Enhancing Community Interactions with Data-Driven Chatbots – The DBpedia Chatbot

Ram G Athreya
Arizona State University
Ram.G.Athreya@asu.edu

Axel-Cyrille Ngonga Ngomo
Data Science Group, Paderborn University, Germany
axel.ngonga@upb.de

Ricardo Usbeck
Data Science Group, Paderborn University, Germany
ricardo.usbeck@upb.de

ABSTRACT

In this demo, we introduce the DBPEDIA CHATBOT, a knowledge-graph-driven chatbot designed to optimize community interaction. The bot was designed for integration into community software to facilitate the answering of recurrent questions. Four main challenges were addressed when building the chatbot, namely (1) understanding user queries, (2) fetching relevant information based on the queries, (3) tailoring the responses based on the standards of each output platform (i.e. Web, Slack, Facebook) as well as (4) developing subsequent user interactions with the DBPEDIA CHATBOT. With this demo, we will showcase our solutions to these four challenges.

KEYWORDS

DBpedia, chatbot, knowledge base, question answering

ACM Reference Format:
Ram G Athreya, Axel-Cyrille Ngonga Ngomo, and Ricardo Usbeck. 2018. Enhancing Community Interactions with Data-Driven Chatbots – The DBpedia Chatbot. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3184558.3186964

1 INTRODUCTION

Growing domain-specific communities (e.g., the DBpedia [6] community)[1] are often characterized by new users asking previously answered questions upon joining the community. Mailing lists, forums (e.g., Facebook, Twitter) or the public Slack channel and FAQs are examples of the types of measures that are often put in place to mitigate this common problem. However, these solutions are static and fail to interact with new community members. We address this interaction issue for the DBpedia community by presenting the DBPEDIA CHATBOT, which leverages existing communication sources (between users) as well as DBpedia to enhance community interactions in the DBpedia community.

In this demo, we will 1) describe and showcase the idea of technology federation through a chatbot to facilitate more efficient communication in a community via examples run on laptops, 2) we will ask users for their feedback to enhance the community value and 3) show preliminary evaluations of the DBPEDIA CHATBOT.

The contributions of this demo are as follows:

[1]http://wiki.dbpedia.org/about/dbpedia-community-0

(1) We implemented a baseline approach to intent classification for technology federation,
(2) We implemented a rule-based approach to answer questions related to the DBpedia community,
(3) We combined several tools for answering factual questions,
(4) We expose the research work being done in DBpedia as product features. For example:
   - **Genesis** [5]: We use the APIs from the Genesis project to show similar and related information for a particular entity.
   - **QANARY** [4]: We use WDAqua's question answering approach to answer factual questions that are posed to the DBPEDIA CHATBOT.

Our implementation is open-source and a screencast as well as source code can be found at **https://github.com/dbpedia/chatbot**. A live demo can be found at **http://chat.dbpedia.org**.

2 RELATED WORK

One of the early chatbots was ELIZA [11], which was designed around a pattern matching approach. ELIZA was intended to sound like different persons, e.g. like a psychotherapist. Later, the ELIZA-inspired chatbot ALICE [10] was introduced. ALICE was implemented based on a scripting language called AIML[2], which is next to RiveScript[3], which is currently among the most used languages for chatbots. More recent chatbots, such as Rose, combine chat scripts with question understanding to better mimic domain experts [1]. Recent approaches have also considered tackling the combination of RDF and chatbots, e.g., [2, 3, 9]. Due to the lack of space, we refer the interested reader to [1, 7] for further references.

3 APPROACH

Our architecture is based on a modular approach to account for different types of user queries and responses, as can be seen in Figure 1. The DBPEDIA CHATBOT is capable of responding to users via (1) simple short text messages or (2) through more elaborate interactive messages using the underlying knowledge graph. Users can communicate with the DBPEDIA CHATBOT through (3) text but also through (4) interactions such as clicking on buttons or links as well as giving feedback. Furthermore, we focused on designing this bot so that it can be connected to a multitude of platforms to increase outreach to the community. The core of the DBPEDIA CHATBOT is based on the Spring framework[4] and can run on a single core machine or be distributed across a cluster. Incoming user feedback is stored in Couch DB in an anonymous way for further research .

