Заметим, что после приведённого алгоритма разумно добавить некоторую фильтрацию получившегося множества. Например, можно удалить «вложенные» друг в друга коллокации, но оставить пересекающиеся. При экспериментах из получившихся n-грамм удалялись «крайние» стоп-слова (таким образом, в модели не учитывались n-граммы типа «на столе», «в банке»). При вычислении функции `significance_score` можно использовать не только частоты появлений слов в коллекции, но и информации о частях речи, именованных сущностей и т.п. В данной работе, такой подход не рассмат-

Algorithm 1 Сбор статистики совстречаемости токенов

Вход: D — множество документов; k — максимальная длина n-граммы; t — порог для хранения n-граммы;

Выход: S — словарь, содержащий количество совстречаемостей для каждой n-граммы;

$A := \{\}$ // *словарь возможных позиций для каждого документа*
для всех $w \in W$:
  $S[w] := 0$; // *инициализация*
для всех $d \in D$:
  $A[d] := [0, \ldots, \text{len}(d) - 1]$
  для всех $w \in d$:
    если is_word(w) **то**
      $S[w] := S[w] + 1$; // *счётчик не увеличивается, если токен — не слово*
для $n := 2, \ldots, k$:
  для всех $d \in D$:
    если $len(A[d]) := 0$ **то**
      продолжить
    $a := [\,]$
    для всех $i \in A[d]$:
      // *multiset преобразует последовательность в мультимножество*
      $g := \text{multiset}(d[i : i + n - 1])$ // *не включает $i + n - 1$ символ*
      если $len(d) - i \geqslant n - 1$ **и** $S[g] \geqslant t$ **то**
        $a := a + [i]$ // *конкатенация*
        если $i - 1 \in a$ **то**
          $g := \text{multiset}(d[i : i + n])$
          $S[g] := S[g] + 1$
    $A[d] = a$

ривался, однако после выделения проводилась дополнительная группировка n-грамм по разным группам (секция 4.3).

Также заметим, что в качестве предложений необязательно подавать все исходные документы. Например, можно подавать только «ключевые» фразы, либо генерировать синтетические документы только из синтаксически связанных слов.

Algorithm 2 Сбор статистики совстречаемости токенов

Вход: d — документ; t_1, t_2 — пороги для выделения n-грамм;

Выход: G — список n-грамм документа d;

$H = \{\ \}$ // *словарь, содержащий для н-граммм значения критерия слияния*
$h = \{\ \}$
для $i := 0, \ldots, \text{len}(d) - 2$:
    $h[[i]] = 0$
для $n := 2, \ldots, k$:
    new_h $= \{\}$
    для всех $j \in h.keys()$:
        если $j[-1] + 1 \geqslant \text{len}(d)$ **то**
            продолжить
            $v = Score(d[j[0] : j[-1]], d[j[-1] + 1])$
        если $v \geqslant t_1$ **то**
            new_h$[j + [j[-1] + 1]] = v$
        $h =$ new_h
        $H = H \cup h$
$G = \{\ \}$
для $j \in H.keys()$
    если $H[j] \geqslant t_2$ **то**
        G.add$(d[j])$ // $d[j]$ — *соответствующая индексам* j *n-грамма*

3.2 Модели выделения именованных сущностей

В диалогах между клиентом и оператором часто возникают упоминания имён друг друга, названий компаний/каких-то продуктов, упоминания улиц, городов и т.д. Для повышения качества моделирования диалога может иметь смысл некоторые сущности учитывать особенно (например, в тематической модели можно использовать именованные сущности как модальности), а некоторые просто удалить или заменить на специальный тег (например, все имена заменить на тег <name>).

В зависимости от типа сущностей, хорошее качество у тех или иных алгоритмов. Существуют подходы основанные на правилах [9], машинном обучении [10]. Для популярных сущностей (имена, геолокации) state-of-the-art моделью является комбинация рекуррентной нейронной сети и случайных марковских полей (RNN + CRF) [1].

Заметим, что для большинства подходов требуется обучающаяся выборка, не совпадающая с исходной, что может вызывать трудности при применении моделей на новых коллекциях.

