

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИТМО»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

РАЗРАБОТКА АЛГОРИТМОВ РАБОТЫ С ФОРМАЛЬНОЙ
МОДЕЛЬЮ ДИАЛОГОВ, ПРЕДСТАВЛЕННЫХ В ВИДЕ ГРАФОВ

Автор: Савон Юлия Константиновна _____

Направление подготовки: 01.03.02 Прикладная
математика и информатика

Квалификация: Бакалавр

Руководитель ВКР: Ульяновцев В.И., к.т.н. _____

Санкт-Петербург, 2020 г.

Обучающийся Савон Юлия Константиновна
Группа М3437 Факультет ИТиП

Направленность (профиль), специализация
Математические модели и алгоритмы в разработке программного обеспечения

Консультанты:

а) Ступаков И.М., канд. тех. наук, доцент

ВКР принята «_____» _____ 20__ г.

Оригинальность ВКР _____%

ВКР выполнена с оценкой _____

Дата защиты «26» июня 2020 г.

Секретарь ГЭК Павлова О.Н.

Листов хранения _____

Демонстрационных материалов/Чертежей хранения _____

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИТМО»

УТВЕРЖДАЮ

Руководитель ОП

проф., д.т.н. Парфенов В.Г. _____

« ____ » _____ 20 ____ г.

**ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ**

Обучающийся Савон Юлия Константиновна

Группа М3437 **Факультет** ИТиП

Квалификация: Бакалавр

Направление подготовки (специальность): 01.03.02 Прикладная математика и информатика

Направленность (профиль): Математические модели и алгоритмы в разработке программного обеспечения

Тема ВКР: Разработка алгоритмов работы с формальной моделью диалогов, представленных в виде графов

Руководитель Ульянцев В.И., к.т.н., доцент факультета информационных технологий и программирования Университета ИТМО

2 Срок сдачи студентом законченной работы до: «10» июня 2020 г.

3 Техническое задание и исходные данные к работе

Требуется провести исследование и разработать набор алгоритмов для выявления отвлечений в графовой модели для голосовой диалоговой системы. Алгоритм принимает набор диалогов, в размере нескольких тысяч. Предварительно выстроенный граф и кластеризацию для фраз оператора.

На выходе ожидается получить набор отвлечений и перестроенный граф. В качестве метрики качества будет использоваться сравнение с уже существующими графами, которые создавались вручную.

4 Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов)

Пояснительная записка должна описывать предметную область диалогов представленных в виде графов. Так же формулировать цель и задачу выделения отвлечений, содержать описание алгоритмов их поиска. Должны быть описаны сложности и методы их разрешения, если они возникали. Кроме того должны быть приведены примеры работы алгоритмов и сравнение с существующими решениями. Кроме того пояснительная записка должна содержать описания задач из смежных областей и их то, как эти задачи связаны с задачей решаемой в работе.

5 Перечень графического материала (с указанием обязательного материала)

Графические материалы и чертежи работой не предусмотрены

6 Исходные материалы и пособия

- а) Среда разработки Visual Studio Code;
- б) ГОСТ 7.32–2001 «Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления».

7 Дата выдачи задания «01» сентября 2019 г.

Руководитель ВКР _____

Задание принял к исполнению _____ «01» сентября 2019 г.

Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИТМО»

АННОТАЦИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Обучающийся: Савон Юлия Константиновна

Наименование темы ВКР: Разработка алгоритмов работы с формальной моделью диалогов, представленных в виде графов

Наименование организации, где выполнена ВКР: Университет ИТМО

ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ
РАБОТЫ

1 Цель исследования: Разработка алгоритма выделяющего отвлечения в диалоге, представленном в виде графа.

2 Задачи, решаемые в ВКР:

- а) Разработать алгоритмы выделения отвлечий;
- б) Реализовать описанные алгоритмы;
- в) Перестроить граф в соответствии с используемой моделью в компании;
- г) Проанализировать результаты работы алгоритмов;
- д) Интегрировать разработки в инфраструктуру компании.

3 Число источников, использованных при составлении обзора: 0

4 Полное число источников, использованных в работе: 6

5 В том числе источников по годам:

Отечественных			Иностранных		
Последние 5 лет	От 5 до 10 лет	Более 10 лет	Последние 5 лет	От 5 до 10 лет	Более 10 лет
0	2	1	1	0	2

6 Использование информационных ресурсов Internet: да, число ресурсов: 1

7 Использование современных пакетов компьютерных программ и технологий:

Пакеты компьютерных программ и технологий	Раздел работы
Интегрированная среда разработки PyCharm	Глава 3
Программное обеспечение для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации Docker	
Распределённая система контроля версий Git	Глава 3

8 Краткая характеристика полученных результатов

9 Гранты, полученные при выполнении работы

10 Наличие публикаций и выступлений на конференциях по теме выпускной работы
По теме этой работы был сделан доклад на Конгрессе Молодых Ученых.

