Intent Sets - Architectural Choices for Building Practical Chatbots

Saurabh Srivastava
ssri@iitk.ac.in
Indian Institute of Technology
Kanpur, India

TV. Prabhakar
tvp@iitk.ac.in
Indian Institute of Technology
Kanpur, India

Abstract
“Chatbot” is a colloquial term used to refer to software components that possess the ability to interact with the end-user using natural language phrases. Many commercial platforms are offering sophisticated dashboards to build these chatbots with no or minimal coding. However, the job of composing the chatbot from real-world scenarios is not a trivial activity and requires a significant understanding of the problem as well as the domain. In this work, we present the concept of Intent Sets - an Architectural choice, that impacts the overall accuracy of the chatbot. We show that the same chatbot can be built choosing one out of many possible Intent Sets. We also present our observations collected through a set of experiments while building the same chatbot over three commercial platforms - Google Dialogflow, IBM Watson Assistant and Amazon Lex.

CCS Concepts: • Software and its engineering → Software architectures; Software design tradeoffs.

Keywords: Intent Sets, Chatbots, Architectural Choices

1 Introduction
Chatbots (or bots in short) are software components which can interact with users in English or any other natural language. These bots are becoming increasingly popular. Many commercial platforms provide a dashboard to build these bots. These dashboards provide the required tools to define, test, modify and deploy these bots with no or minimal custom coding. The job of a bot involves accepting a user query, and replying with a suitable response. These platforms usually expect the bot details to be expressed using four key pieces of information. In the absence of standard terminology, we refer to it as the Intent-Entity Paradigm [7].

• **Intents:** The platform categorises the query in one of the predefined classes, called Intents. These classes roughly represent the types of queries that this chatbot is supposed to handle. For example, if we are building a bot to answer sports queries, possible Intents could be football_query, cricket_query, chess_query etc. These platforms usually have a default Intent too (i.e. the "Others" category) which is used if the query cannot be matched to any defined Intent.

• **Entities:** The platform can be trained to look for specific information within a query called Entities. Entities represent real-world objects or their specific properties. For instance, to answer a Cricket query about the runs a particular player scored in a calendar year, values for two Entities, namely Player name and Year would have to be provided by the user. Entities, when they are bound to a particular Intent, are also known as Slots or Parameters. These platforms usually provide prompts - predefined classes, called Intents - to fill values for any required Slots (e.g. “For which calendar year?”). Some platforms also allow setting default values for the Slots, in case the user doesn’t provide a value explicitly.

• **Fulfilment Logic:** For each query, the bot must provide a response to the user. This response depends on the Intent associated with the query and the details provided by the user as the part of Slot-filling process. Based on this information, the platforms provide the developers to construct a response to be sent to the user. This process is termed as the Fulfilment for a specific combination of Intents and Slots. The responses can be supplied as templates with placeholders (e.g. “$player_name scored $fetched_runs in the calendar year $year”) or they can be generated arbitrarily on a case-to-case basis.

• **Sample Queries:** The platform expects the developer to provide some examples of the questions a user may ask. We refer to them as Sample Queries. These examples make up the training data that the platforms use to train their models internally (the details of the training process are...
The lump-sum accuracy of a bot can then be measured by firing some unseen queries at it and measuring the number of times it mapped the query to the expected Intent. Another dimension to evaluate the bot is its ability to parse parameter values, with the least number of prompts.

The first step, thus, in the process of building a bot, is to come up with appropriate Intents to cover all possible types of queries that the user may ask. Next, there needs to be an understanding of the Entities that will crop up in the conversations, and relate them with the respective Intents. Finally, a set of Sample Queries for each defined Intent needs to be collected, and supplied to the platform, before it can be tested. We do not put our focus on Fulfilments, assuming that the same involves applying business logic to prepare a response, a task that is independent of the chatbot itself. There are two natural questions that arise out of this process. First, can there be more than one way to pick the Intents and Entities for the same use case? Second, if there are indeed multiple choices available, does the accuracy of the bot gets affected when choosing one option over others? In this work, we attempt to look at these questions in greater detail. We name these choices as Intent Sets, and comment upon their properties with the help of some experiments.

The rest of this paper is structured as follows. We begin with mentioning some related literature in Section 2. We then introduce the concept of Intent Styles in Section 3. This followed by observations from some experiments that we performed by building and evaluating a sample bot over three commercial platforms in Section 4. Finally, we proceed to conclude the paper citing future possibilities in Section 5.

