Диалоговая система (чатбот и читчат)

АРХИТЕКТУРА РАБОЧЕГО ПРОТОТИПА

https://github.com/Koziev/chatbot

Примечания

- Действующий прототип в виде докер-образа можно скачать отсюда https://github.com/Koziev/chatbot/releases
- В дефолтном профиле бот имеет имя “Вика”, выбранное по фонетическо-распознавательным критериям.
- Скриншоты диалогов сделаны для версии бота в Телеграме.
- Префиксы H: и B: для текстовых фрагментов диалогов соответствуют репликам человека и бота соответственно.
- Префиксы P, Q и A соответствуют предпосылке (premise), вопросу и ответу.
- Термины “факт” и “предпосылка” употребляются в тексте синонимично.

КЛЮЧЕВЫЕ ОСОБЕННОСТИ

Гибридный подход – ML-based и правила. Некоторые виды правил генерируются из слабоструктурированных датасетов (chit-chat stories, continuation rules).

Retrieval-based + **generative** движок – ответы бота строятся на основе информации в базе знаний, обеспечивая консистентность ответов на перефразировки вопросов.

Динамика базы знаний – новые факты могут добавляться в базу по ходу диалога, обеспечивая боту долговременную память в рамках всей сессии и за ее пределами.

Проактивность – бот стремится продолжить диалог, задает пользователю вопросы для пополнения базы знаний, интерпретирует ответы на основе оперативного контекста.

https://github.com/Koziev/chatbot

ОСНОВНЫЕ МОДУЛИ

База знаний (факты и FAQ) – информация о себе, собеседнике и мире

Модели pq-релевантности и синонимичности – поиск информации в базе знаний

Построитель ответа – генерация текста ответа из заданного вопроса и найденного факта

Интерпретатор – раскрытие анафоры, эллипсиса, разбивка на клаузы, приведение к SVO

Правила, сценарии, вербальные формы – событийное программирование бота

Классификаторы – интент, сентимент, эмоции etc

https://github.com/Koziev/chatbot

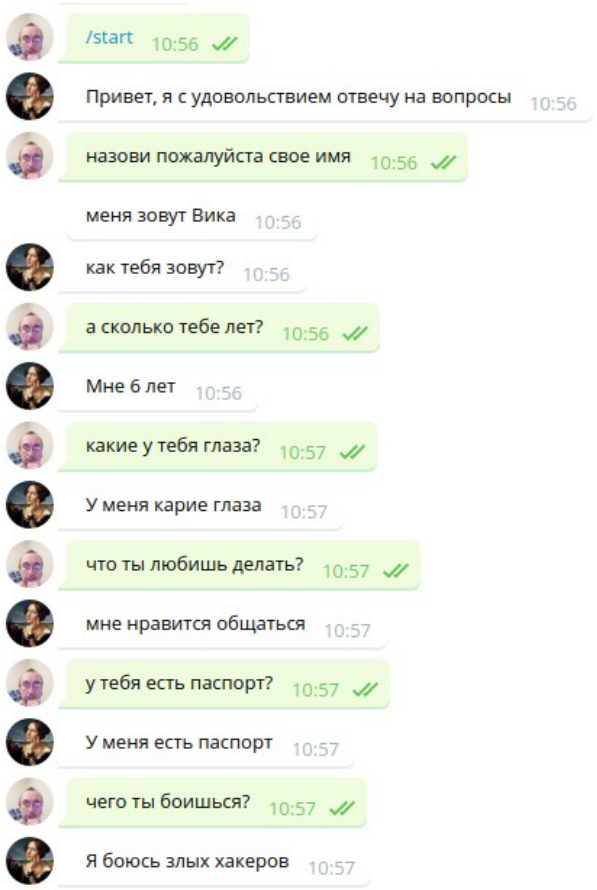
БАЗА ЗНАНИЙ

База знаний состоит из двух частей: **список фактов** и **FAQ**.

