Journey Experimenting Different Approaches in developing Conversational Agents

Points:

<https://blog.rasa.com/how-to-build-a-voice-assistant-with-open-source-rasa-and-mozilla-tools/>

STT (Speech to Text Module):

We selected Mozilla DeepSpeech Model as that was the only open source model available at that time. It was not able to recognize Indian accent. We downloaded a huge voice corpus from “Common Voice” and separated 80000 Indian voices. We were yet to train with Indian voice.

As STT response was very slow because it took around 40 sec to respond in performance desktop without GPU, training the model took a back seat. Reason was, response speed has to be increased first.

Then we tried Azure Speech API. Its performance was almost real time. As making our own STT was not our goal we continued with Azure API.

But the grand plan was to record all the speech templates from users in production, and take Azure STT as annotation for the recorded speech template and train our deepspeech STT model. This was never implemented.

Links:

DeepSpeech model: <https://github.com/mozilla/DeepSpeech/tree/v0.6.1>

Voice Data Set: <https://commonvoice.mozilla.org/en/datasets>

TTS (Text to speech):

Tacotron model was used for TTS. The pretrained model performance was already good. So we went with it.

Even in this case we ended up using Azure TTS. Reason was Azure TTS was taking punctuations into consideration while speaking. This was more human like than our Tacotron model.

If the sentence is too long Tacotron model will take lot of time or it will infinitely repeat same word again and again.

As Azure API were free because we were using Professional version of Visual Studio, we didn’t put much effort in refining the existing model.

Link: <https://github.com/mozilla/TTS/wiki/Released-Models>

Another big reason for going with Azure for STT, TTS was due to GDPR compliance. To avoid legal issues Azure is a better option.

Chatbot:

Use case novelty:

We humans are social beings. Every day we interact with people from different walks of life. They might be our friends, mentors, vendors, etc. When we have a conversation with them, it is not necessary that we make the first move. A friend might talk to us when he feels we are emotionally low, a vendor may talk if he feels he can make a sale, a mentor talks to us if he feel we are struggling with something. So, a person intervenes if he feels some value can be added to us or to him.

Our key differentiator is that, our bot has a personality i.e. vendor, mentor, etc. All conversations done by the bot are strongly driven by its personality but at the same time, bot is conscious about making the conversation engaging for the user and discourse being maintained all the time. The bot engages user by providing insights about the topics and entities as well.

In our case the emotions of the user along with his behavior on the website and mouse, keyboard clicks gives trigger to conversation model.

Based on those triggers, conversation model intervenes. The model will stick to its personality (teacher in our case) throughout the conversation.

Technology novelty:

Our use case has a unique requirement. Say a students is doing course in our website. On average course is going to be more than 3 months. During this entire duration conversation model will intervene based on requirement. Over a period of time of use the responses can be predicted. User should feel our bot as a companion but not as a robot.

So new contextual responses which cannot be predicted by user is important. For a person to update the models frequently to achieve this is a huge task.

So we wanted a framework which gives us flexibility to integrate state of the art end-to-end models for new response generation.

Reason why Rasa was selected as framework was because it is opensource and all data, model cab be on premise. It is not like Microsoft bot or Facebook wit where model lies in cloud.

Rasa working:

Rasa has 2 main components. NLU and NLG.

Whatever utterance user gives the intent and entity is classified using NLU (DIET classifier along with tokenizers and featurizers in our case).

Eg: I want to play ping-pong.

Intent is: user wants to play game

Entity is: ping-pong

One advantage with Rasa is, it can match similar utterances.

Eg: I am not feeling good. (This is in dataset)

Say user says “I am not at all feeling good”. Here rasa can understand that “I am not feeling good” and “I am not at all feeling good” are same statements.

So adding similar data for training can be avoided.

Now model classified intent of the user. So model need to take action based on user’s intent.

We already have responses for corresponding intents. So model uses policies like mapping policy, TED policy to match correct responses to the intent.

