**Never Stop Learning !**

Building a continuous retraining pipeline for conversational AI Assistants

Introduction

The exponential progress in the development of deep learning-based language models has affected every possible natural language processing (NLP) and natural language understanding use-case and conversational AI is no exception for that. In the past conversational agents were solely able to only answer prefixed FAQ questions or provide a simple button interface for the users to select a set of predetermined answers. Their backend logic was either completely rule based or hardcoded with a set of possible conversational paths, these chatbots were not able to answer questions contextually or engage in an actual sequential conversation with the user to perform a task.

Conversational AI has come a long way since then and today’s conversational AI assistants that operate on the level 3 intelligence tier can answer even vague inquiries with a high-level of confidence and are applied in numerous industries ranging from recruitment to end-to-end customer support, enabling users to apply for a vacancy or purchase a product just by having a conversation. And the development of present conversational AI assistants are also more inference based focusing on the users present context in addition to multiple factors like long-term conversational history before generating a single response to the user.

But one of the major problems in taking conversational AI solutions to production is that over-time when the assistant receives more and more messages, the performance of the assistant degrades in a slow linear fashion, commonly known as model-drift. This is because the language understanding models of the bot are operating based on weights trained on conversations that happened a long time back and if the domain of the bot has changed, for example, in the context of a recruitment chatbot if a totally new vacancy was added to the vacancy database by the HR team recently, and if a candidate asks the conversational assistant about the requirements for that vacancy, the assistant would not be able to understand the user’s inquiry with a high level of confidence because it has not been trained on conversations referring to that particular new vacancy.

And the best solution for model-drift is continuous retraining of the bot, so that over time the bot not only adapts to a changing domain but also learns from past user conversations to increase it’s language understanding accuracy over time.

In this book we will go through how to construct a continuous retraining pipeline that will enable a conversational AI assistant, to learn from the mistakes made on past conversations and keep improving the user-interaction experience over time.

The 5 levels of intelligence in conversational AI

The 5 levels of intelligence in conversational AI is a benchmark metric developed by the Rasa to measure, the level of variability and complexity in conversational paths that a chatbot can handle. The higher the level of intelligence a chatbot operates on the lesser the information the user must know to utilize the chatbot. We will go through each of the 5 levels one by one .

Level 1 : This is most basic format a user could interact with a computer. For example, command line applications are a good example for this category. At this level of intelligence, we put the burden on the end-user to actually know exactly what he is looking for and also to know the exact commands he has to use to get the information he is looking for. For example: select \* from products

Level 2 : Level 2 assistants are basic FAQ bots, they can answer a prefixed set of questions with same answers and in rare cases can follow up with a couple of questions. The disadvantage in level 2 assistants is that the user’s inquiries should match what the bot is expecting, for example if the bot asks “what is your monthly salary?” , then the response it’s expecting is an integer like ”$50,000” but if the user says “My salary is fifty thousand dollars”, the bot would ultimately fail and ask the user to repeat the answer

Level 3 : Level 3 is where AI comes into play in the context of conversational assistants. Chatbots operating on level 3 intelligence can understand users and respond contextually. And most importantly they can handle deviations in the conversational path by the user. For example after the bot asks “Can I know your name?” and the user says ,“why do you need to know my name?”, a contextual bot will be able provide the reason to the user why their name is needed. These assistants can understand the user’s context and handle deviations in the conversational flow smoothly by utilizing deep learning architectures like LSTMs which can retain the long term context of a sequence , to classify the intents and entities in each user message. Intents represent the intention of the user for a given message. For example in the message “I want to book a reservation at 3:00pm”, the intent is that the user wants to book a reservation and the entity here is the time slot “3:00pm” which is a key piece of information present within the message that the bot would require to perform an action. This is presently the highest level of intelligence chatbots in practical use-cases have attained and we too, will be focusing on developing a retraining pipeline for a chatbot operating in this tier of intelligence.

Level 4 : Level 4 assistants can understand a user’s inquiry and suggest them solutions, without the user having to explicitly ask the right questions. For example the user can say “Tommorow I would have to travel outside the city for a meeting in the evening” and then the bot would say “Cool, shall I book a room at ABC Hotel tommorow? ”

Level 5 : Level 5 assistants are adaptive, these types of agents can guide the user through a conversational path by understanding the user’s needs without the user having to mention them. For example “Why do you think a ABC hotel is good?” , the assistant would then provide the reason on why it chose “ABC hotel” as the best place book a room. This type of conversational assistants are still only in research and aren’t used for practical use-cases.

An intro to the Rasa framework

As simple as it may sound, developing a contextual conversational assistant is tough!

In most computer vision or NLP problems one model architecture can be ideally used in a straightforward manner to solve the problem, for example like detecting an object in an image. But in when building contextual conversational assistants developers need to consider a range of models to use, for example, there are independently different model architectures used for intent classification, entity recognition and dialogue management. Usually, these models are trained individually and then are aggregated together to perform tasks like intent classification and entity recognition simultaneously on an input message. And training , validating and running inference on these models in addition to building consistent data pipelines can quickly become complex as the number of conversational flows needed to be handled by the chatbot starts to increase.

Rasa is an open-source conversational AI development framework that is completely focused on simplifying the complexity of handling multiple models and defining conversational flows when developing chatbots. In rasa you can train a basic chatbot by just defining some sample conversational flows in yaml and since Rasa is completely Python based you can also define custom actions, which can be a programmed backend action like sending a request to an API to book a room. This simplicity and flexibility are what allows rasa to be used in both simple and heavily complex contextual chatbots. In this book, we will primarily use Rasa to build a basic recruitment chatbot for which we will be constructing a continuous retraining pipeline for.

Model drift and continuous learning

What is model drift?

Model drift is the degradation (or drift) in accuracy for a model that has been in production for a long time, the main root cause of model drift is the change distribution of input data which in turn effects the statistical relationships between the features of the dataset and the predictions made, this can be due to various environmental or seasonal factors. When this change in relationship is not reflected in a model’s weight parameters a gradual drop in a model’s performance starts to take place.

How can continuous retraining save the day?

There are different strategies that can be used to solve model drift depending on the context of the ML problem, for example in computer vision model drift can be solved by periodically analyzing the accuracy of the model overtime and then retraining the model or updating the architecture. In the context of conversational AI model drift the can be detected by analyzing the rate of intent misclassifications and number of times the “fallback” intent is triggered. (Fallback intent is the intent which is triggered when the model cannot confidently classify a message into any of the intent categories). And the best solution for model-drift, is to periodically retrain the bot from “fallback” conversations, so that the weights of the intent classification model in the chatbot are adjusted to the changing dynamics of user’s conversational flows.

Building your first conversational AI assistant using rasa

Getting Started

Intents and Entities

Getting your stories right!

Custom actions

Collecting user information using forms

Handling (and saving) fallbacks

### Build a retraining pipeline using only two stage fallback conversations, store the intents and ### the fallback and then create a yaml file from that to train the bot. Use dvc to push and pull ### the dataset containing the generated nlu file from the two stage fallback data.