An Approach for Ex-Post-Facto Analysis of Knowledge Graph-Driven Chatbots – the DBpedia Chatbot

Rricha Jalota[1][0000−0003−1517−6394], Priyansh Trivedi[2], Gaurav Maheshwari[2], Axel-Cyrille Ngonga Ngomo[1][0000−0001−7112−3516], Ricardo Usbeck[1,2][0000−0002−0191−7211]

[1] Data Science Group, University of Paderborn, Germany
[2] Fraunhofer IAIS, Standort Dresden, Germany

Abstract. As chatbots are gaining popularity for simplifying access to information and community interaction, it is essential to examine whether these agents are serving their intended purpose and catering to the needs of their users. Therefore, we present an approach to perform an ex-post-facto analysis over the logs of knowledge base-driven dialogue systems. Using the DBpedia Chatbot as our case study, we inspect three aspects of the interactions, (i) user queries and feedback, (ii) the bot's response to these queries, and (iii) the overall flow of the conversations. We discuss key implications based on our findings. All the source code used for the analysis can be found at `https://github.com/dice-group/DBpedia-Chatlog-Analysis`.

Keywords: Empirical Study · Knowledge-driven Chatbot · Intent Clustering · Knowledge Graph · Conversational AI

1 Introduction

Recent years have seen a resurgence [10] of chatbots. The solutions are now used by a large number of businesses, in the entertainment industry, and for curiosity-driven purposes. While some dialogue agents imitate customer-service behavior to carry out certain tasks (e.g., Siri, Alexa, pizza delivery/hotel reservation chatbots), others act as an interface to explore underlying knowledge bases or other structural databases. These latter kinds of chatbots provide unified, malleable access to information, potentially collected from a wide variety of heterogeneous data. Recently, these data-driven agents have attracted significant research interest leading to considerable enhancement in their capabilities [4]. However, only a few studies have investigated how the existing systems perform and have leveraged their findings. Therefore, in this study, we present a generalizable approach to examine how users interact with knowledge-driven chatbots and their expectations with these agents.

Here, we intend to analyze these conversations for understanding **(i) how users interact with knowledge-driven chatbots, (ii) whether the chatbots can sufficiently satisfy the expectations of the users, and (iii) the**

possible avenues for improving chatbot quality and subsequently the user experience. To that end, we suggest three general analytical streams for investigating knowledge-driven chatbots. We run our analysis on a completely anonymized version of log files, broadly categorized in the following three classes:

- **Request Analysis:** We measure the intents, and complexity within user utterances to understand their perception towards the chatbot.
- **Response Analysis:** We characterize the common errors made by the chatbot as well as the reasons behind it.
- **Conversation Analysis:** We attempt to uncover common topics of conversation and inspect the use of anaphora in the conversations.

In particular, we investigate log files from the DBpedia Chatbot [4]. The log files provide domain-specific (here DBpedia-centered) information to enhance community interaction/engagement and answers factual questions on any topic using the DBpedia Knowledge Graph. Thus, this chatbot acts both as an agent that renders frequently asked questions (FAQs) and as an interface for knowledge-driven question answering, making it a unique case study. The DBpedia Chatbot has been running for around 26 months at the time of writing. During this period, it has been involved in over 9084 conversations. The findings from the ex-post-facto analysis of the DBpedia Chatbot suggest that while users do ask complex queries, they engage more in banter and simple questions. They also tend to use colloquial language, make spelling errors and feel reluctant to use appropriate casing or notation for proper nouns and abbreviations. This indicates that they anticipate intrinsic human-level comprehension from a machine, which in turn denotes the need for better natural language processing (NLP) tools or providing more intuition about the limiting cases of the chatbot.

We believe the analysis and findings from this case study will benefit all those genres of conversational interfaces that either engage in customer-service [18] or empower data-driven applications [15].

Our contributions in this paper are two-fold: 1) we propose retrospective data-driven approaches to inspect the performance and usage patterns of a knowledge-driven chatbot, and 2) based on the findings, we suggest solutions to improve DBpedia Chatbot's architecture and user-experience. The source code for our analysis of the DBpedia Chatbot[3] can be found online[4] along with the source code of the chatbot[5] itself. To ensure full compliance with the General Data Protection Regulation, we do not share or publish the dataset.