Факты хранятся в естественном, неструктурированном виде, как обычные предложения:

меня зовут Вика
мне 6 лет
я люблю мультики про роботов
у меня карие глаза

При старте экземпляра чатбота из заданного текстового файла (**профиль бота**) загружается несколько десятков фактов, описывающих “личность” бота – его имя, возраст, круг интересов, предпочтения и так далее. Эта информация, а также добавляемые по ходу диалога новые факты (об этом см. слайды 5 и 6) используется для генерации содержательных, консистентных ответов чатбота на вопросы пользователя в читчате:



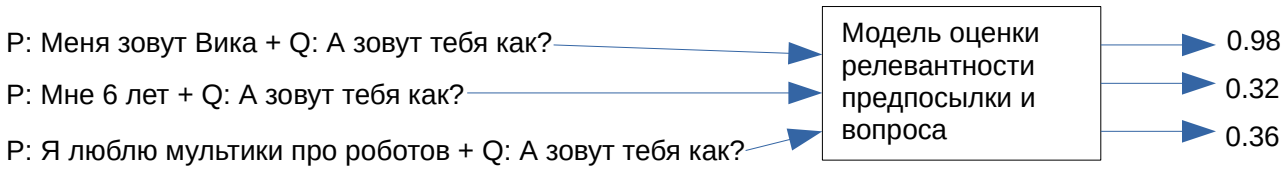
https://github.com/Koziev/chatbot

ПОИСК РЕЛЕВАНТНОГО ФАКТА В БАЗЕ ЗНАНИЙ

Допустим, собеседник вводит вопрос:

H: *а как тебя зовут?*

Из всех фактов в базе знаний и вопроса составляются пары, которые прогоняются через **модель pq-релевантности**:



С содержательной точки зрения эта модель оценивает, содержит ли предпосылка информацию, необходимую для ответа на вопрос. На выходе получаем отранжированные по релевантности предпосылки и выбираем самую подходящую. В нашем случае это “Меня зовут Вика”. Эта препосылка далее используется для генерации ответа бота – см. следующий слайд.

ПОСТРОЕНИЕ ОТВЕТА

Предпосылка, найденная на предыдущем шаге, и заданный вопрос поступают в модель построения ответа. Эта модель выдает текст:

P: *Меня зовут Вика.*
Q: *А зовут тебя как?*
A: ***Вика***

В некоторых случаях ответ генерируется только на основе вопроса, так как в нем содержится вся необходимая информация:

Q: *Сколько будет, если к 2 прибавить 2?*
A: ***4***

Может возникнуть ситуация, когда для вывода ответа нужно 2 предпосылки (подробнее на отдельном слайде):

P1: *У роботов нет паспорта.*
P2: *Я робот.*
Q: *У тебя есть паспорт?*
A: ***нет***

ДРУГИЕ ВОПРОСЫ

Для факта “Меня зовут Вика” релевантными будут и другие вопросы, например:

Верно ли, что ты – Вика?
Тебя случайно не Алиса зовут?
Маруся – это ты, да?
У кого имя “Вика”?

Когда модель pq-релевантности хорошо обучена, она учитывает подобные перефразировки вопросов и фактов.

Как показано на предыдущем слайде, текст ответа во всех этих случаях генерируется моделью построения ответа:

P: *Меня зовут Вика.*
Q: *Верно ли, что ты – Вика?*
A: *да*

P: *Меня зовут Вика.*
Q: *Тебя случайно не Вика зовут?*
A: *меня зовут Вика*

P: *Меня зовут Вика.*
Q: *Маруся – это ты, да?*
A: *нет*

P: *Меня зовут Вика.*
Q: *У кого имя “Вика”?*
A: *у меня*

ДОБАВЛЕНИЕ ФАКТОВ ПО ХОДУ ДИАЛОГА

База знаний это просто список строк с высказываниями в естественном виде. Количество записей и их содержимое никак не ограничены (в разумных пределах). В любой момент в этот список можно добавить новые факты. Чатбот немедленно может использовать их в диалоге, не переобучая свои нейросетевые модели.