3.3 Модели исправления опечаток

В диалогах между клиентом и оператором часто встречаются опечатки и ошибки (со стороны клиента). Существует два подхода адаптации моделей к таким коллекциям: применение алгоритма исправления опечаток или внесении в модель информации не только на уровне слов, но и на уровне частей слов.

Основная схема для алгоритма исправления опечаток включает в себя два этапа: построение кандидатов на исправление и выбор наилучших кандидатов. На первом этапе для каждого слова в предложении генерируются «кандидаты», например, с помощью удаления из слова определённых букв. На втором этапе, с помощью языковых моделей или обученного классификатора из полученных предложений выбирается наилучшее.

Заметим, что целевое использование, описанной в этой работе модели — построение end-to-end системы, принимающей на вход коллекции диалогов, выдающую список тем коллекции. Таким образом, любые алгоритмы предобработки должны работать достаточно быстро, так чтобы весь процесс укладывался в несколько часов.

Один из самых быстрых алгоритмов исправления опечаток — алгоритм Jamspell[5]. Jamspell работает по описанной выше схеме, генерирует кандидатов с помощью замены символов на другие, удаления и добавления символов. В качестве кандидатов выбираются слова, которые имеют ненулевую вероятность в языковой модели. На втором этапе используется триграммная языковая модель. Для хранения триграмм используются фильтр Блума и perfect hash, что делает алгоритм достаточно быстрым.

Одной из слабых сторон Jamspell считается его плохое качество при исправлении «реальных» ошибок, а не синтетических. Для адаптации модели к задаче было предложено несколько модификаций.

Во-первых, языковая модель, использующаяся для выбора лучших кандидатов, должна быть обучена по коллекции, на которой применяется модель. Такая модель будет учитывать специфику коллекции, но может иногда выбирать ошибочные варианты исправления (т.к. в исходной коллекции, очевидно, содержатся слова с опечатками). Основное предположение, обосновывающее эту модификацию, то что в коллекции слов

[5]ссылка github

с опечатками и ошибками меньше чем правильных слов. При этом, в коллекции могут содержаться слова, которых вообще не было в исходном словаре (например, специфичные аббревиатуры).

Во-вторых, можно расширить множество генерируемых кандидатов. Так, одна из самых популярных ошибок в коллекции обращений — «слепление» двух слов в одно («чтоделать»). Также можно добавить ошибки по фиксированному словарю ошибок (например, «здраствуйте» в «здравствуйте»), а также учесть фонетические особенности языка (например, «чаво» в «чего»). Заметим, что изменение генератора почти не влияет на итоговую сложность модели, т.к. основные вычислительные трудности связаны с выбором лучшего кандидата (однако, нужно следить, чтобы кандидатов не становилось слишком много).

Второй, принципиально другой способ работы с опечатками — интеграция в модель посимвольной информации. Например, в модели fasttext [5], векторное представление слова составляется как сумма векторных представлений его буквенных триграмм.

Аппарат модальностей в тематическом моделировании позволяет учитывать одновременно и слова, и триграммы как разные модальности в одной модели. Заметим, что тогда модель может стать чуть менее интепретируемой, зато получит некоторую устойчивость к опечаткам.

Ещё более серьёзная модификация — использование для учёта буквенных триграмм гиперграфовой тематической модели[6]. Каждое ребро гиперграфа будет представлять собой документ, входящее в него слово, входящие в слово буквенные триграммы [8]. В такой модели (в отличии от стандартного подхода) есть явная связь слова с буквенными триграммами, из которых оно состоит.

4 Задача выделения тематик в диалогах

В этом разделе будет рассмотрена более подробно постановка задачи и предложенные способы её решения.

4.1 Постановка задачи

Целью данной работые является создание модели, выделяющей в коллекции диалогов интерпретируемые структуры — темы. Тема задаётся некоторым набором токенов

[6]магистерская диссертация «Многомодальные тематические модели на гиперграфах»

и диалогов. «Идеальная» модель должна выделять такие темы, что все диалоги внутри темы будут соответствовать одному и тому же запросу (intent) пользователя.

Заметим, что понятие запроса сложноформализуемо, например близкие диалоги «как мне записаться к врачу» и «не могу записаться к врачу» соответствуют разным запросам пользователя. Встречаются ситуации, когда одинаковым ответам оператора, соответствуют разные запросы пользователя.