2 Related Work

In recent times, several new frameworks and approaches have been proposed to evaluate the usability of dialogue systems and the motivation behind their

[3] `http://chat.dbpedia.org`
[4] `https://github.com/dice-group/DBpedia-Chatlog-Analysis`
[5] `https://github.com/dbpedia/chatbot`

use. The evaluation of these systems can be carried out in experimental environments where a set of people examine a chatbot's performance for a certain time period and report their experience [14]. Or, they can be done in a completely natural setup where the chatbot interacts with the target audience and eventually, the log files, collected over a period of time, are studied [12,23]. Employing the natural evaluation setup, chatbot log files from a task-oriented chatbot of a telecommunications company were examined [2] to detect whether only conversations were sufficient to determine users' topics of interests and their level of satisfaction. For this purpose, conversations were characterized as sequences of events for network modeling and thereafter, network analysis techniques were applied. Conversation analysis techniques were also applied to a banking chatbot's log files [17]. In particular, the use of the intercoder reliability metric led to the discovery of multiple patterns of conversation breakdown. Various other experiments have also been conducted in experimental scenario-based settings to compare repair strategy [3] and user preferences in cases of unsuccessful interactions, and to improve the design of chatbots based on experiences of first-time users [13]. In contrast to the works mentioned above, which primarily examined domain-specific service-oriented chatbots, we carry out a global performance evaluation of a hybrid chatbot that is not only capable of delivering domain-specific information or engaging in banter but also acts as an interface to explore a large knowledge graph.

We perform a completely data-driven analysis and suggest three possible analytical steps for Knowledge Graph-driven Chatbots (see Section 5) to inspect various nuances of the usage-patterns and user-satisfaction of a goal-oriented chatbot.

3 Description of the DBpedia Chatbot

The open-source, open-access knowledge graph (KG) DBpedia [16], and its official community mailing list served as the heterogeneous data source for building the DBpedia-chatbot [4]. Developed for the purpose of enhancing community interactions in the DBpedia community, the chatbot is capable of interactively introducing users to the DBpedia KG, providing them with an intuitive interface to 1) query existing knowledge by the means of an underlying Knowledge Graph Question Answering (KGQA) [6] system, 2) answering queries regarding DBpedia service checks like "*Is lookup online?*", and 3) information related to specific user chapters. The DBpedia Chatbot, hence, follows the pattern of task-oriented chatbots [14,2].

Upon receiving user input, the chatbot classifies the intent of input as either factual questions answerable via underlying KG, questions related to DBpedia community, or banter. The factual questions are answered using a combination of tools based on the **q**uestion **an**swering vocabul**ary** - QANARY [5], and WolframAlpha,[6] while a rule-based system with pre-defined responses handles the

[6] `http://products.wolframalpha.com/api/`

DBpedia questions and the banter. The underlying system is trained by leveraging the existing FAQs, the DBpedia Discussion,[7] and DBpedia Developers mailing lists.[8] We refer interested readers to the DBpedia Blog[9] and the accompanying paper [4] on the abilities and the working mechanism of the chatbot.

4 Dataset

In this section, we introduce various statistical aspects of the chatbot's log dataset, which we subsequently analyze. The dataset is based on the anonymized[10] logs of the DBpedia Chatbot collected over two years and as of July 2019, contains 90,800 interactions. The chatbot was used by a total of 9084 users, with the most common channel being web (with 9078 users), followed by Slack and then Facebook Messenger. Every conversation in the dataset begins with the chatbot introducing itself and its capabilities to the user. The user then queries the chatbot, to which the chatbot responds with an answer and a feedback request. If the user submits feedback, the chatbot reacts to it accordingly. Otherwise, another dialogue begins if the user asks a new query. We present some preliminary characteristics of the dataset in Table 1, including:

- *Conversation length:* Number of interactions between a user and the chatbot.
- *User-request length:* The number of characters in a user utterance.
- *Feedback-asked:* The number of times the chatbot asks a user for feedback.
- *Feedback-received:* Amount of user responses to chatbot feedback requests.
- *Feedback-content:* The number of received positive and negative feedback.

Table 1: DBpedia Chatbot log data from Aug. 2017 to July 2019.