[2]http://www.alicebot.org/aiml.html
[3]https://www.rivescript.com
[4]https://spring.io/

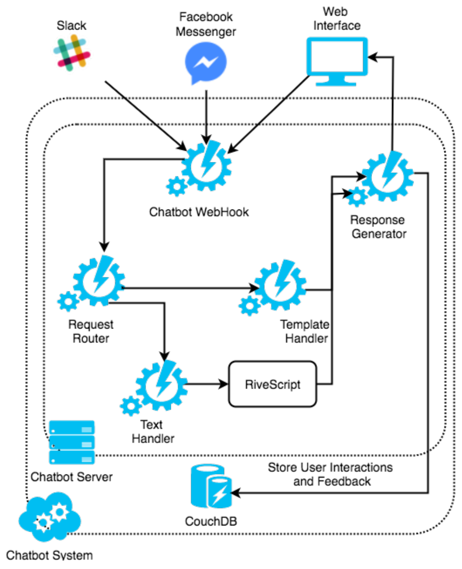


Figure 1: Architecture of DBpedia Chatbot.

In the following sections, we explain our approach in detail. We will also demonstrate all possible interactions at the demo booth.

Query Lifecycle. We devised a modular pipeline to interweave novel knowledge base technologies, in this case DBpedia-related, into our approach. The process by which the DBpedia Chatbot handles incoming requests can be divided into four steps:

(1) **Incoming Request:** For each platform, a webhook handles the incoming request and forwards it to the routing module.
(2) **Request Routing:** Incoming requests are routed based on their respective type. Pure text requests are handled by the Text Handler while parameterized requests are handled by the Template Handler.
   - **Pure Text Requests:** A pure text request is basically a text message from the user. We use RiveScript to identify the intent of the message and classify it into the following types (see Section Intent Classification): DBpedia Questions (rule-based community questions, service checks, language chapters, etc.), Factual Questions (including location questions) and Banter.
   - **Parameterized Requests:** These are triggered when users click on links in information already presented (e.g., "Learn More" button when presented with information about a particular resource).
(3) **Generate Response:** The response from the handlers is converted to a format that is suitable for each platform.
(4) **Send Response:** Finally, the response is sent back to the respective platform. Additionally, for the Web interface, the client side code to handle the responses is written using standard front-end technologies such as HTML, CSS and Javascript.

Intent Classification. Intent classification is the first step while processing natural language queries since there are several kinds of questions that the DBpedia Chatbot needs to handle. All natural language questions are passed to RiveScript which was developed upon the underlying communication protocols for intent classification. Based on contextual rules defined in RiveScript, the incoming request is classified into the categories aforementioned and passed on to respective request handlers. Algorithm 1 shows the mechanics of the intent classification process and Figure 2 shows the overall workflow of how a query is internally processed by the chatbot. The different handlers and the APIs that support them are explained later.

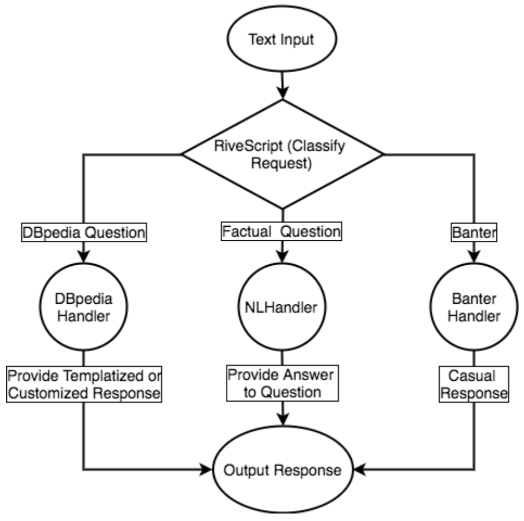


Figure 2: Query Fulfillment Workflow.

Algorithm 1 Intent Classification Pseudocode

```
1:  Consider user query $Q$
2:  Intialize RiveScript variables
3:  Sanitize common misspellings in $Q$ using JLanguageTool
4:  Apply substitutions on $Q$ {Eg: I'm => I am}
5:  Load all RiveScript rules to memory
6:  Sort rules based on priority, rule structure, position of wildcards, etc.
7:  for all $rule$ in $rules$ do
8:      if $Q$ matches $rule$ then
9:          return $rule->classification, rule->args$
10:     end if
11: end for
```

Rule-Based Community Questions. The DBpedia Handler is tailored towards conversations around the DBpedia community and based on existing useful background conversations. Here, we explored DBpedia's mailing lists to answer DBpedia-related questions via a rule-based approach.