Поэтому, для оценивания качества модели, обучающейся без учителя (unsupervised learning), необходимо использовать размеченные данные. В данной задаче, использовалось множество размеченных пар диалогов. Заметим, что такой подход имеет достаточно много плюсов. Например, с помощью такой разметки можно оптимизировать гиперпараметры (в том числе количество тем). Три асессора оценили каждую пару из указанных множеств по следующей шкале:

- 0 — между диалогами нет ничего общего
- 1 — в диалогах речь идёт о близких понятиях, но запрос различный
- 2 — диалоги имею один и тот же запрос пользователя
- ? — у одного из двух диалогов невозможно определить запрос пользователя

Учитывая наличие описанной выше разметки логично будет ставить задачу максимизации точности модели по полученной разметке.

4.2 Моделирование иерархической структуры

Основная сложность описанной выше задачи — необходимость различать ситуации «1» и «2». Для учёта такой градации будем использовать двухуровневую модель кластеризации. Для любых двух диалогов мы можем ввести следующие оценки:

- 0 — диалоги находятся в разных кластерах верхнего уровня
- 1 — диалоги находятся в одном кластере верхнего уровня, но в разных кластерах нижнего уровня
- 2 — диалоги находятся в одном кластере нижнего уровня
- ? — после этапа предобработки документ оказался пустым

Такая формализация позволяет легко сопоставлять два списка оценок пар диалогов, и легко измерять долю правильных ответов модели.

Для оценивания первого уровня кластеризации модели можно использовать стандартные метрики accuracy, precision, recall, f1-меру по спискам, в которых 1 и 2 считаются эквивалентными. Для оценивания второго уровня модели при фиксированном первом, можно использовать те же стандартные метрики, не учитывая класс 0 (так как согласно процедуре выставления оценки, выставление оценки 0 зависит только от кластеризации на первом уровне). При вычислении precision и recall класс 1 считается негативным ответом, класс 2 — позитивным.

Единственный вопрос, который остаётся решить — способ построения двухуровневой кластеризации. Один из самых простых способов — двухэтапная кластеризация. Сначала производится кластеризация множества (это будут кластера первого уровня), затем каждый кластер трактуется как отдельная колллекция и производится кластеризация каждого кластера в отдельности (это будут кластера второго уровня). Проблема такого подхода — необходимость явно задавать число кластеров, на которые необходимо разбить каждый кластер. Аналогично, можно произвести кластеризацию множества на большое число кластеров (это будут кластера второго уровня), а затем объединить некоторые из них в большие кластера (это будут кластеры первого уровня).

Ещё один способ — использование иерархического тематического моделирования. Подход описанный в [3] также является двухэтапным. Сначала строится тематическая модель для первого уровня иерархии, затем строится другая тематическая модель так, чтобы каждая из тем первой модели являлась линейной выпуклой комбинацией тем второй модели. Формально это можно записать через регуляризатор матрицы Φ:

$$R(\Phi) = -\sum_{t \in T_1} n_t \, \mathrm{KL_w}\left(p(w|t) || \sum_{\tau \in T_2} p(w|\tau) p(\tau|t)\right)$$

При таком подходе не нужно задавать число тем для каждой темы первого уровня, необходимо только задать общее число тем на втором уровне. В данной работе, в основном, развивается подход иерархического тематического моделирования. Это снова связано с тем, что предполагаемое использование модели из этой работы — построение end-to-end системы. Такая система должна хорошо работать с коллекцией из любой области. Работа моделей, основанных на предобученных представлениях, зависит от того, насколько хорошо предобученные представления соответствуют поступившей на вход коллекции. Например, если модель представлений была обучена по википедии, то для коллекции содержащей большое количество технических аббревиатур, для большин-

ства важных слов просто не будет векторного представления. Для повышения качества в таких ситуациях, модели можно дообучать на входной коллекции, но это выходит за рамки данной работы.

4.3 Повышение качества второго уровня иерархии

Построение двухуровневой кластеризации — сложная задача. Алгоритмы кластеризации данных — грубые, при проведение плоской стандартной кластеризации, в один кластер объединяются объекты хоть как-то друг на друга похожие или сильно отличающиеся от других.