Обучающийся Савон Ю.К. _____

Руководитель Ульянцев В.И. _____

« _____ » _____ 20 ____ г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1. Обзор диалоговой модели.....	7
1.1. Описание постановки задачи поиска отвлечений в диалговой модели графа	7
1.2. Анализ задачи выделения отвлечений.....	7
1.3. Постановка задачи.....	8
1.4. Выводы по первой главе.....	8
2. Теоретическая часть	10
2.1. Задача выделения отвлечений	10
2.2. Соответствие поставленным требованиям	11
3. Описание алгоритмов поиска отвлечений и объединения кластеров	12
3.1. Подход поиска отвлечений с большим количеством рёбер входящих в вершину.....	12
3.2. Подход поиска отвлечений с поиском циклов	13
3.3. Выборка хороших кластеров	14
3.4. Функция сравнения сообщений	16
3.5. Слияние кластеров операторских сообщений	17
3.6. Оценка для сравнения графов	17
4. Результаты экспериментов.....	19
4.1. Используемые данные	19
4.2. Результаты улучшения кластеризации	20
4.3. Результаты работы алгоритма поиска отвлечений по рёбрам	21
4.4. Результаты алгоритма поиск циклов	21
4.5. Программная реализация	24
ЗАКЛЮЧЕНИЕ	25
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	26

ВВЕДЕНИЕ

Эта работа посвящена исследованию в области диалоговых систем.

Диалоговая система – алгоритм, который умеет принимать участие в диалоге на естественном языке и использует правила общения между людьми.

В качестве примера диалоговых систем можно привести:

- чат-боты
- голосовые помощники
- автоответчики в колл-центрах

Такие диалоговые системы могут быть как довольно простыми (например чат-бот отвечающий на заранее известный набор команд), так и сложными (например бот, отвечающий на вопрос на естественном языке и в качестве ответа возвращающий некоторую информацию из базы знаний).

В последнее время набрали популярность технологии распознавания и генерации речи, которые позволили создавать диалоговые системы, ведущие голосовой разговор. Данная работа рассматривает алгоритмы именно для голосовых диалогов. Пример решения такой диалоговой системы можно рассмотреть в статье [5].

Такие звонки с одной стороны должны быть не отличимы от звонков человека, с другой они должны придерживаться некоторого сценария.

Сценарий диалога с оператором — некоторый алгоритм, предоставленный человеку, который звонит по некоторому набору номеров (либо же принимает входящий звонок или общается посредством программного обеспечения для аудиосвязи). Целью сценария обычно является получить или донести до клиента информацию.

Несмотря на то, что под сценарием диалога понимается некоторый алгоритм, необходимо понимать что для человека и диалоговой системы это принципиально разные сущности. Между скриптом¹ и алгоритмом, с точки зрения набора действий для машины есть большая разница.

¹Здесь и далее в тексте **скрипт** и **сценарий** будут использоваться как синонимы

Для человека это скорее структурированный список вопросов которые он должен задать и некоторая дополнительная база знаний с ответами на нестандартные вопросы. Кроме того человек может помнить некоторые факты и выдавать их дополнительно в зависимости от контекста. Он сам умеет обрабатывать ситуации, такие как отвлечение от основных вопросов или переспрашивание.

Для скрипта же, любую реакцию надо прописывать, все возможные данные хранить и обновлять. Кроме того есть требование поддерживать этот скрипт доступным для восприятия человеком (например лингвистом), поскольку возникает необходимость в ручном анализе и редактировании. В данном случае такой скрипт будет представлен в виде графа с дополнительной информацией.

Голосовая диалоговая система — программа, которая используя сценарий умеет проводить диалог с клиентом, интерпретировать и записывать информацию полученную от клиента, а так же состояния завершенного разговора. Кроме того она умеет отвечать на заранее прописанный в скрипте набор вопросов и возвращаться обратно к диалогу.

На данный момент существуют графы для диалогов, которые создаются вручную. Но писать их долго, а продумывать все важные случаи реакций сложно и трудоёмко.

Кроме того хочется иметь возможность усложнять вариативность диалогов. В связи с этим, ставится глобальная задача по созданию графа из набора проведенных диалогов.

Достаточно часто возникает ситуация, когда некоторый общий вопрос может возникнуть в любом месте диалога (например «А что это за компания?»). Таким образом было решено разделить их в отдельные подграфы и сделать возможность переходить в них при некоторых условиях из каждой вершины. В дальнейшем мы будем называть такие случаи **отвлечениями**.

В работе будут рассмотрены различные алгоритмы автоматического поиска таких отвлечений.

ГЛАВА 1. ОБЗОР ДИАЛОГОВОЙ МОДЕЛИ

1.1. Описание постановки задачи поиска отвлечений в диалговой модели графа

Глобальная задача, состоит в следующем: необходимо по набору диалогов восстановить диалоговую систему.