Characteristic	**Measure**	
Number of user-utterances	Absolute	30263
Conversation length	Avg.	10
	Max.	1661
User-request length	Avg.	113
	Max.	3389
Feedback	Feedback-asked	28953
	Feedback-received	7561
	Positive-feedback	3406
	Negative-feedback	4155

[7] `https://sourceforge.net/p/dbpedia/mailman/dbpedia-discussion/`
[8] `https://sourceforge.net/p/dbpedia/mailman/dbpedia-developers/`
[9] `https://wiki.dbpedia.org/blog/meet-dbpedia-chatbot`
[10] No unique identifiers or demographics were collected by the DBpedia Chatbot.

5 Approach

In this section, we describe our proposed approach for analyzing various aspects of domain-specific, knowledge-driven chatbots. The goal of these analyses is (i) to understand the nature of user-requests: query-patterns and user-intentions; (ii) examine whether the chatbot can serve its purpose and satisfy user-requests; and (iii) get insights about the conversation flow to improve the chatbot's architecture. We divide our analysis, based on different aspects of the conversation, into three major categories, namely:

- **Request Analysis**: where we analyze a user's request based on the intent and complexity of the utterance. We propose to use either dependency parsing-based techniques or sentence embeddings to capture a query's intent and thereafter, applying an unsupervised clustering method for classifying the intent of the utterances. Furthermore, we present a rule-based dependency parsing approach for determining an utterance's complexity.
- **Response Analysis**: where we intend to identify common errors made by chatbot by analyzing common entity types in failed responses and length of conversations. We employ two different **n**ame **e**ntity **r**ecognition (NER) systems, namely `spaCy` and DBpedia Spotlight for identifying entity types.
- **Conversation Analysis**: where we identify common user topics. We propose to use DBpedia Spotlight to identify major conversation themes.

5.1 Request Analysis

In this section, we examine the manner in which users perceive and interact with a knowledge-driven chatbot, in particular the DBpedia Chatbot, to determine the important avenues of improvement. This is accomplished by identifying the **intent**, **complexity**, and **kind** (factual/non-factual) of user utterances.

Intent Analysis: The intent of a user utterance broadly refers to the desired outcome of a particular interaction with a chatbot. Our motivation is to check the coverage of a knowledge-driven chatbot. Due to the lack of ground truth data pertaining to queries' intent, we perform this experiment in an open-ended setting. This means, instead of classifying utterances into fixed classes, we use unsupervised clustering algorithms[11] to group user utterances and treat them as latent intents.

For better generalization, we first detect entity mentions (using DBpedia Spotlight [9]) and replace them with their corresponding schema type. For instance, a query, "*Who is Emmanuel Macron?*", is first normalized to "*Who is* `Person`*?*" and then undergoes the transformation required for clustering.

That is, due to the variety of queries, we rely on two approaches: 1) extracting the verb-object pairs and vectorizing them through an embedding matrix, 2) utilizing sentence embeddings that capture the semantics of the entire utterance.

[11] Inspired by `https://building.lang.ai/sorry-i-didnt-get-that-how-to-understand-what-your-users-want-a90c7ca18a8f`

1) For extracting the candidate pair, i.e., the verb-object pair in the query, we employ the state-of-the-art Stanford's dependency parser [22]. User queries can lack either a verb or an object or both. If the `verb-object` pair is not found, we look for one of the following pairs: `noun-nmod, noun-nsubj/ pron-nsubj, propn-nmod, nmod-case or noun-cop` in the order mentioned. This order was determined via a preliminary analysis over the dataset. Table 2 shows candidate pairs generated via dependency parsing on a few queries. These pairs are vectorized using fastText [21] subword vectors.

Table 2: Candidate pairs from dependency parsing

Query	**Dependency Relation**	**Candidate Pair**
give the German history	verb-object	(give, history)
What is the plot of Titanic?	noun-nmod	(Titanic, plot)
What is a computer?	pron-nsubj	(What, computer)
what about your breakfast	nmod-case	(breakfast, about)
Who is president of Country	propn-nmod	(president, Country)
now we want to be your friend	noun-cop	(friend, be)

2) Taking into account that the dataset is replete with malformed queries, there are many utterances that do not fit the dependency relations described above. Thus, simply relying on candidate pairs is not sufficient for intent classification. In this regard, to capture a deeper insight, we employ Multilingual Universal Sentence Encoder [26] to encode user utterances and subsequently cluster the vectorized utterances based on their semantic similarity.