Построение второго уровня кластеризации предполагает нахождение каких-то различий в похожих объектах (выделенных с помощью кластеризации первого уровня), причём примерно таким же способом как и на первом уровне. При кластеризации второго уровня часто выделяются уменьшенные «копии» кластеров первого уровне, в то время как хотелось бы находить в документах, входящих в один кластер, какие-то различия.

Основной способ борьбы с этим явлением, предложенный в данной работе — использование на разных уровнях модели разного набора модальностей. В токенах модели было выделено две группы с помощью анализа по частям речи: «тематичные» и «функциональные». Функциональные токены — любые, содержащие глаголы. Тематичные токены — любые, не содержащие глаголы, и содержащие хотя бы одно существительное или прилагательное. Будем называть «основной» группой все токены.

Такое разделение признаков придумано исходя из постановки задачи, «функциональные» токены слабо влияют на тематику коллекций и не нужны при построении первого уровня иерархии. «Тематичные» токены слабо влияют на определение запроса пользователя, гораздо большую роль играют токены, означающие действия.

Таким образом, на первом уровне иерархии большой вес должны иметь «тематичные» токены, на втором — «функциональные». Основные токены нужны на обоих уровнях для связи разных уровней иерархий между собой (иначе получались бы две независимые кластеризации).

Заметим, что использование разных модальностей на разных уровнях — не единственный способ борьбы с проблемой кластеризации на втором уровне. Было проведено несколько экспериментов по использованию на втором уровне «фиктивных» тем, каждая из которых была похожа на тему первого уровне. Этого можно достигнуть с

помощью задания структуры иерархии (у «фиктивной» темы второго уровня только один родитель — соответствующая тема первого уровня) и сглаживания документов (для каждого документа повышается вероятность встретить его в «фиктивной» теме соответствующей его теме на первом уровне). При такой схеме легко «переобучить» модель копировать структуру первого уровня, никак не внося изменения на втором. Построение такой структуру требует использования регуляризаторов декоррелирования, сглаживания предметных тем, причём коэффициенты регуляризаторов желательно подбирать отдельно для каждой модальности. Эксперименты показали, что такая модель не позволяет получить прироста в качестве, поэтому в данной работе она детально рассмотрена не будет.

5 Вычислительные эксперименты

Для тестирования различных моделей были использованы 2 коллекции диалогов различных call-центров. В первой коллекции — 95520 диалога, во второй коллекции — 90789 диалогов. Каждый диалог ведётся между двумя людьми: оператором и клиентом. Средняя длина диалога — 6 реплик с двух сторон суммарно. Качество моделей оценивается с помощью размеченного асессорамиы набора пар диалогов.

Для первой коллекции было размечено 13848 пар обращений (они были разделены на две группы: 122247 пар и 1601 пара), для второй коллекции была размечено 1601 пара обращений.

Подбор гиперпараметров модели производился на «большой» размеченной выборке для 1-ой коллекции (далее в таблицах и графиках 1-1). Для измерения качества моделей на «маленькой» размеченной выборке для 1-ой коллекции (далее 1-2) и выборке на 2-ой коллекции (далее 2-1) использовались гиперпараметры, полученные для 1-1 разметки. Такая процедура оценивания была выбрана для того, чтобы избежать эффекта переобучения из-за оптимизации гиперпараметров под выбранную метрику.

Для обеих коллекций использовались одинаковые процедуры общей предобработки. Например, удалялись реплики, которые соответствуют хотя бы 2-ум из 3-ёх критериев:

- Многострочность: реплика состоит больше чем из 6 строк / 10 слов, и число строк хотя бы в полтора раза больше, чем количества знаков препинания. Это правило выделяет копирование содержимого веб-страницы.

- Пропорция одинаковых слов: реплика состоит больше чем из 10 слов, и её длина в словах более чем в 2.5 раза превосходит количество уникальных слов. Это правило выделяет реплики, содержащие многократное повторение одной и той же фразы с небольшими изменениями.

- Скорость убывания частоты слов: сравниваются частоты трёх самых распространённых слов внутри реплики (не учитывая при этом предлоги и другие распространённые слова). Это правило выделяет реплики, непохожие на естественную речь.