Диалоговая система в данном случае представлена в виде графа, но так же существуют и другие представления.

Целью данной работы ставится найти отвлечения в восстановленном графе.

На данный момент кроме восстанавливаемой нами модели есть графы который составляется вручную. В этом графе вершинами являются кластеры фраз диалговой системы, а рёбрами фразы человека. То есть при ответах человека происходит переход в разные новые вершины.

Особенностями данной структуры, которые важно отметить, является то, что помимо обычных переходов существуют так же скрытые переходы, которые по умолчанию могут встретиться в любом месте. В качестве примера можно привести вопрос: «А какую компанию вы представляете?». При звонке люди могут задать этот вопрос не сразу.

Для того, чтобы не рассматривать каждый такой случай при переходе из каждой вершины, выделяются **отвлечения**.

Отвлечение - вопрос, который может быть задан в любом месте диалога. Кроме того так же к одному из подтипу отвлечений относятся случаи, когда оператор не услышал фразу или вопрос и вынужден переспрашивать. В зависимости от модели это может выделяться в специальную сущность или являться отвлечением.

В задаче восстановления будет использоваться несколько другой граф. В нём вершинами будут являться кластеры и фраз оператора, и фраз человека. А ребрами последовательные пары фраз из кластеров в диалоге.

1.2. Анализ задачи выделения отвлечений

На вход подаётся набор диалогов по которым нужно получить граф для диалговой системы, который бы мог проводить аналогичные диалоги.

Граф, если получать его путём обычной кластеризации операторских фраз, получается очень громоздким. В нём плохо видно структуру, его сложно анализировать.

Поскольку конечной целью является построить автомат аналогичный тому, что работает в продуктовой части команды, то появляется требование привести его в состояние, когда его можно изучать вручную.

То есть так же как и в графе создаваемом вручную, появляется необходимость выделить отвлечения. В этом случае его структура становится более удобной для изучения. Его можно использовать как вспомогательный инструмент.

Такие выделения отвлечений позволяют грамотно обрабатывать сценарии, которых не было в изначальном наборе диалогов. Если такие отвлечения не выделить, то в случае вопроса, не предусмотренного в этом месте диалоговая система либо ответит не попадая, либо зависнет.

Необходима хорошая кластеризация текстов, поскольку любой алгоритм выделения отвлечений так или иначе будет опираться на эту кластеризацию. При таких кластеризациях полезно учитывать имеющуюся информацию, то есть фразы до текущей и после.

1.3. Постановка задачи

В связи с описаной выше проблемой громоздкости графа, появляется необходимость выделить отвлечения, которые соответственно могли бы встретиться в любом месте.

В случае, когда диалог проводит оператор вопросы-отвлечения нет необходимости расписывать, но в отличие от оператора диалоговой системе нужен чёткий скрипт, который бы обрабатывал все возможные случаи.

Необходимо выделить отвлечения и перестроить граф таким образом, чтобы отвлечения выделились в отдельный подграф, куда можно было бы перейти из любого места диалога. Такой граф будет покрывать большее количество диалогов. Кроме того, и сам граф, и отвлечения становится проще анализировать при необходимости вручную.

1.4. Выводы по первой главе

Рассмотрена предметная область графовой структуры диалогов. Разобрана структура продуктового графа и структура графа для вос-

становливаемой модели. Проведён анализ задачи восстановления графа. Поставлена задача выделения отвлечений.

ГЛАВА 2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

2.1. Задача выделения отвлечений

Поскольку граф на данном этапе всё ещё перестраивается, то информация о произнесенных человеком фразах является полезной, так как содержит в себе контекст. Поэтому мы оставляем фразы человека в качестве вершин.

Нужно понимать, что изначально кластеризация проводилась только по фразам оператора. Фразы вершин же были разделены на группы, где для каждой группы совпадала предыдущая и последующая вершины оператора. И уже внутри этих групп бились на некоторые подгруппы.

Рассмотрим два подхода в решении задачи выделения отвлечений:

- Первый подход заключается в том, чтобы выделить вершины, в которые идёт много рёбер. Порог считается функцией от количества кластеров на которые бьются фразы оператора.

Мы предполагаем, что поскольку отвлечение встречается в разных местах, то и рёбра будут идти в него из множества различных вершин. Такая гипотеза является хорошей, поскольку в обычном графе в вершину обычно приходит одна или две ветки, в случае же отвлечения их должно быть много, или же оно встречается крайне редко.

- В качестве другого подхода можно выделить циклы и сказать, что вершина следующая в диалоге за вершиной повторения с некоторой вероятностью будет являться началом ответа.