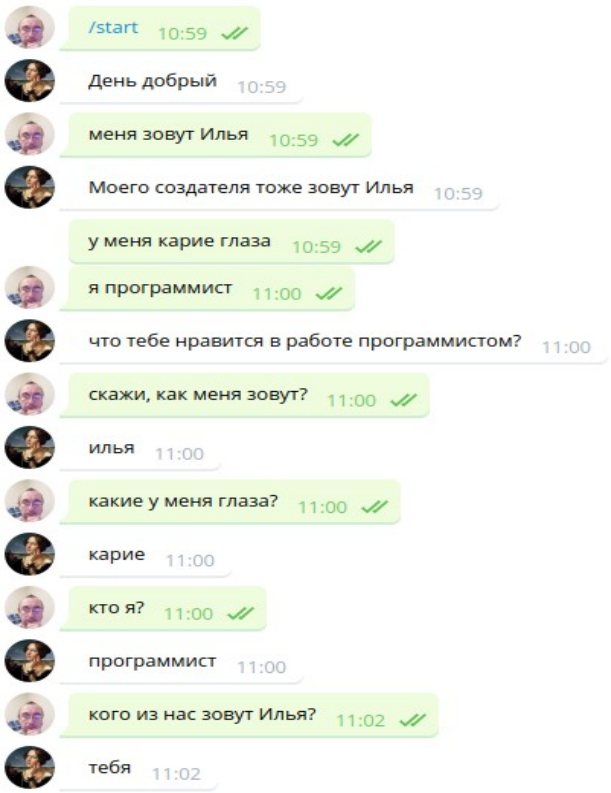
По умолчанию все не-вопросы и не-императивы, внесенные собеседников, запоминаются в базе знаний как новые факты. Например, собеседник сообщает:

Н: *меня зовут Илья*

В список фактов будет внесена эта строка (технически факты, сообщаемые собеседником, перефразируются во 2-е лицо "*тебя зовут Миша*"). Этот новый факт **сразу, на равных** начнет участвовать в процессе подбора релевантной предпосылки для новых вопросов. Например, собеседник захочет проверить, понял ли его чатбот:

Н: *кого из нас зовут Илья?*

Штатная работа модели pq-релевантности выберет факт "*тебя зовут Миша*" для построения ответа "*тебя*".



https://github.com/Koziev/chatbot

ИНТЕГРАЦИЯ С ПОСТАВЩИКАМИ ФАКТОВ

Источником новых фактов в базе знаний могут также быть сторонние сервисы, дающие информацию в виде простых предложений.

Если факты хранятся в серверной реляционной СУБД, то интеграция сводится к выполнению SQL операторов INSERT и UPDATE.

Пример – дата и время

В текущей реализации бота есть автоматическое обновление записи о текущем времени и дате. В базе знаний соответствующие факты выглядят так:

сейчас 19 часов 26 минут
сегодня 4 августа 2020 года

Этого достаточно, чтобы модели pq-релевантности и построителя ответа отвечали на вопросы:

Н: *который час?*
В: *19 часов 26 минут*

Предполагается, что также будут интегрироваться почтовые сервисы (вопросы “*нет ли новых писем от Иры?*”), прогноз погоды и так далее.

ВЫВОД ОТВЕТА ИЗ ДВУХ ПРЕДПОСЫЛОК

Построитель ответа умеет работать с двумя предпосылками:

P1: *ты не любишь любые молочные продукты*
P2: *кефир делают из молока*
Q: *тебе понравится кефир?*
A: *нет*

FAQ

FAQ – отдельная часть базы знаний, содержащая пары “эталонный вопрос + ответ”:

Q: *Почему трава зеленая?*
A: *В траве содержится хлорофилл, который отражает только зеленый свет*

Получив вопрос, бот ищет самый похожий вопрос в FAQ и выводит текст ответа без модификаций. Такой механизм обеспечивает “доставку” контента без искажений, что важно в некоторых бизнес-сценариях.

NB: по сути это k-nearest neighbors с k=1 и специальной метрикой.

Плюсы

- Все эталонные вопросы и ответы в одном месте
- Добавить новую пару Q+A просто
- Текст ответа выводится так, как он записан в БД (генератор ответа не участвует)
- Для сопоставления вопроса используется модель синонимичности (автоматически учитываются перефразировки вопроса)

Минусы

- Вопросы, заданные к другой части ответа, надо прописывать явно как эталонные

Пример синонимичного вопроса:

По какой причине у травы зеленый цвет?