2 Related Work

Brandtzaeg et al. [2] provide an overview of how we need to rethink our user interfaces in the wake of the Chatbot revolution. Researchers have started investigating these changes ([3] [13] etc.), as more systems are built with a "communicational interface" ([14] [5] [8] etc.). The success of Chatbots is fueled by advances in Neural IR, which deploys Deep Learning methods internally. These methods are thoroughly surveyed in multiple works ([6] [9] etc.). However, the models built this way are hard to explain, change or tune. This is why the Chatbots built by commercial platforms have unpredictable behaviour. The efforts for investigating Information Retrieval with Chatbots, where the source data is in the form of documents are already ongoing ([15] [11]). Some researchers are also working on a close, but a different task of generating textual responses from tabular data ([1] [12] etc.). However, these works focus on text generation rather than looking into the issues that arise when a commercial platform is used to build the conversation interface.

To the best of our knowledge, our work is the first to address architectural issues with solutions involving the use of chatbots. We have previously investigated [10] a formal approach towards picking a chatbot platform, given a set of quality attributes to achieve. This work intends to continue efforts towards bringing out architectural abstractions to support the use of chatbots in practical use cases.

3 Intent Sets

We now provide a semi-formal definition of Intent Sets as:

An Intent Set is a collection of Intents and their associated Parameters, which can collectively cover all possible queries that a chatbot may encounter.

An Intent Set categorises any possible natural language phrase into one of the pre-defined categories, i.e. assign an Intent to it. From the outset, it might seem a challenging task to come up with a set of Intents, that can cover any user
input. However, commercial platforms provide a default Intent pre-added to a new Chatbot project, which serves as an “others” category for the queries. This ensures that the architect can handle queries which are irrelevant or ambiguous in a graceful fashion (similar to a catch block that follows a try block in many programming languages). So, the task of defining the Intents can be done incrementally, by creating Intents in the order of their priority, leaving the rest to be handled by the default Intent.

In any case, modelling a real-world use case with the Intent-Entity paradigm is not straightforward. As an example, consider the Cricket Statistics Website called Statsguru [4]. The website can be browsed to get peculiar details from the world of Cricket about players, teams, venues etc. However, the website provides a tradition Query-By-Example (QBE) styled interface for filtering the required information. Consider the problem of building a chatbot, which can accept queries in the form of a question, “map” it to a structured form, say an SQL statement or a Relational Algebra (RA) expression, pull out the required information and present it back to the user as a response. The difference between the QBE interface and a chatbot is highlighted in Figure 1.

For simplicity, we assume that the user asks about only one piece of information per query. The QBE interface provides hints on how to categorise a query in one of the pre-defined sets, i.e. Intents. For example, a query may be asking a stat about a “player”, a “team” or a particular “match”. We can thus create three Intents for our chatbot, and along with the default Intent, we should be able to cater to any possible query that the chatbot may receive. The finer filters that we see in the QBE interface need to be abstracted to Entities and mapped to define Intents as a set of Parameters required to prepare a response. Similarly, there can be another dimension to categorise the queries - “batting”, “bowling” or “fielding” queries. Figure 2 shows how two Intent Sets can be “equivalent” to each other. It can be gleaned that the overall information required to process a query remains the same. What differs from one Intent Set to another, is the top-level categorisation of the query, and associated Parameters with the Intents. We now list some properties of Intent Sets:

- If the platform maps every query to the correct Intent, and parses all the Parameter values correctly, then the response produced by any Intent Set for a given chatbot is exactly the same. This may seem counter-intuitive to the current work, as it makes the study of Intent Sets uninteresting. However, no platform can practically have a 100% accuracy for either of these activities, meaning in practice, different Intent Sets would have different accuracy.
- An Intent Set divides the real-world into a finite number of disjoint scenarios, the union of which, constitutes all possibilities that the chatbot can handle. The important term here is disjoint. For example, IS$_1$ in Figure 2 assumes that every
query can be about (at most) one of the three - teams, players or matches. However, if it is theoretically possible to formulate a query that can be simultaneously, say, a “team” as well as an “individual” query, then IS$_1$ ceases to be a valid Intent Set for the chatbot.