Помимо этого, все слова в коллекции приводились в нормальную форму, из коллекции удалялись стоп-слова и стоп-выражения, некоторые простые сущности (адреса электронной почты, html-теги, названия сайтов) заменялись на соответствующий тег.

Коллекция строилась на документах, являющихся конкатенациями реплик пользователя одного диалога. Были попытки учесть реплики оператора при построении модели, однако они не давали никакого прироста качестве по причине низкого качества ответов оператора.

5.1 Базовые модели

В качестве первого эксперимента рассмотрим несколько базовых моделей, описанных выше: тематические модели PLSA и LDA, кластеризации K-means над tf-idf представлениями, сжатыми с помощью PCA, и K-means над представлениями, полученными с помощью усреднения эмбеддингов skip-gram (предобученных на русской википедии) и применения алгоритма umap.

Для начала, оценим качество построения 1-ого уровня кластеризации без учёта второго уровня (таблица 1). Алгоритмы сравнивались по accuracy и f-мере, указанным выше способом. Как и ожидалось, kmeans над tf-idf представлениями работает хуже других подходов. Качество тематических моделей сильно превосходит модели кластеризации над эмбеддингами, что можно объяснить тем, что входящие диалоги достаточно длинные, взвешенная сумма вещественных представлений слов получается достаточно «размытым» представлением диалога.

Для всех алгоритмов выбиралось оптимальное (по метрике accuracy в ситуации, когда 1 и 2 считаются одним классом) число кластеров (от 10 до 30), для алгоритмов уменьшения размерности подбиралась итоговая размерность представлений, для эмбеддингов подбиралась стратегия агрегации векторов слов (среднее или tf-idf веса). В

	1-1		1-2		2-1	
	acc.	f1	acc.	f1	acc.	f1
PLSA	0.714	0.690	0.678	0.747	0.772	0.672
ARTM	**0.762**	**0.732**	**0.754**	**0.802**	**0.807**	**0.773**
Kmeans (tf-idfs)	0.667	0.658	0.563	0.673	0.699	0.562
Kmeans (emb.)	0.731	**0.730**	0.652	0.745	0.793	0.636

Таблица 1: Сравнение базовых моделей на 1-ом уровне

модели ARTM использовались регуляризаторы сглаживания p_{tdw}, сглаживания и разреживания матриц Φ и Θ.

Оценим качество построения 2-ого уровня кластеризации, учитывая, что первый уровень зафиксирован (таблица 2). Будем измерять accuracy и f1-меру указанным выше способом. Заметим, что предложенный способ сравнения, не позволяет достоверно оценить качество всей модели, а лишь помогает оценить качество построения второго уровня иерархии.

По итогам оценивания тематические модели на «отложенных» выборках показали лучшие результаты. На основной выборке все модели сработали примерно одинаково. Для алгоритмов hKmeans подбиралось число кластеров, на которые разбивался каждый кластер (от 3 до 10), для тематических моделей подбиралось число тем на втором уровне модели.

Как было описано выше, построение второго уровня кластеризации — достаточно тяжёлая задача, необходимо разбить на группы точки пространства, которые достаточно близко к друг другу находятся. Схожее качество объяснимо сложностью задачи, никакая модель не получает преимущество, длина документов и прочие факторы уже не играют такую сильную роль как на первом уровне.

	1-1		1-2		2-1	
	acc.	f1	acc.	f1	acc.	f1
hPLSA	0.628	**0.719**	**0.741**	**0.824**	0.617	0.629
hARTM	**0.635**	0.710	0.681	0.771	0.636	**0.674**
hKmeans (tf-idf)	0.616	**0.719**	0.704	0.807	0.603	0.633
hKmeans (emb.)	0.627	**0.717**	0.685	0.778	**0.643**	**0.673**

Таблица 2: Сравнение базовых моделей на 2-ом уровне

	1-1	1-2	2-1
hPLSA	0.603	0.675	0.633
hARTM	**0.636**	**0.683**	0.631
hKmeans (tf-idf)	0.568	0.593	**0.649**
hKmeans (emb.)	0.615	0.638	0.641

Таблица 3: Общее сравнение базовых моделей по точности

Оценим общее качество моделей (таблица 3). Заметим, что по обеим разметкам для первой коллекции у тематических моделей большое преимущество, на второй коллекции результаты у всех подходов близки друг к другу.