А эти вопросы не синонимичны:

Что, кроме травы, имеет зеленый цвет?
Какой цвет у травы?
Какой цвет у хлорофила?

ИНТЕРПРЕТАТОР

Интерпретатор – нейросетевая модель, которая принимает на вход реплику собеседника и несколько предыдущих реплик диалога, и выдает одну или несколько полных клауз с заполненными эллипсисами, раскрытой анафорой и т.д.

Реальный **чит-чат** – обмен короткими фразами. Человек обычно способен восстановить полный ответ по оперативному контексту:

В: *как же тебя зовут, а?*
Н: *меня – Стас, а тебя?*

Полный ответ человека в этом примере подразумевает 2 клаузы:

Н: *Меня зовут Стас. Как тебя зовут?*

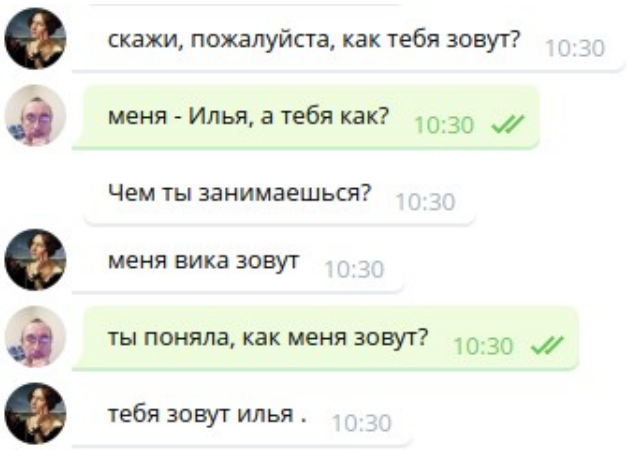
Иногда нужно раскрывать **анафору**:

В: *ты собак любишь?*
Н: *не люблю я **их***

Гэппинг:

В: *ты кошек любишь?*
Н: *их - нет*

Восстановленная, нормализованная и разбитая на клаузы реплика собеседника позволяет обрабатывать ее классическими инструментами NLP (регулярные выражения etc).



https://github.com/Koziev/chatbot

ПРАВИЛА, СЦЕНАРИИ, ВЕРБАЛЬНЫЕ ФОРМЫ

Движок реализует гибридный подход: правила, задаваемые вручную наряду со статистическими моделями.

Преимущества правил

1) Быстрое добавление и изменение поведения (не нужно переобучать модели).

2) Возможность приоритизации правил (весами или порядком применения)

3) Нет проблемы дисбаланса объема интентов

4) Можно создавать правила, для которых не известны все возможные примеры (как пример – оператор * в регулярных выражениях).

Виды правил в движке чатбота

1) "**comprehension rules**" – срабатывают до того, как текст реплики начинает обрабатываться классификаторами; позволяют привести различные перефразировки к нормальному виду, например "*сообщи твое имя*" = "*как тебя зовут*"

2) "**instead-of rules**" – в случае срабатывания предотвращают дальнейшую обработку в пайплайне, позволяют задавать особую реакцию на какие-то фразы вместо более общей стратегии на базе классификации интентов.

3) "**after rules**" – срабатывают после того, как отработали все шаги пайплайна. Например, если пользователь сообщил, что любит музыку – можно запустить сценарий разговора на околомузыкальные темы.

4) "**smalltalk rules**" – особый вариант "after rules", описывающих реплики в ответ на какие-то ключевые фразы собеседника. Эти реплики добавляются в выходной буфер вместе с ответом, сгенерированным пайплайном бота.

5) **сценарии** – особые группы правил с задаваемым графом переходов для более глобального управления дискуссией;

6) **вербальные формы** – вариант сценария, в котором бот ожидает заполнения заданного набора слотов, задавая уточняющие вопросы.

https://github.com/Koziev/chatbot