Besides, if the chatbot only serves the purpose of providing a conversational wrapper over some multi-dimensional data (e.g. for Statsguru [4]), there are some derived properties for these Intent Sets:

- **Only nominal attributes can be used to create Intent Sets.** This is rather straightforward. Only an attribute that can take a fixed number of possible values can be used to create finite disjoint partitions of the data. If required, non-nominal attributes can be grouped into categories, by defining ranges. However, platforms usually provide better support modelling such attributes as Entities.

- **The queries that the chatbot can handle must be atomic for the attribute used to create the Intent Set.** This follows from the fact that the platform categorises the query in one and only one out of the defined Intents. Formally, if the RA expression that fetches the data looks like:

\[ \Pi(A_1, A_2, A_3, \ldots) \sigma(A_i = x \land A_j = y \land A_k = z \ldots) \]

and, the Intent Set was created with the attribute $A_i$, then one and only one condition involving $A_i$ could be part of the RA expression.

### 4 Experimental Observations

As discussed in Section 3, the activities of Intent matching and Parameter parsing involve Natural Language Processing (NLP) tasks, where errors cannot be avoided fully. The platform may not map some queries to their correct Intents and may mess up the parsing of Parameters as well. Platforms usually keep the details of these activities “blackboxed”, so there is no way to predict these failures beforehand. To observe how the choice of picking one Intent Set over the other may affect a bot’s accuracy, we performed a set of experiments. We built a bot called Cricket Novice, which can answer basic questions about the game of Cricket. It can also answer some statistical questions about players in International Cricket, such as those with best average or strike rate. We describe our experimental setup briefly:

- We created **25 Descriptive** Intents with a static response. These Intents did not have any Parameters, as they only informed the user about the basics of the game. For instance, these Intents answer queries such as “How is Cricket played?” or “What is the role of an Umpire?”.

- We hand-picked 35 rows of statistics about individuals from Statsguru [4], and stored them in a table. So, Cricket Novice also acts as a conversational wrapper to extract stats out of this table. The structure of this table and some of its rows are shown in Table 1.

<table>
<thead>
<tr>
<th>Match Type</th>
<th>Stat Type</th>
<th>Stat Type 1</th>
<th>Stat Type 2</th>
<th>Stat</th>
<th>Player</th>
<th>Team</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>test innings</td>
<td>highest batting</td>
<td>runs</td>
<td>Brian Lara</td>
<td>West Indies</td>
<td>scored 400* runs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>odi career</td>
<td>highest bowling</td>
<td>wickets</td>
<td>Muttiah Muralitharan</td>
<td>Sri Lanka</td>
<td>took 534 wickets</td>
<td></td>
<td></td>
</tr>
<tr>
<td>odi career</td>
<td>lowest bowling</td>
<td>strike rate</td>
<td>Rashid Khan</td>
<td>Afghanistan</td>
<td>bowled 2623 deliveries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>test career</td>
<td>highest batting</td>
<td>average</td>
<td>Sir Donald Bradman</td>
<td>Pakistan</td>
<td>had an average of 99.94</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t20 innings</td>
<td>highest batting</td>
<td>strike rate</td>
<td>Dwayne Smith</td>
<td>West Indies</td>
<td>had a strike rate of 414.28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t20 career</td>
<td>highest batting</td>
<td>centuries</td>
<td>Rohit Sharma</td>
<td>India</td>
<td>scored 4 centuries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t20 career</td>
<td>lowest bowling</td>
<td>average</td>
<td>Rashid Khan</td>
<td>Afghanistan</td>
<td>has an average of 12.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>test career</td>
<td>lowest bowling</td>
<td>economy rate</td>
<td>Daniel Vettori</td>
<td>New Zealand</td>
<td>had economy rate of 5.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>test innings</td>
<td>best bowling</td>
<td>figures</td>
<td>Jim Laker</td>
<td>England</td>
<td>had figures of 51.2-23-10-53</td>
<td></td>
<td></td>
</tr>
<tr>
<td>test career</td>
<td>highest batting</td>
<td>centuries</td>
<td>Sachin Tendulkar</td>
<td>India</td>
<td>scored 51 centuries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>test career</td>
<td>highest batting</td>
<td>double centuries</td>
<td>Sir Donald Bradman</td>
<td>Australia</td>
<td>scored 12 double centuries</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 1.** Some of the stats known to the Cricket Novice. The shaded columns are used by the bot to stitch a response.