To cluster the vectors obtained via the two methods described above, we 1) reduce their dimensions (to 50) using Principal Component Analysis (PCA)[12] and 2) standardize them to obtain normally distributed data. From our observations, without standardization, the clustering algorithms did not perform well and categorized most of the data as noise.

We 3) applied t-Distributed Stochastic Neighbor Embedding (t-SNE) [19] as a preprocessing step to enhance the performance of the 4) density-based clustering performed via the Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [20] algorithm.[13]

As for the clustering algorithm itself, we chose HDBSCAN primarily because we found it to be faster on our dataset and superior at clustering data of varying densities. We also found that unlike other clustering algorithms that assume considerable domain knowledge, HDBSCAN has more intuitive parameters to make data-driven decisions. Figures 1 and 2 depict the results from both approaches.

[12] Applied as a preprocessing step for t-SNE algorithm, see `https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html`

[13] We refer the interested readers to also check `https://stats.stackexchange.com/questions/263539/clustering-on-the-output-of-t-sne`

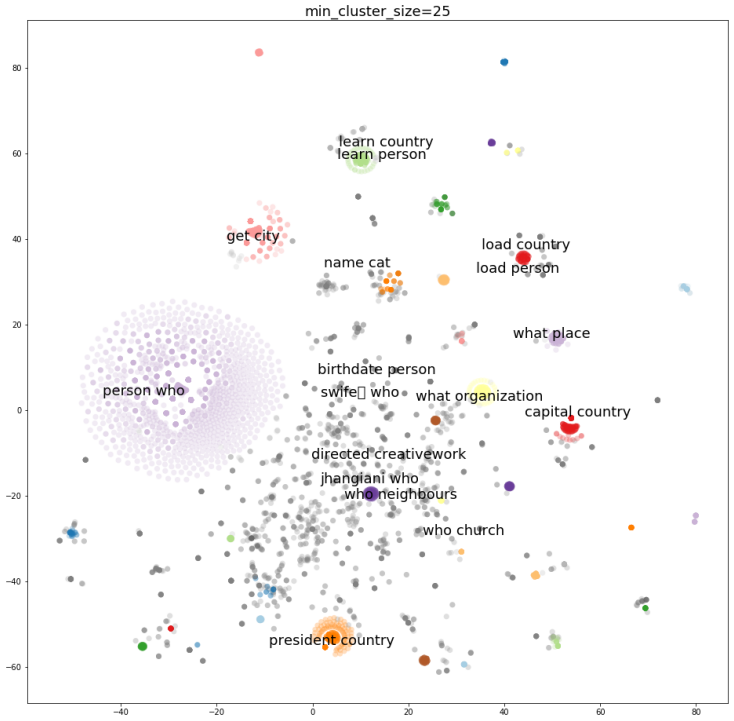

Fig. 1: Visualization of clusters obtained via HDBSCAN on the selected candidate-pair vector embeddings. Each cluster consists of at least 25 samples. The top 10 clusters out of a total of 35 have been labeled with their top terms.

Ex-post-facto research designs can be used if no requirements or experimental investigation strategies exist and noisy variables cannot be controlled. Thus, ex-post-facto designs only allow correlative statements on vast amounts of data. These vast amounts of data can be collected with little financial and personal effort using chatbot logs. By classifying utterances into well-defined categories, one could perform such a correlative analysis. However, this usually requires annotated data for the classification algorithm to generalize well.

Hence, we propose the above-described enhanced mechanism to automate the clustering of utterances based on semantic similarity and actionable word pairs, in the absence of labeled data. These clusters will guide our future research agenda to satisfy user information needs.

Complexity of utterances: With the increasing research interest in developing solutions [25,1] for complex question answering over knowledge-graphs, it is crucial to investigate the number of such questions that are actually asked by the users in real-world settings to such knowledge-graph driven interfaces. Since factual questions constitute a substantial part of interactions with knowl-