Data Sources. The official mailing lists of DBpedia were chosen as the primary data source for this task. This includes the DBpedia Discussion[5] and DBpedia Developers mailing lists[6]. Conversational threads from the mailing lists were extracted to find interesting

[5]https://sourceforge.net/p/dbpedia/mailman/dbpedia-discussion
[6]https://sourceforge.net/p/dbpedia/mailman/dbpedia-developers

question-answer pairs that could be used for creating conversational scenarios for the chatbot. This dataset was augmented with some conversations from Slack that were handpicked in an ad-hoc manner.

Data Cleanup Tasks. The mailing list dump (mbox file)[7] was taken as an input and pre-processed to remove undesired messages based on the criteria mentioned in subsequent sections. The result from pre-processing was stored in a JSON file with the key being the original message subject and all associated messages were stored as an array for further processing. Several threads were excluded or sanitized in the final dataset, including the following criteria:

- All messages that are request for comments, call for papers, announcements etc.
- Messages that do not have question words in their subject or body. Question words considered were: What, When, Why, Which, Who, How, Whose, Whom.
- Words such as reply, fwd etc from the subject
- Reply sections to reduce redundancy
- Unnecessary HTML tags, whitespaces, new lines, etc.

TF-IDF Vectorization. The remaining messages were vectorized for use in the latter clustering step.[8] Then the subject of each message was tokenized and stemmed using the Porter Stemmer within the Pandas framework.[9] This stemmed output was used as input to a TF-IDF vectorizer to convert the text input to a matrix array containing frequencies of each term in every message. The total number of extracted features was 135.

K-Means Clustering. The TF-IDF vector was passed as input to a K-Means algorithm to cluster interesting topics using cosine similarity which formed the baseline for our rule-based conversation module. Some of the major categories that were identified and clustered through the algorithm are: 'About DBpedia', 'DBpedia Lookup'[10], 'DBpedia Datasets Download/Dump', 'DBpedia Releases' and 'DBpedia Extraction Framework'.

Rule-Based Conversation. The topics and threads that were derived from the previous steps were converted into rules in RiveScript in the form of patterns called *triggers*. These triggers are similar in structure to regular expressions but have more sophisticated and expressive features such as variable output, redirections between rules, dialogue blocks for continued conversation, etc. A typical (simple) rule has a structure akin to that shown in Equation 1. The first line specifies the input pattern that needs to be matched. The second line specifies the response from RiveScript, which is then further processed by the request handlers aforementioned. This particular rule can be used to answer queries such as *"How can I contribute to DBpedia?"*, *"Get involved with DBpedia."*, etc.

$$\begin{gathered} +[how][*](contribute||contributing||involved)[*]dbpedia[*] \\ -\{"type" : "template", "params" : "dbpedia - contribute"\} \end{gathered} \tag{1}$$

[7]https://github.com/dbpedia/dbpedia-chatbot-data

[8]A converted CSV file can be found at https://raw.githubusercontent.com/dbpedia/dbpedia-chatbot-data/master/data/dbpedia-discussion-archive-dataset.csv

[9]https://pandas.pydata.org/

[10]http://wiki.dbpedia.org/projects/dbpedia-lookup

Query Fulfillment. Based on the intent detected in the previous step, an appropriate request handler is triggered. Within the context of DBpedia-related questions, the DBpedia handler generates a response which is usually a templated answer based on the scenario. However, for certain queries such as *"Is DBpedia down right now?"* additional steps are carried out by the DBpedia Chatbot to check whether the DBpedia website and SPARQL endpoint are working and the result of this finding is returned back to the user as the response.

Natural Language Questions. To answer natural language or factual questions, the bot uses currently a combination of WDAqua's QANARY [4][11] question answering system and WolframAlpha. [12] The responses from QANARY are typically DBpedia URIs or RDF literals[13] with proper semantic annotation, while responses from WolframAlpha typically need to be linked to DBpedia. The chatbot uses DBpedia Spotlight [8] and the DBpedia Lookup service for this purpose. The results are cascaded based on priority levels with a higher priority given to RDF literals since they are typically a precise answer to a question. A lower priority is assigned to DBpedia URIs if both RDF literals and URIs are generated as candidate results for a query. Finally, the responses from both systems are amalgamated using a UNION strategy and the results are presented to the user. Additionally, for location-related questions the DBpedia Chatbot employs OpenStreetMap[14] data to acquire relevant geographic information.