- We then created two Intent Sets with **Statistical** Intents to cover all the user queries that Cricket Novice could be asked. These Intent Sets varied in the number of Intents, as well as their Parameters. In many cases, even a genuine Cricket query was supposed to be handled by the default Intent due to lack of data in our table. Figure 3 provides an overview of the two Intent Sets used for our experiments.

- For every row in the stats table, we created 3 sample user queries which the user could ask to retrieve that stat. For example, “Who has the highest batting average in Test matches?” and “Which player holds the record for the best batting average in Test cricket?” are formulations of the same query. These Sample Queries are used to train the platform for the Intent matching step. We created similar formulations for the descriptive Intents as well.

- We picked three platforms - Dialogflow, Watson Assistant and Lex, and created two versions of Cricket Novice on all of them. The **Descriptive** Intents remained the same for both versions. The **Statistical** Intents were created separately using the Intent Sets shown in Figure 3. We
Figure 3. Statistical Intents for Cricket Novice. The Descriptive and default Intents were common.

- We added the Sample Queries for Statistical Intents in phases. During Phase 1, we only provided one-third of the Sample Queries for each Intent. In Phase 2 and 3, we added the rest of the two-third formulations.
- We also added a set of 10 negative examples in Phase 4, to aid triggering of the default Intent. These queries were meant to reduce the false-positives, i.e. the cases where the query was mapped to one of the defined Intents when ideally, it should have triggered the default Intent.
- We created a set of 20 unseen queries, which were fired at the bot after each phase and observed their behaviour. If the bot could match the query to the correct Intent, as well as parsed all the Parameters correctly, we called it a success (a score of 1). If the matched Intent was correct, but the bot botched up the Parameter-parsing, we called it a partial success (a score of 0.5). Otherwise, we called it a failure (a score of 0). For each phase, we summed up scores for all 20 queries as the Utility Score of the bot.
- We also performed an Ordering Experiment over the bots created on Dialogflow and Watson Assistant. In this experiment, we evaluated the states of the bots after Phase 2. These states were reached via two different ways - by adding more Sample Queries to Phase 1, as well as removing Sample Queries from Phase 3. In both cases, the final set of Sample Queries available to the platform was the same; however, the order in which they received them changed. This experiment was done because, in an attempt to provide a better testing experience, Dialogflow and Watson Assistant re-train their models even on slightest changes in Sample Queries. This hinted towards an incremental approach towards building the model, meaning that the order in which these platforms received the queries, affected the final model. Lex, on the other hand, allows users to make all changes to the bot, and then explicitly re-build the model.

Figure 4 shows our observations in a nutshell. They can be summarised as:

1. The two Intent Sets produce significantly different results. The Utility Scores showed variations for the same bot across different phases, Intent Sets as well as platforms.
2. The accuracy of the system generally got better with the additions of more Sample Queries.
3. Except for Watson Assistant, Intent Set\(_2\) seemed to be performing better than Intent Set\(_1\).
4. There was no conclusive inference on the effect of counterexamples on the overall accuracy of the system.
5. During the Ordering Experiment, we observed that the bot built over Dialogflow showed significant variations...
when the order of the samples was changed for both Intent Sets. The bot built over Watson Assistant showed minor variations with Intent Set 1, but no variations were seen with Intent Set 2.

We must clarify that these observations are not evidence of the superiority of one Intent Set over others, or one platform being better than others. These observations are specific for the set data we used to build our bot, and cannot be generalised. The notable point here is that picking of one Intent Set over others changes the behaviour of the bot significantly.

5 Conclusion and Future Work
In this work, we presented the concept of Intent Sets, an architectural choice, which can impact the accuracy of a chatbot. We discussed our observations noted while creating an experimental bot using two Intent Sets over three platforms - Dialogflow, Watson Assistant and Lex.

We briefly discuss some future dimensions for our work: First, for chatbots that work over tabular data; there can be efforts at “predicting” if one Intent Set is better than another, based on a statistical analysis of the data. One exciting dimension to explore is the relationship between the number of possible values for an attribute and the Intent Set that is built using it. Another dimension could be performing similar experiments for more complex data, and provide helpful guidelines to practitioners about picking an Intent Set over others. Second, there can be efforts to comment on the “blackboxed” platforms, by observing their behaviour over multiple Intent Sets for the same problem. This may help in picking the right platform for the right problem.

References