В дальнейших экспериментах в основном будут развиваться подходы, связанные с тематическим моделированием. Это связано с тем, что в эти модели очень легко внедрить дополнительную информацию/дополнительные предположения, позволяющие улучшить результат.

5.2 Выделение n-грамм

Стандартный способ улучшения качества тематических моделей — использование моделей выделения коллокаций/нграмм/терминов. Рассмотрим влияние метода TopMine и его модификаций, предложенных в пункте 3.1, на качество модели PLSA. Также склеим частицу «не» с следующими за ней глаголами (модификация `«neg_verbs»`). Для всех моделей в таблице, использующих модальность n-грамм, подбирался коэффициент модальности, регуляризаторы не использовались.

	1-1		1-2		2-1	
	acc.	f1	acc.	f1	acc.	f1
hPLSA	0.628	0.719	0.741	0.824	0.617	0.629
+neg_verbs	0.615	0.704	0.696	0.789	**0.651**	0.660
+ TopMine	**0.733**	**0.818**	**0.832**	**0.902**	0.622	0.674
+ wordset	0.720	0.810	0.769	0.862	0.618	**0.698**
+ overlapped	0.664	0.745	0.778	0.848	0.594	0.674

Таблица 4: Сравнение выделения терминов для 1 уровня моделей

Результаты моделей можно увидеть в таблицах 4 и 5. По результатм можно сделать вывод, что жадная «поуровневая» оптимизация параметров модели не является

	1-1	1-2	2-1
hPLSA	0.603	**0.675**	0.633
+neg_verbs	0.609	0.646	0.644
+ TopMine	0.612	0.644	0.633
+ wordset	0.608	0.644	0.644
+ overlapped	**0.635**	**0.674**	**0.655**

Таблица 5: Сравнение выделения терминов для 1 уровня моделей

до конца правильной. На первом уровне, модификации wordset и overlapped не оказывают серьёзного влияния, что легко объяснимо тем, что эти модификации и нужны только для учёта спефичиных редких ситуаций. На втором уровне влияние комбинации всех модификаций более существенное. Первый уровень модели служит грубым разделителем, в то время как второй производит более тонкое разделение. Этим и объясняется то, что эффект от модификаций есть только на втором уровне.

5.3 Распознавание именованных сущностей

Рассмотрим, насколько именованные сущности влияют на построенную модель. Для начала рассмотрим, насколько в топ-словах (максимальных по ϕ_{wt} в теме t для тематической модели, с максимальными tf-idf для кластеризации представлений) содержатся имена собственные (таблица 6, левый столбец). Доля не является критичной, но даже несколько популярных имён, вносящих шум в модель, могут серьёзно влиять на качество.

	Нет обработки	Есть обработка (RNN+CRF)
hPLSA	3%	0.3%
hKmeans (emb.)	1%	0.2%

Таблица 6: Доля имён в топ-словах кластеров для коллекции 1

Доля именованных сущностей достаточно небольшая, однако даже такой процент лучше устранить (непонятно, как будет вести себя модель при добавлении дополнительных регуляризаторов/модальностей, количество именованных сущностей в стоп-словах может возрасти). К тому же, именованные сущности никак не влияют на интент сообщений, поэтому любые появления имён/фамилий/отчеств можно заменять на тег <PERSON>.

	1-1	1-2	2-1
hARTM	0.635	0.674	0.655
+NER правила	0.634	0.661	0.635
+NER RNN-CRF	**0.64**	**0.68**	**0.662**

Таблица 7: Влияние выделения именованных сущностей на качество

Рассмотрим два подхода выделения NER: с помощью правил/словарей и с помощью использования RNN+CRF модели. Результаты можно увидеть в таблице 7. Использование обоих способов понижает количество именованных сущностей в топ-словах кластеров (левый столбец 6), однако качество модели после выделения сущностей нейронной сетью выше. Отметим также, что использование нейронной сети на практике может быть вычислительно неэффективно, т.к. исполнение неосновной операции будет занимать достаточно большое время даже с учётом вычислений на GPU.