Knowledge Cards. The DBpedia Chatbot enriches RDF URIs from the responses for users and displays important attributes about the respective entity. These knowledge cards are designed similar to the infoboxes shown in Wikipedia. This feature is based on the underlying knowledge graph, a list of all classes in the ontology namespace.[15] Note, this feature is agnostic to any RDF knowledge base. For a given class, we found the total number of occurrences of that class in the knowledge graph. Then, we extracted all `rdfs:domain` properties for the respective class and calculated the number of distinct occurrences of each individual property in the knowledge graph. We derived a relevance score $r \in [0, 1]$ for each property p for the given class c: $r_{p,c} = N_p/N_c$, where N_p is the number of distinct occurrences of the property and N_c is the number of distinct occurrences of the class.

For a given entity, 1) we extracted all its classes and 2) available properties and 3) found the top properties for each class and verified if they existed for the given entity. If they did, we shortlisted those properties and displayed the top n properties to the user which were ranked by their relevance score.

Follow-Up Interactions. To enhance the user experience whenever the user makes a query to the system, helpful follow up interactions are provided in the form of buttons or links which the user can interact with to continue the conversation with the chatbot through a tap or click. Such a feature allows the user to delve deeper into a particular topic or result. For example, if the user searches Barack

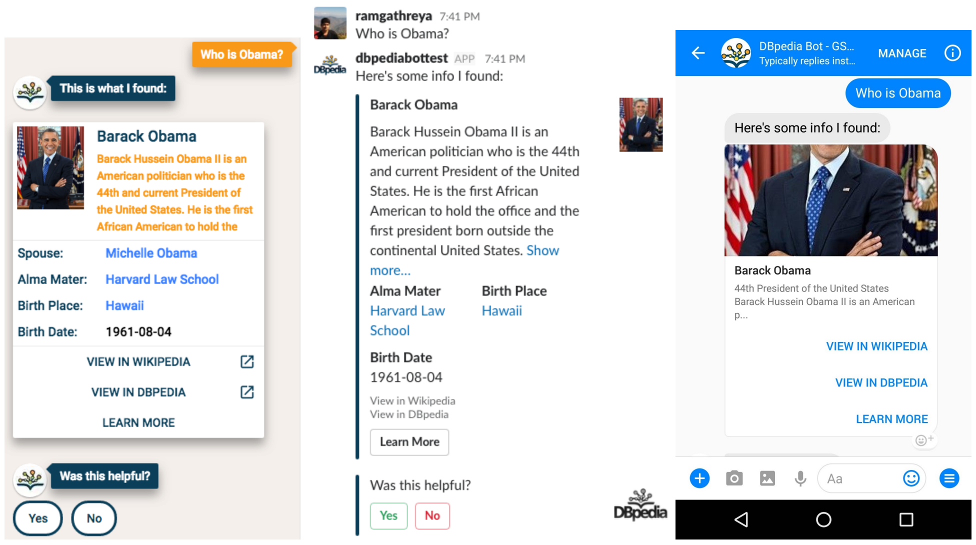
[11]Based on the DBpedia version 2016-10

[12]http://products.wolframalpha.com/api/

[13]https://www.w3.org/TR/rdf11-primer/

[14]https://www.openstreetmap.org/about

[15]In our case it is http://dbpedia.org/ontology/


Who is Obama?

This is what I found:

Barack Obama

Barack Hussein Obama II is an American politician who is the 44th and current President of the United States. He is the first African American to hold the

Spouse:	Michelle Obama
Alma Mater:	Harvard Law School
Birth Place:	Hawaii
Birth Date:	1961-08-04

VIEW IN WIKIPEDIA

VIEW IN DBPEDIA

LEARN MORE

Was this helpful?

Yes No

ramgathreya 7:41 PM

Who is Obama?

dbpediabottest APP 7:41 PM

Here's some info I found:

Barack Obama

Barack Hussein Obama II is an American politician who is the 44th and current President of the United States. He is the first African American to hold the office and the first president born outside the continental United States. Show more...

Alma Mater: Harvard Law School

Birth Place: Hawaii

Birth Date: 1961-08-04

View in Wikipedia

View in DBpedia

Learn More

Was this helpful?