Здесь мы пользуемся наблюдением, что после отвлечения на странный вопрос, оператор зачастую повторяет ту же или схожую фразу для возвращения в сценарий. Более того, эта идея используется в графе, который реализован для реального окружения. Там диалоговая система так же повторяет фразу, её сокращённую версию или её иную формулировку, которую произносил, перед тем, как перейти в отвлечение.

Поскольку подходы используют разные идеи их так же имеет смысл комбинировать и использовать данные полученные в обоих подходах.

2.2. Соответствие поставленным требованиям

Такие алгоритмы позволят найти для диалогов которые были проведены с системой, отвлечения, и сопоставить их с выделенными лингвистами вручную для графа.

ГЛАВА 3. ОПИСАНИЕ АЛГОРИТМОВ ПОИСКА ОТВЛЕЧЕНИЙ И ОБЪЕДИНЕНИЯ КЛАСТЕРОВ

3.1. Подход поиска отвлечений с большим количеством рёбер входящих в вершину

Выбираются вершины, в которые входит много ребер. Для значений размеров кластеров в интервале от двенадцати до пятнадцати был выбран параметр три. При увеличении количества кластеров соответственно должен увеличиваться и порог.

В некоторых случаях отвлечения не ограничиваются одной вершиной. В связи с этим появляется необходимость определить, где заканчивается то или иное отвлечение, и где соответственно оператору необходимо вернуться в вершину с которой он в это отвлечение ушёл. Для этого предположим следующую гипотезу – если в вершину мы можем попасть только пройдя через отвлечение, то это значит, что фраза является частью соответствующего отвлечения.

Для того, чтобы найти соответствующий хвост для отвлечения используем следующий алгоритм: заблокируем вершину и рассмотрим все вершины, достижимые из вершины старта. Важно так же помнить о возможности того, что в кластерах могут быть допущены небольшие ошибки, поэтому если почти весь кластер недостижим после блокировки, то он так же входит в отвлечение.

Ниже представлен псевдокод функции поиска таких хвостов для отвлечений:

```
function Digration finder(graph, node, dialogs)
  for dialog ∈ dialogs do
    flag ← True
    for msg ∈ dialog do
      msgs[msg.cluster] ++
      if (msg.cluster == node.cluster) and flag then
        achieved.add(msg.cluster)
        achievedmsgs[msg.cluster].add(msg)
      else
        flag ← False
    end if
```

```

    end for
end for
for cluster ∈ graph.clusters do
    if clusternotinachieved then
        diression_clusters.add(cluster)
    else
        if achieved_msgs[cluster]/msgs[cluster] > 0.9 then
            diression_clusters.add(cluster)
        end if
    end if
end for
end function

```

Во время тестирования на реальных диалогах человека с человеком была выявлена важная особенность. На работу алгоритма очень сильно влияет качество кластеризации. Поскольку фразы в голосовых диалогах не всегда верно переводятся в текст, сами фразы сравнительно короткие. А в случае людей-операторов ещё и очень вариативные, то кластеризация оказалась очень некачественной.

3.2. Подход поиска отвлечений с поиском циклов

Поскольку говорит на сторонние темы только человек, а оператор идёт по скрипту, то в подавляющем большинстве случаев после ответа на сторонний вопрос, оператор задаёт свой вопрос заново. В связи со спецификой человеческий диалогов можно выделить несколько полезных особенностей:

- Подавляющее большинство поддиалогов отвлечений достаточно короткие.
- Вопрос повторяет оператор, поэтому циклы можно искать с повторением только операторской вершины.
- Возможно появление отвлечения внутри отвлечения, поэтому необходимо найти оба.

Для решения особенности первого пункта было добавлено ограничение на расстояние между ними в диалоге. Оно должно быть не слиш-

ком велико, поскольку ясно что отвлечение бывает длинным очень редко, а иначе можно случайно выкинуть почти весь диалог.

Проблема последнего пункта решается тем, что мы удаляем циклы по мере нахождения. Таким образом внутренний цикл будет удалён раньше внешнего.

В этом случае в качестве потенциальных отвлечений мы выбираем вершины человеческих фраз которые первые следуют после начала цикла. После перевода графа в продуктовый режим у нас эти вершины станут рёбрами и таким образом в графе появятся рёбра-триггеры, которые будут перенаправлять ход диалога в вершину-отвлечение.

Ниже находится псевдокод для поиска одного цикла:

```
function Remove cycles(dialog)
  for msg ∈ dialog do
    msg_num ++
    if (last_msg[msg.cluster] − msg_num) ≤ 5 then
      cycle_end ← last_msg[msg.cluster]
      dialog.remove_cluster(cycle_start, msg_num)
    end if
  end for
end function
```

3.3. Выборка хороших кластеров

В качестве решения проблемы некачественной кластеризации было предложено использовать данные из фраз человека. До этого для всех фраз человека между двумя фразами оператора, они кластеризовались и никак не использовались.