В целом, использование выделения именованных сущностей помогло немного улучшить качество итоговой модели. Заметим также, что прирост мог быть более существенным, если выделять не только имена, но и другие типы именованных сущностей.

5.4 Исправление опечаток

Рассмотрим, насколько опечатки влияют на построенную модель. Рассмотрим базовую модель Jamspell и модель с использованием модификаций, описанных в пункте 3.3 (таблица 8). Заметим, что базовый алгоритм даже немного ухудшает итоговое качество модели. Это связано с тем, что в коллекции много специфичных терминов (названий документов, названий инстанций), которые модель может ошибочно принимать за слова с ошибками.

	1-1	1-2	2-1
hARTM	0.64	0.68	**0.662**
+ base spellcheck	0.635	0.674	0.655
+ mod. spellcheck	**0.657**	**0.686**	**0.663**
hARTM (subtoken mod.)	0.601	0.64	0.633
hypergraphARTM	0.605	0.651	0.64

Таблица 8: Влияние исправления опечаток на качество

Модели, использующие буквенные триграммы не очень хорошо себя показали. Но заметим, что буквенные триграммы делались на основе модели PLSA без учёта n-грам и распознавания именованных сущностей. Одна из проблем моделей, использующих триграммы, сильная заточенность модели, отвечающей за модальность триграмм, на самые популярные слова в теме (например, первые топ-нграммы в теме — состовляющие первого топ-слова в теме). Проблема гиперграфовой модели — отсутствие механизма фоновых тем, запрещающего нахождение в транзакции даже одной сущности не влияющей на тематику контейнера.

5.5 Группировка токенов по уровням

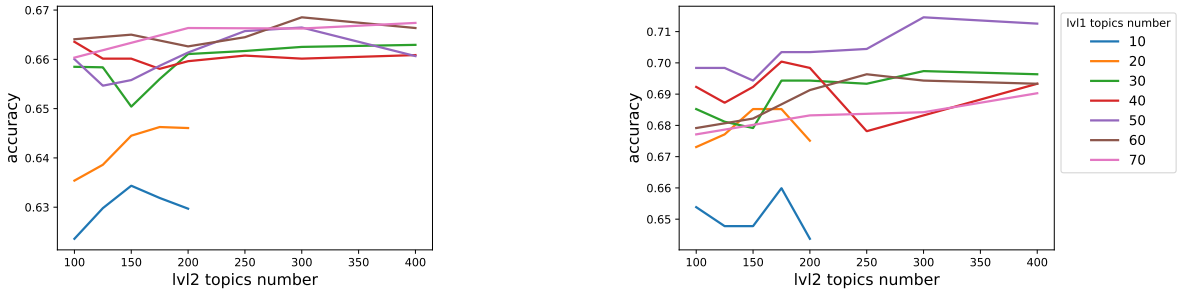
Рассмотрим влияние использования разных модальностей на разных уровнях на качество модели. Первый уровень иерархии строился по тематичным токенам (относительный вес 1.0) и основным токенам (относительный вес 0.5). На втором уровне вес тематичных токенов понижался до 0.5, но больший вес приобретали функциональные токены (относительный вес 0.5). Основные токены нужны для связи разных уровней иерархий между собой, иначе получались бы две независимые кластеризации. Преимущество, полученное от использования такой схемы модальностей показано в таблице 5.5.

	1-1	1-2	2-1
hARTM	0.657	0.686	0.663
hARTM (разные уровни)	**0.667**	**0.715**	**0.672**

Повышение точности по всем трём набора данных показывает успешность данной техники. Заметим также, что ещё более высокое качество можно получить с помощью детального подбора весов модальностей. В данном эксперименте все шесть весов подбирались по сетке [0, 0.5, 1] (коэффициенты являются относительными). Метод с описанным выше набором весов показал самое лучшее «среднее» качество (по всем трём наборам данных).

5.6 Подбор числа тем

Заключительный эксперимент данной работы — попытка использовать предложенную модель для оценки числа тем. На графиках 1 показана зависимость качества модели от числа тем на каждом из уровней.