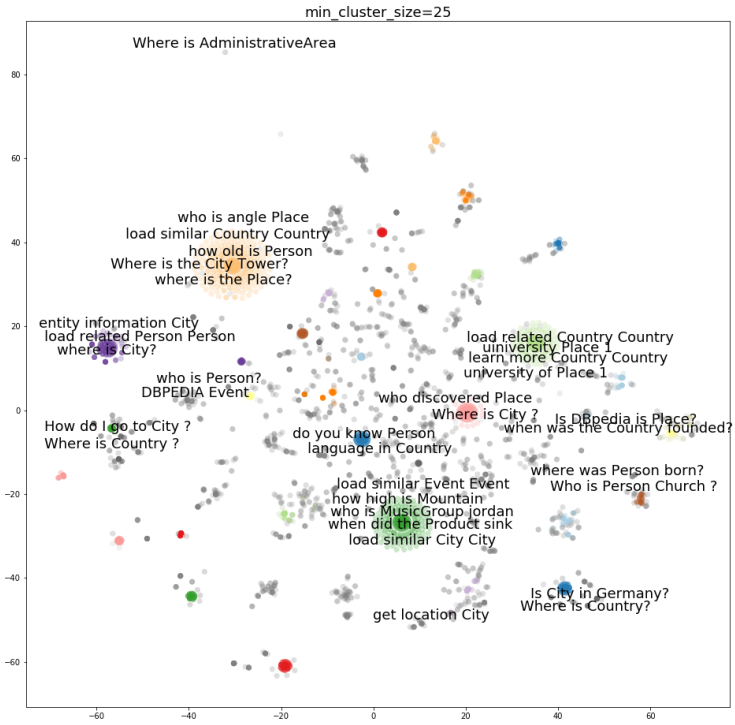

Fig. 2: Visualization of clusters obtained via HDBSCAN on sentence embeddings. Each cluster consists of at least 25 samples. The top 10 clusters out of a total of 33 have been labeled with their top terms.

edge graph-driven chatbots, we perform an experiment to better understand the nature of these questions.

This experiment is primarily based on a distinction of question complexity. A question is deemed to be complex if it contains a relative clause that modifies the noun or pronoun in it. An example of such an utterances is "*Can you give me the names of women born in the Country during the 19th century?*". Contrarily, simple questions are devoid of any modifiers and follow a simple sentence structure, such as "*Who is Jimmy Wales?*", "*When was Donald Trump born?*".

This distinction closely follows the distinction in KGQA, where a question is defined to be simple if it is answerable via a single triple pattern. Existing literature in the field of KGQA [6] consists of different approaches specific to the aforementioned distinction of question complexity. Thus, estimating the distribution of simple and complex questions can guide the KGQA development.

To determine the complexity of a query, we examine its dependency parse tree. First, we look for a candidate relation pair and then check whether any of its child nodes (relation denoted by `(token-head)->child`) exhibit a clausal or nominal modifier. We then search for the following dependency relations:

`(obl-verb)-> amod or nmod or nummod`, `(nsubj-verb)->acl` and `(obj-verb)-> amod or mod or nummod` to estimate the occurrence of complex utterances.

Using this approach, we estimate that only 3.3% of utterances given to DBpedia Chatbot were classified as complex questions, based on a sample set of 5,000 utterances.

Miscellaneous Analysis : Our goal through this analysis is to examine whether the users conform to the limitations of a chatbot even after being informed in advance. It is plausible that users perceive the capabilities of a chatbot to be analogous to Google Assistant, Apple Siri or Amazon Alexa.

Despite the fact that the DBpedia Chatbot is a domain specific dialog system that introduces its capabilities prior to the first user-utterance in every conversation, just like the other task-oriented chatbots [14,2], we notice several banter utterances from the user. To estimate the frequency of banter utterances, we manually inspect 2000 utterances, randomly sampled from the dataset. Through this, we find that about 12.6% are domain-agnostic, non-greeting utterances which we label as banter such as "united states president".

Moreover, an inspection of the language of utterances[14] suggests that users attempted to query the chatbot in their native languages; Russian, Arabic, German, Korean, Portuguese - naming a few, notwithstanding the language limitations of the chatbot.

5.2 Response Analysis

In the response analysis, we attempt to characterize common errors made by the knowledge-driven chatbots by investigating the requests corresponding to the responses that received negative feedback. We also attempt to discover the reason behind the chatbot's (in particular, the DBpedia Chatbot's) inability to answer those queries, using the following metrics:

Number of Failed Responses per conversations: Approximately 6.9% of the chatbot's responses were marked as incorrect through the feedback form (optionally presented along with every response from the chatbot), indicating factual inaccuracy or general dissatisfaction for that response. Note that this estimation is not representative of all the cases where users experienced dissatisfaction, since responding to the feedback is optional.