Было решено кластеризировать тексты пользователей. Но поскольку как описывалось выше, кластеризация не достаточно хорошая, то необходимо было отсеять плохие кластеры во избежание каскадных ошибок.

Для этого использовалось попарное сравнение фраз внутри каждого кластера. Для этого сравнивались наборы слов внутри фразы с весами. Если точность превышала порог 0.5, то пара считалась хорошей.

Константа 0.5 была выведена эмпирически. Для большего значения в кластере начинало содержаться большое количество фраз разных по смыслу.

Проверять все пары оказалось очень долго $O(n^2)$, а в предыдущей части восстановления все операции имели ассимптотику не более чем $O(n \log n)$, то получилось так, что эта часть занимала значительно больше половины времени от всего восстановления графа.

Тогда было решено для каждого кластера брать случайную выборку, равная утроенному размеру кластера, ассимптотически это занимало уже $O(n)$. В силу достаточно больших размеров кластеров (размер их для основной массы данных составляет несколько сотен фраз нескольких сотен фраз), статистически показывало те же результаты, что и полная выборка.

Утверждение 1. При размере диалога от 40 фраз необходимое количество экспериментов для попадания в доверительный интервал $Q = 0.95$ и доверительной вероятности¹ $\varepsilon = 0.1$ достаточно $3n$ экспериментов.

Доказательство. Значение $p = 0,5$ – наихудшее, в том смысле, что для него вероятность порогового отклонения превысит выбранное значение ε , при наибольшем количестве экспериментов. Таким образом достаточно доказать утверждение для него. Для такого случая результат равен 96, причём вне зависимости от размера возможных исходов, т.е. вне зависимости от размера кластера. Эксперименты рассчитанные для разных значений приведены в учебнике.[1]

Таким образом для более маленьких кластеров можно рассчитать эти значения перебрав все пары полностью, а для больших кластеров такого количества экспериментов будет достаточно.

Ниже приведён псевдокод для функции, считающей метрику для вершин с большим количеством фраз:

```
function IsGoodCluster(msgs)
    num_of_comparings  $\leftarrow 3 \cdot \text{msgs.size}()$ 
    for (msg1, msg2)  $\in \text{msgs.getPairs}(\text{num\_of\_comparings})$  do
        good_pairs+ = msg_compare(msg1, msg2)
    end for
```

¹Определение доверительного интервала можно изучить здесь.[3]

```

cluster_threshold  $\leftarrow$  0.5
return good_pairs / num_of_comparings  $\geq$  cluster_threshold
end function

```

Функция *msg_compare(msg1, msg2)* отвечает за сравнение двух сообщений.

3.4. Функция сравнения сообщений

Для алгоритма выше необходима функция сравнения двух сообщений. Для неё должны быть выполнены следующие условия:

- Функция должна быть бинарной. *True* – в случае схожести сообщений, *False* – иначе.
- Функция должна работать за $O(1)$, при условии, что длину сообщений мы так же принимаем за $O(1)$.

Второе условие необходимо, поскольку в противном случае при среднем размере групп сообщений от нескольких сотен разница во времени будет в разы увеличиваться.

В качестве алгоритма для сравнения сообщений был выбран следующий: берутся наборы слов в виде сетов и пересекаются с учётом весов слов. Это значение записываем в числитель. В качестве знаменателя берём объединение наборов слов с теми же коэффициентами слов и соответствующими количественными коэффициентами и записываем в знаменатель.

Полученная дробь должна превышать некоторый порог, который ищется вручную.

В качестве альтернативных вариантов функций могут быть использованы следующие:

- Сравнение фраз на основе семантической близости. Об этом алгоритме можно прочитать в следующей статье.[2]
- Сравнение фраз с помощью эмбедингов и на основе косинусного расстояния. Об этом подходе можно почитать здесь.[4]

Все перечисленные и описанные функции подходят под перечень изначальных условий, поскольку функции возвращающие не целые значения в диапазоне от 0 до 1 можно превратить вступенчатую, сделав её бинарной. Все они работают за асимптотически необходимое время.

3.5. Слияние кластеров операторских сообщений

Изначальный алгоритм, который создавал вершины операторов предполагал точный выбор количества кластеров. Новое решение предполагает избыточное разбиение на кластеры. Поскольку количество кластеров на которые разделяются операторские сообщения можно контролировать вручную, это регулируется достаточно просто.

После такого разбиения выделяются кластеры, фразы в которых максимально похожи между собой, в таком разбиении будет меньше ошибок вида – в одном кластере содержательно разные сообщения.

Далее после выбора хороших человеческих вершин, для каждой операторской вершины рассматриваются все другие операторские вершины. Далее было несколько версий метрик для сравнения соседних вершин.

В первой версии рассматривались все соседские вершины. Они не делились группы по положению до или после и соответственно это было недостаточно хорошим решением.