(1-1) (1-2)

Рис. 1: Зависимость качества от числа тем на каждом из уровней

Эксперимент позволяет хорошо подобрать число тем на первом уровне — оно должно быть не меньше 30. Это согласуется с априорными представлениями о коллекции, не нужно слишком большое число тем, чтобы разделить коллекцию на тематические группы. Подбор числа тем на втором уровне по выбранной разметке работает не так хорошо, но очевидно, что число тем должно быть не меньше 200 (до этой отметки, некоторые модели ведут себя нестабильно). Для более точной оценки, необходимо смотреть на дополнительные факторы (например, попытаться повысить когерентность — меру интерпретируемости тем).

6 Заключение

В данной работе был предложен способ решения задачи выделения тематик в неразмеченной коллекции диалогов с помощью использования тематического моделирования. Был предложен способ формализации задачи, использующий разметку пар диалогов по близости. Было показано, как улучшить качество модели, используя модели выделения n-грамм, распознавания именованных сущностей, исправления опечаток. Были предложены небольшие модификации моделей выделения n-грамм и исправления опечаток, позволяющие улучшить качество решения задачи.

На примере данной работы показана основная проблема при построении многоуровневой кластеризации данных — каждый следующий уровень будет сильно похож на предыдущий. Для решения этой проблемы предлагается использовать на разных уровнях модели разные наборы модальностей. Предложен способ формирования модальностей модели на основе анализа частей речи слов, учитывающей специфику постановки задачи. Проведены эксперименты, показывающие успешность данного подхода. Также показан способ подбора количества тем на основе построенной разметки.

Список литературы

[1] *Anh L. T., Arkhipov M. Y., Burtsev M. S.* Application of a hybrid bi-lstm-crf model to the task of russian named entity recognition // *CoRR.* — 2017. — Vol. abs/1709.09686. http://arxiv.org/abs/1709.09686.

[2] *Blei D. M., Ng A., Jordan M.* Latent Dirichlet allocation // *JMLR.* — 2003. — Vol. 3. — Pp. 993–1022.

[3] *Chirkova N., Vorontsov K.* Additive regularization for hierarchical multimodal topic modeling // *Journal Machine Learning and Data Analysis.* — 2016. — Vol. 2, no. 2. — Pp. 187–200.

[4] Efficient estimation of word representations in vector space / T. Mikolov, K. Chen, G. Corrado, J. Dean // *arXiv preprint arXiv:1301.3781.* — 2013. http://www.bibsonomy.org/bibtex/24a3db34a5744ad8a2704d42f0ef00905/scheuerpflug.

[5] Enriching word vectors with subword information / P. Bojanowski, E. Grave, A. Joulin, T. Mikolov // *Transactions of the Association for Computational Linguistics.* — 2017. — Vol. 5. — Pp. 135–146.

[6] *Harris Z.* Distributional structure // *Word.* — 1954. — Vol. 10, no. 23. — Pp. 146–162.

[7] *Hoffmann T.* Unsupervised learning by probabilistic latent semantic analysis // *Machine Learning.* — 2001. — Vol. 42, no. 1. — Pp. 177–196.

[8] *Попов А. С.* Использование гиперграфовой тематической модели в задаче кластеризации диалогов. — 2019. — 03. — Pp. 329–342.

[9] Introducing baselines for russian named entity recognition / R. Gareev, M. Tkachenko, V. Solovyev et al. — Vol. 7816. — 2013. — 03. — Pp. 329–342.

[10] *Ivanitskiy R., Shipilo A., Kovriguina L.* Russian named entities recognition and classification using distributed word and phrase representations // SIMBig. — 2016.

[11] *McInnes L., Healy J., Melville J.* Umap: Uniform manifold approximation and projection for dimension reduction // *arXiv preprint arXiv:1802.03426.* — 2018.

[12] Scalable topical phrase mining from text corpora / A. El-Kishky, Y. Song, C. Wang et al. // *PVLDB.* — 2014. — Vol. 8. — Pp. 305–316.

[13] *Vorontsov K.* Additive regularization for topic models of text collections // *Doklady Mathematics*. — 2014. — Vol. 89, no. 3. — Pp. 301–304. http://dx.doi.org/10.1134/S1064562414020185.

[14] *Vorontsov K., Potapenko A.* Additive regularization of topic models // *Machine Learning*. — 2015. — Vol. 101, no. 1-3. — Pp. 303–323.