Rasa Link: <https://rasa.com/docs/rasa/installation/>

DIET classifier: <https://medium.com/the-research-nest/using-the-diet-classifier-for-intent-classification-in-dialogue-489c76e62804>

End-to-End model:

Open domain models are trained over general reddit conversations, tweets, etc. Here is model is trained on open chat data. So model can generate response wrt whatever we ask. It basically tries to generate contextual responses. We wanted such capability in our framework for new response generation.

But there was an unanswered question.

Can an open domain end-to-end model be made goal oriented?

No literature confirmed that it can be done.

Initially we came across “Google Meena”. Results published in paper were quite promising but as model was not open sourced. Model was also very heavy due to 2.2 billion parameters. So we dropped it.

Next we came across “DialoGPT”. Microsoft claimed that DialoGPT is state of the art for single turn conversation. We trained small model (due to lack of powerful GPUs) it for our single turn dataset and results were satisfactory.

DialoGPT link : <https://github.com/microsoft/DialoGPT>

Multi turn training method was given in the paper but code was not available. So we didn’t go further with DialoGPT, as multi turn was our requirement.

Then we came across OpenAI GPT. We wanted a lighter model to run in a performance desktop. Their GPT-1 small models meets our requirements. So we started experimenting with GPT multi turn.

GPT link : <https://github.com/huggingface/transfer-learning-conv-ai>

On our training set GPT was performing well and finally we were able to make open domain bot to goal driven bot.

Another point which we observed in GPT was that by switching personality, responses can be very interesting.

Eg. For “hi” by user, “I am maya, your teacher assistant” is expected response when personality is “teacher”.

Now if we change the personality to “I am your high school sweet heart” the response generated was “I am maya, your high school sweet heart”

Switching personality occasionally will get the user interested as responses are very new and sometimes funny.

Integration of Rasa and GPT:

We used Rasa as the master in our conversation framework. Though we have a trained goal driven end-to-end model, for some situations like pulling information from database, from cloud or handling chit-chat (it can be done but it is not scalable because it will turn into open domain bot) we cannot rely on it.

So basically rasa was used to control GPT and the entire conversation experience.

Any utterance from user will first go to Rasa. Rasa will then classify whether the utterance is “an information request for database”, “information request from cloud”, “chitchat” or the story line.

As long as the utterance is related to story line Rasa sends it to GPT for handling. GPT responds to all the conversations. During the entire flow Rasa has no idea wrt what GPT has responded. It only knows that GPT has responded,

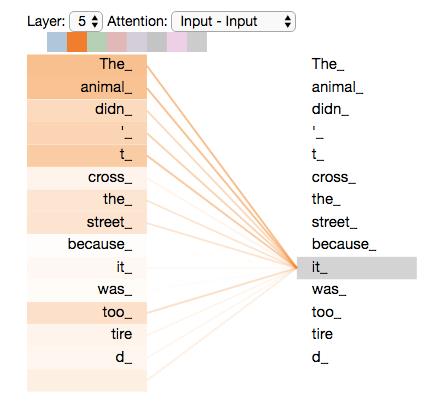
Suddenly if user tries to chit-chat, Rasa takes control of conversation and tries to pull user back to context. During this time GPT has no idea that Rasa is responding and does not know what Rasa has responded.

For pulling information over the web Google Knowledge Graph was used.

As architecturally both the models are kept independent, say a new model is available it can then be easily replaced.

Transformers:

We selected Generative Pre-trained Transformers (GPT) from Hugging Face. The main reason behind the selection is because of the Attention Mechanism.



Attention mechanism helps model to associate “it” in the sentence to “The animal”. We wanted such kind of strength in our conversation agent as well.

Every time Transformers model takes in the context and generates the response. So here in the context we gave sentences describing personality of the Teacher and history of conversations between user and the GPT model every time to generate next response. This idea is inspired from Persona-Chat dataset.