Sustaining Conversations after Negative Feedback: Here, we intend to estimate the effect of erroneous response on the conversation by computing the average number of messages exchanged after the first negative feedback. In the case of the DBpedia Chatbot, we find that this number is approx. 7 (std. dev = 16.54). Together, these numbers, when compared with the average conversation length across all conversations, which is approx. 10, suggests that users still interact with, and derive merit from the chatbot despite an erroneous response. This could be (partly) attributed to either the largely atomic nature of conversations (held by the DBpedia Chatbot) or to its ability to recover from failures using fallback mechanisms.

[14] Using the python library `langdetect` `https://pypi.org/project/langdetect/`

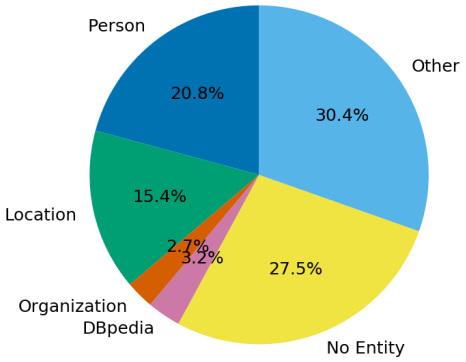

Fig. 3: Entity type distribution from 1000 manually annotated failed utterances.

Entity Types in Utterances prior to Negative Feedback: To gain an even deeper understanding of failing conversations, we manually annotate a subset of utterances with incorrect responses. We intend to characterize the utterances that caused a failure, to build targeted mechanisms to address these pitfalls. The distribution of manually annotated 1000 utterances has been reported in Fig. 3. A majority of 30.4% utterances, which have been marked as `Other`, consisted of entities like astronomical objects, movies, etc. (e.g. "*Is pluto a planet?*").

We then compare the accuracy of DBpedia Spotlight [9] and pre-trained `spaCy NER` in spotting the person and location mentions that were identified above. The results of this experiment have been reported in Table 3. We find that while DBpedia Spotlight performs better than `spaCy NER` in our context, there is a need for using more robust entity detection and linking mechanism on noisy data. In hindsight, we also need better dialogue modules for utterances with no or uncommon entity types.

In general, the failure cases in other chat logs can also be examined by (i) calculating the number of failed responses in every conversation, (ii) checking the length of conversations after a negative feedback and (iii) inspecting the utterances prior to the negative feedback for domain-specific vulnerabilities (entity-types in our case).

Table 3: `spaCy-NER` and DBpedia Spotlight accuracy for detecting person and location mentions.

System	**Person**	**Location**
`spaCy-NER`	41.3%	42.2%
DBpedia Spotlight [4]	69.2%	46.1%

5.3 Conversation Analysis

Finally, we aim to uncover the common user topics that users ask a knowledge-driven chatbot and the use of anaphora in their conversations. This is to understand the potential improvements in the chatbot's architecture and the NLP tools used in the pipeline from the perspective of both knowledge-graph question answering and human-computer interaction.

For **extracting the topics**, we use DBpedia Spotlight [9] which provides us the underlying schema type for named entities. The schema-entity pair is obtained for every user-request. This enables us to measure the commonly-asked topics as indicated by the density of the schemas.