Yes No

DBpedia Bot - GS... Typically replies inst... MANAGE

Who is Obama

Here's some info I found:

Barack Obama

44th President of the United States

Barack Hussein Obama II is an American p...

VIEW IN WIKIPEDIA

VIEW IN DBPEDIA

LEARN MORE

Aa

Figure 3: Screenshot of three different platform views (left to right): Web, Slack, Facebook Messenger.

Obama, the DBPEDIA CHATBOT can also show related entities such as the Democratic Party or similar people such as Bush or Clinton.

Banter or Casual Conversation. Banter messages are typically casual in nature and have very little bearing on the core functionality of a chatbot. Still, we provide a basic set of responses for such queries. Some examples of banter could be messages such as *"Hi"*, *"What is your name"*, etc. As a fallback in cases of no applying rule, Eliza [11] is used as a filler.

Deployment to Platforms. Our approach is accessible from different platforms to increase the usability and allows as many users as possible to access it. In particular, the DBPEDIA CHATBOT can be accessed via a web interface (optimized for both Desktop & Mobile), Slack and Facebook Messenger (which we hope to integrate soon into DBpedia's official page). Screenshots of all platforms can be seen in Figure 3.

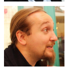
4 CONCLUSION

Since its launch (between July 2017 to Dec 2017), the DBPEDIA CHATBOT has been used by over 1400 users with an average conversation length of 16 messages. These statistics suggest that there is a genuine interest for users to interact with the DBPEDIA CHATBOT. We hence aim to extend DBPEDIA CHATBOT in the future to overcome the current limitations, e.g. we will include other approaches such as the DBpedia relationship finder[16], more data sources (stackoverflow) and evaluate feedback on such interactions, e.g., we will investigate details on types of questions asked by users compared with the top ones in DBpedia mailing lists.

Acknowledgements This work has been supported by the BMVI projects LIMBO (project no. 19F2029C) and OPAL (project no. 19F2028A) as well as by the German Federal Ministry of Education and Research (BMBF) within 'KMU-innovativ: Forschung für die zivile Sicherheit' in particular 'Forschung für die zivile Sicherheit' and the project SOLIDE (no. 13N14456).

Authors: Ram G Athreya is a graduate student at Arizona State University and Google Summer of Code '17 participant.

Prof. Axel-Cyrille Ngonga Ngomo has been a Google Summer of Code mentor since 2015.


Dr. Ricardo Usbeck has been a Google Summer of Code mentor since 2016.

REFERENCES

[1] Sameera A Abdul-Kader and John Woods. 2015. Survey on chatbot design techniques in speech conversation systems. *Int. J. Adv. Comput. Sci. Appl.* (2015).

[2] H. Al-Zubaide and A. Issa. 2011. OntBot: Ontology based chatbot. In *International Symposium on Innovations in Information and Communications Technology*.

[3] A. Augello, G. Pilato, G. Vassallo, and S. Gaglio. 2009. A Semantic Layer on Semi-Structured Data Sources for Intuitive Chatbots. In *2009 International Conference on Complex, Intelligent and Software Intensive Systems*. 760–765.

[4] Andreas Both, Dennis Diefenbach, Kuldeep Singh, Saeedeh Shekarpour, Didier Cherix, and Christoph Lange. 2016. Qanary - A Methodology for Vocabulary-Driven Open Question Answering Systems. In *ESWC*. 625–641.

[5] Timofey Ermilov, Diego Moussallem, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. 2017. GENESIS: a generic RDF data access interface. In *Proceedings of the International Conference on Web Intelligence*. 125–131.

[6] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[7] Michael McTear, Zoraida Callejas, and David Griol. 2016. Conversational Interfaces: Past and Present. In *The Conversational Interface*. Springer, 51–72.

[8] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia spotlight: shedding light on the web of documents. In *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS*. 1–8.

[9] Mon-Tin Tzou, Chun-Hung Lu, Chin-Chien Wang, Cheng-Wei Lee, and Wen-Lian Hsu. [n. d.]. Extending knowledge of AIML by using RDF. ([n. d.]).

[10] Richard S Wallace. 2009. The anatomy of ALICE. *Parsing the Turing Test* (2009), 181–210.

[11] Joseph Weizenbaum. 1966. ELIZA—a computer program for the study of natural language communication between man and machine. *Commun. ACM* (1966).

[16]http://www.visualdataweb.org/relfinder.php