Во второй версии были выбирались все вершины до, исходя из предположения, что они будут больше коррелировать с содержанием операторских вершин и не будет проблемы с тем, что не учтён порядок.

3.6. Оценка для сравнения графов

В качестве верной модели использовалась модель написанная вручную. Модель хранится в формате *json* и в ней фразы человека соответственно являются условиями перехода.

Для того, чтобы можно было сравнивать текущую модель и изначальную, необходимо было перевести вторую модель к формату первой.

Здесь стоит напомнить, что в восстанавливаемой модели кластеры человеческих фраз так же являлись вершинами. Таким образом для того, чтобы была возможность перевести их в рёбра и соответственно сопоставить изначальному графу, для каждой вершины с человеческими кластерами должны были быть верны следующие условия:

- У вершины должна быть ровно одна предыдущая, причём эта вершина должна быть вершиной операторских фраз.
- Последующая вершина должна быть вершиной обязана так же быть операторской и единственной для данной.

- Ни одна из набора фраз данного кластера не должна совпадать с фразами из кластеров, в которые ведут рёбра из предыдущей операторской вершины.

Проверив, что все необходимые условия соблюдены, можно преобразовывать граф, выделяя отвлечения следующим образом:

- Каждая вершина новой модели сопоставлялась одной изначальной. Сопоставление проводилось путём сравнения произносимой фразы.
- Вершины фраз человека были выделены в группы, каждая группа относилась к одному ребру и являлась будущим набором исходящих рёбер.
- Технические вершины в изначальном графе игнорировались, но учитывалось их место в последовательности модели.
- Сравнивались наборы вершин отвлечения в изначальном графе и в полученном.
- При создании метрики для графа была использована статья [6]

ГЛАВА 4. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

4.1. Используемые данные

В качестве данных использовалось два принципиально разных типа наборов диалогов. Первые проводились уже с существующим скриптом. Второй тип, это данные диалогов человека с человеком.

1.Особенность первого заключается в том, что там легко кластеризовать фразы оператора, так как они произносятся всегда одинаково, за исключением вариаций для коротких вариантов.

Но необходимо понимать, что случаи повторения фразы имеют другие вершины соседей поскольку рассчитаны на специфичные случаи. Такие объединения были полезны для алгоритма поиска циклов, но находить их было сложно в силу того, что они явным образом не попадали под условия. Так же можно было выявить слишком заниженные коэффициенты, при слиянии фраз, которые не являлись одинаковыми.

Важным вариантом использования таких данных являлась возможность перевести полученный граф в тот же формат и сравнить полученный результат с оригиналом.

2.Особенности второго соответственно следующие: во первых там говорят разные операторы и у них разный стиль подачи одних и тех же данных. Во вторых диалоги более сложные и зачастую отходят от скрипта. В третьих люди решают более сложные вопросы и умеют давать ответы на не предусмотренные скриптом вопросы. На этот вариант данных стоит ориентироваться, но в связи с отсутствием оригинального скрипта в удобном формате напрямую сравнить полученный результат представляется возможным только вручную.

В обоих случаях есть сложности с переводом речи людей в текст, поэтому иногда даже рассматривая текст диалога вручную нельзя понять что человек имел ввиду.

Для всех датасетов первого типа соответственно существуют файлы, содержащие в себе оригинальный граф, который принимается за верный. Все данные сериализованы в формате JSON.

На Рисунке 1 представлен результат полученного после восстановления графа:

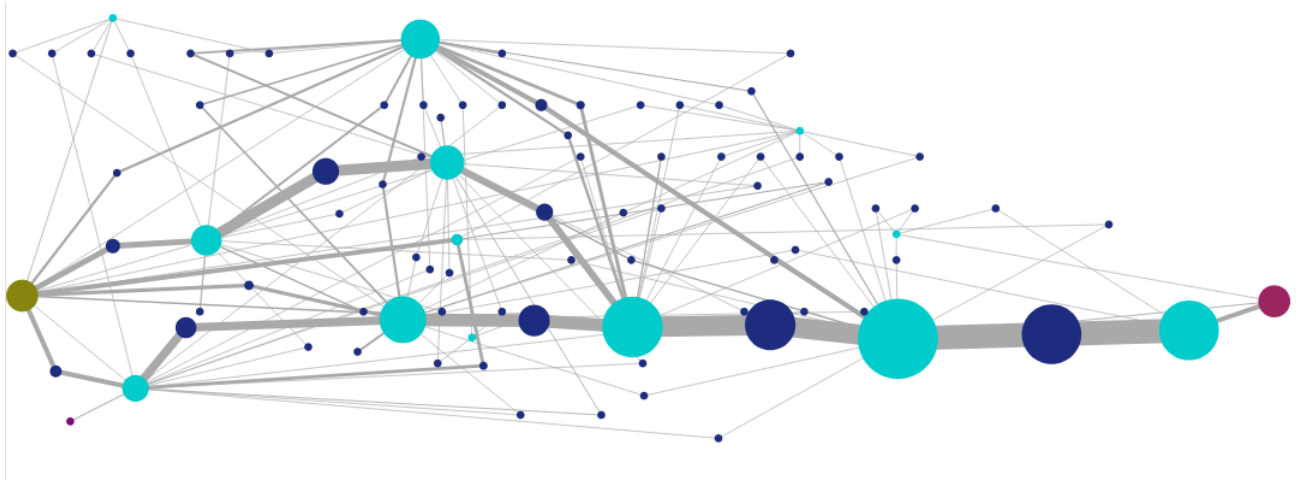


Рисунок 1 – Восстановленный граф

В этой модели голубым цветом обозначены операторские вершины, синим соответственно человеческие. Кроме того, для удобства анализа выделены две фиктивные вершины: стартовая и вершина окончания диалогов. В дальнейшем для приведённых данных они в расчёт не брались.

4.2. Результаты улучшения кластеризации

Алгоритм вносит небольшие изменения и способен объединять некоторые одинаковые кластеры.

Основная сложность состояла в анализе, так как для данных из разговоров с искусственным интеллектом кластеризация фраз оператора изначально была очень хорошей в силу того, что фразы повторялись. Для случаев разговоров человека с человеком анализ результатов приходилось проводить вручную в силу отсутствия разметки данных.

Были проведены исследования по влиянию параметров алгоритма на результат. В результате чего были выявлены следующие закономерности:

- Пороговое значение схожести сообщений для каждого кластера должно варьироваться от 0.2 до 0.5. В этом промежутке в качестве пар схожих сообщений выбираются схожие пары сообщений. Для меньшего значения соответственно практически полностью одинаковые. Для большего у них появляется вариативность.
- Пороговое значение количества хороших пар в кластере должно варьироваться так от 0.25 до 0.5. Для больших значений в одном кластере начинают появляться наборы фраз несущие разный смысл.

Ниже, на Рисунке 2 представлены примеры различных значений промежутков порога для количества хороших пар в кластере:

нет	до свидания	але
нет нет	спасибо до свидания	на какую тему подскажите
нет	спасибо всего доброго до свидания	даша
нет александр александрович	всё спасибо вам за напоминание всего доброго	сколько минут
		интернет
		я

(а) 0.3
(b) 0.5
(с) 0.6

Рисунок 2 – Различные значения пороговой функции количества хороших пар

4.3. Результаты работы алгоритма поиска отвлечений по рёбрам

Для данного алгоритма критично качество кластеризации и наличие небольшого количества выбросов.

В случае с данными из реальных диалогов зачастую не было видно правильно выделенных отвлечений из-за смешения кластеров и операторских и человеческих вершин.

Для данных из диалогов с существующей диалоговой системой алгоритм находил все отвлечения и так же отмечал небольшое количество дополнительных вершины, которые отвлечениями не являются. Основная причина столь большого различия в том, что операторские фразы во втором случае кластеризуются практически идеально.

4.4. Результаты алгоритма поиск циклов

После того, как мы нашли циклы мы их удаляем, таким образом основной граф становится менее громоздким и его легче анализировать. Из особенностей реализации важно отметить, что удаляются не все циклы, а только размера не более чем 5. Рассмотрим пример следующего диалога:

Оператор: Здравствуйте

Человек: Ало

*Оператор: Меня зовут Даша, вы ***¹ ?*

*Человек: *шуршание*²*

Оператор: Извините, я вас не расслышал, повторите пожалуйста

Человек: Я говорю давайте задавайте

*Оператор: Вы сейчас ***?*

Человек: сейчас нет

*Оператор: Вы нашли ***?*

Человек: да

*Оператор: Убрать *** в архив?*

*Человек: *неразборчивый текст**

Оператор: Извините, я вас не расслышал, повторите пожалуйста

*Человек: ненадо пока убирать ****

Оператор: Извините за беспокойство, всего доброго, до свидания

Если мы удалим все фразы между двумя просьбами повторить, то потеряем состояния, которые не являются отвлечениями.

Ниже приведена Таблица 1, в ней видно как уменьшается размер графа при удалении циклов. Для каждого графа в первой строке соответственно записана информация до удаления циклов, во второй – после:

¹Здесь и далее *** будут использоваться для обезличивания частей диалога, которые не содержат важных для алгоритма данных, но несут в себе персональную информацию или информацию о компаниях.

²Здесь и далее *описание* для пропуска не содержательных частей с коротким описанием.