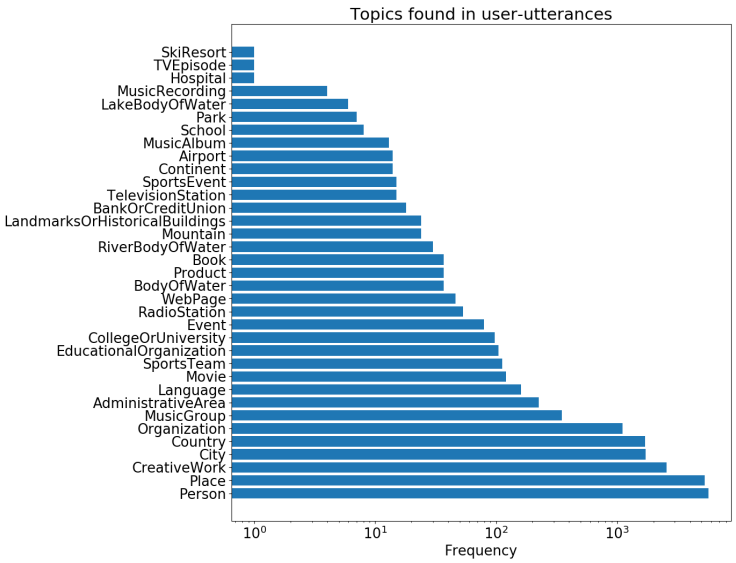

Fig. 4: Topics as identified by DBpedia Spotlight

Fig. 4 suggests that a majority of user-queries were primarily concerned with `Person`, `Place` and `DBpedia` (categorized under `CreativeWork`). Referring to the same entity in a text is a commonly occurring linguistic phenomenon, typically referred to as **anaphora**. Detecting and resolving anaphora (or coreference resolution) is a crucial part of conversational agents, which requires keeping track of conversational context over time. We use the python library, `NeuralCoref`[15], which implements the state-of-the-art coreference resolution approaches [7,8], to estimate the frequency with which anaphora occurs in the data. We find that the library detects only 45 such instances out of 9084 conversations. We attribute this infrequent occurrence of the phenomenon to the nature of the DBpedia Chatbot

[15] `https://github.com/huggingface/neuralcoref`

- of answering factual questions, and DBpedia service inquiries etc., which do not require multi-turn conversations. In contrast, a pizza delivery chatbot that collects required information through multiple rounds of disquisition with the user is more likely to see anaphora in user utterances more frequently.

In general, by fetching the schema to which an entity in utterance belongs, one can identify the topic of the utterances, which can be further used to enhance the backend engine of a chatbot. Additionally, to enhance the human-computer interface of a chatbot, one can inspect the log files for coreferences in queries.

6 Discussion

Based on our ex-post-facto analysis approach, which we applied to the DBpedia Chatbot, we summarize the key implications for future design of the DBpedia Chatbot:

- **Adding support for multilingualism** In our analysis, we found several instances of user utterances in languages other than English, even though the interface clearly states English as the medium of conversation.
- **Smart Suggestions** Upon manual inspection of a user query subset, we found several spelling errors, capitalization/casing errors, and other grammatical errors. To mitigate this, we suggest the use of auto-completion.
- **Detecting implicit feedback and out-of-scope queries** As discussed in Section 5, the user perception of the chatbot often leads to out-of-scope questions, and sometimes, implicit feedback provided not via the feedback button but through utterances. It is imperative, thus, to be able to detect and subsequently handle these utterances explicitly. One promising approach for this is to utilize the automatic clustering discussed in Section 5.1
- **Knowledge-based QA** Most of the user utterances related to KGQA were simple, i.e., questions that are answerable by a single triple. Even though the underlying KGQA system reported very high performance in the simple QA setting [11], we found that the system often failed. The low performance suggests a need for a more robust question answering system. We also found that users expect the KGQA system to act as a search engine and thus the underlying question answering system also needs to support keyword queries.

Consequently, we can derive implications for general knowledge-driven conversational interfaces:

- **Multilingual Support:** (Knowledge-driven) chatbots must support multiple languages from the start, which could be accomplished by either using translation services or by using multilingual NLP tools in the pipeline.
- **Guide User Input:** Chatbots must account for imperfect user input by directing the user towards typing grammatically correct queries using auto-correcting, auto-completion or controlled natural language interfaces.

- **Guiding User Expectations:** Users can mistake simple conversational interfaces or KGQA systems for powerful general AI systems and end up in the uncanny valley. Thus, managing user expectations by detecting and reacting to out-of-scope user intents must be at the core of a chatbot [24].
- **Adding explainability** Finally, we believe that extending the proposition of Ashktorab et al. [3] of adding explainability as a repair strategy to mitigate conversation breakdowns, will keep the users engaged and reduce the amount of negative feedback resulting from the frustration of unsuccessful queries.

7 Conclusion

In this work, we propose a threefold approach to conduct an ex-post-facto analysis on the user interaction logs of knowledge-driven chatbots. This analysis focuses on three broad perspectives, namely, (i) analysis of user utterances, (ii) analysis of user-requests that received negative feedback, and (iii) an overall analysis of conversations. Our goal, through this work, is to identify the avenues for potential improvement through a data-driven method.