Таблица 1 – Применение алгоритма удаления циклов

Номер датасета	размер	количество вершин	количество рёбер
1	10000	141	587
		134	528
2	661	71	194
		67	179
3	1000	155	570
		142	479
4	1086	98	283
		92	252
5	998	159	753
		142	631
6	5000	48	137
		46	120
7	10000	88	296
		87	293
8	10000	112	515
		111	506
9	3325	161	1078
		154	955
10	10000	116	473
		115	457
11	588	122	366
		115	340

Среди этих датасетов есть 2 принципиально разных типа:

Первый тип – **оператор + человек**, эти датасеты нам предоставили компании и в них человек придерживается некоторого скрипта, но заметно от него отклоняется, поэтому при меньших размерах порядок вершин в них такой же. К этому типу относятся соответственно датасеты 2, 4, 5, 9, 11.

Второй тип – **искусственный интеллект + человек**, эти датасеты были собраны соответственно компанией при звонках со скриптами написанными вручную. К этому типу соответственно относятся 1, 3, 6, 7, 8, 10.

В обоих типах при увеличении размера датасета увеличивается количество вершин. Важно отметить, что большинство из вершин, это ответы человека, количество же операторских вершин примерно равно 15. Для случая диалогов человека с человеком размеры графа могут быть

изначально больше, это объясняется тем, что в нём больше вариативность фраз.

Вырезанные ребра и вершины соответственно отделяются и становится проще анализировать граф.

Если рассмотреть какие вершины из операторских попадают в циклы, то хорошо видно тенденцию того, что есть несколько вершин у которых большое количество ответов попадает в циклы. Некоторые такие вершины, как можно заметить исчезают вообще. Для остальных можно подобрать пороговое значение при котором можно будет считать, что вершина является триггером отвлечения.

4.5. Программная реализация

Предложенные алгоритмы были реализованы на языке `Python` с использованием интегрированной среды разработки `Pycharm`. Кроме того использовался текстовый редактор `Visual Studio Code` с внутренним плагином для визуализации графов описанных в формате `JSON`.

Компания использует приватный `Git`-репозиторий, находящийся на хостинге `Gitlab`. Для удобства локальной разработки приложение запускалось в `Docker`-контейнере.

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы была рассмотрена задача выделения отвлечений в графовой модели для голосовой диалоговой системы. Для её решения были разработаны и написаны алгоритмы основанные на особенностях разговоров оператора с человеком. Были учтены особенности существующей модели графовых диалогов и данные полученные при её использовании.

Были написаны алгоритмы выделения циклов и поиска отвлечений по рёбрам. Эти алгоритмы позволяют выделить значительное количество отвлечений и уменьшают размер основного графа. Кроме того в ходе разработки стало ясно, что алгоритмы очень чувствительны к кластеризации и возникла необходимость улучшить её качество. Для чего был разработан алгоритм на основе данных о соседних фразах.

Поскольку работа является частью большего проекта. То интеграция в реальное окружение запланирована по завершению всех частей проекта. Разработка должна будет автоматизировать часть рабочих процессов.

Одно из возможных направлений развития работы, использовать информацию о пользователе в диалоге, сделав его таким образом более естественным. Так же существует возможность выделить некоторые общие отвлечения для всех диалогов, получая таким образом возможность предсказывать отвлечения в виде подсказок.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 *Мухин О.* Моделирование систем: учебник [Электронный ресурс]. — 2014. — URL: <http://stratum.ac.ru/education/textbooks/modelir/lecture34.html> (дата обр. 30.04.2020).
- 2 *Нзюк Н. Б., Тузовский А. Ф.* Классификация текстов на основе оценки семантической близости терминов // *Izvestiya Tomskogo Politehnicheskogo Universiteta Inzhiniring Georesursov*. — 2012. — Т. 320, № 5. — С. 43–48.
- 3 *Чернова Н.* Теория вероятностей: Учеб. пособие/Новосиб. гос. ун-т // Новосибирск. 2007, 160 с. — 2007.
- 4 *Dönmez İ., Pashaei E., Pashaei E.* Word Vector Space for Text Classification and Prediction According to Author. —
- 5 *Williams J. D., Young S.* Partially observable Markov decision processes for spoken dialog systems // *Computer Speech & Language*. — 2007. — Vol. 21, no. 2. — P. 393–422.
- 6 *Wills P., Meyer F. G.* Metrics for graph comparison: A practitioner's guide // *Plos one*. — 2020. — Vol. 15, no. 2. — e0228728.



УНИВЕРСИТЕТ ИТМО



Центр студенческой науки,
конференций и выставок



IX КОНГРЕСС МОЛОДЫХ УЧЕНЫХ

ДИПЛОМ

ПОБЕДИТЕЛЯ КОНКУРСА ДОКЛАДОВ
ДЛЯ ПОСТУПЛЕНИЯ В МАГИСТРАТУРУ

награждается

**ЮЛИЯ
КОНСТАНТИНОВНА
САВОН**

Ректор
Университета ИТМО

В. Н. Васильев

Санкт-Петербург, 2020 год