We substantiate the value of the analysis with experiments over the log files of the DBpedia Chatbot and report multiple findings, see Sec. 6. Broadly, we conclude that in the case of relatively open-ended chatbots, unsupervised clustering through universal sentence embeddings can effectively cluster user-utterances based on their semantic similarity, thereby signaling their intents. Through manual analysis over a subset of the logs, we find that entity mentions cannot be reliably detected through off-the-shelf solutions, and require the development and application of robust entity detection and linking approaches.

In our opinion, our approach has merit outside the narrow domain of the DBpedia Chatbot and can be generalized to other knowledge-driven chatbots. The implications from our findings can be incorporated in future chatbots for better user-experience. However, it is worth noting that, to extend these findings and their implications, other query logs must be examined with the proposed ex-post-facto approach.

Acknowledgments. This work has been supported by the Federal Ministry of Transport and Digital Infrastructure (BMVI) in the OPAL research project (grant 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).

References

1. Abujabal, A., Roy, R.S., Yahya, M., Weikum, G.: Comqa: A community-sourced dataset for complex factoid question answering with paraphrase clusters (2019)
2. Akhtar, M., Neidhardt, J., Werthner, H.: The potential of chatbots: Analysis of chatbot conversations. In: CBI (2019)
3. Ashktorab, Z., Jain, M., Liao, Q.V., Weisz, J.D.: Resilient chatbots: Repair strategy preferences for conversational breakdowns. In: CHI (2019)

4. Athreya, R.G., Ngomo, A.N., Usbeck, R.: Enhancing community interactions with data-driven chatbots-the dbpedia chatbot (2018)
5. Both, A., Diefenbach, D., Singh, K., Shekarpour, S., Cherix, D., Lange, C.: Qanary - A methodology for vocabulary-driven open question answering systems (2016)
6. Chakraborty, N., Lukovnikov, D., Maheshwari, G., Trivedi, P., Lehmann, J., Fischer, A.: Introduction to neural network based approaches for question answering over knowledge graphs. CoRR **abs/1907.09361** (2019)
7. Clark, K., Manning, C.D.: Deep reinforcement learning for mention-ranking coreference models. In: EMNLP (2016)
8. Clark, K., Manning, C.D.: Improving coreference resolution by learning entity-level distributed representations (2016)
9. Daiber, J., Jakob, M., Hokamp, C., Mendes, P.N.: Improving efficiency and accuracy in multilingual entity extraction (2013)
10. Dale, R.: The return of the chatbots. Natural Language Engineering **22**(5), 811–817 (2016)
11. Diefenbach, D., Migliatti, P.H., Qawasmeh, O., Lully, V., Singh, K., Maret, P.: Qanswer: A question answering prototype bridging the gap between a considerable part of the LOD cloud and end-users (2019)
12. Huang, J., Zhou, M., Yang, D.: Extracting chatbot knowledge from online discussion forums. In: IJCAI. vol. 7, pp. 423–428 (2007)
13. Jain, M., Kumar, P., Kota, R., Patel, S.N.: Evaluating and informing the design of chatbots. In: DIS (2018)
14. Jenkins, M., Churchill, R., Cox, S.J., Smith, D.J.: Analysis of user interaction with service oriented chatbot systems (2007)
15. Keyner, S., Savenkov, V., Vakulenko, S.: Open data chatbot (2019)
16. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web (2015)
17. Li, C., Chen, K., Chang, Y.: When there is no progress with a task-oriented chatbot: A conversation analysis (2019)
18. Lommatzsch, A., Katins, J.: An information retrieval-based approach for building intuitive chatbots for large knowledge bases (2019)
19. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
20. McInnes, L., Healy, J., Astels, S.: hdbscan: Hierarchical density based clustering. J. Open Source Software **2**(11), 205 (2017)
21. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in pre-training distributed word representations. In: LREC (2018)
22. Qi, P., Dozat, T., Zhang, Y., Manning, C.D.: Universal dependency parsing from scratch (2018), `https://nlp.stanford.edu/pubs/qi2018universal.pdf`
23. Rivolli, A., Amaral, C., Guardão, L., de Sá, C.R., Soares, C.: Knowbots: Discovering relevant patterns in chatbot dialogues. In: ICDS (2019)
24. Skjuve, M., Haugstveit, I.M., Følstad, A., Brandtzaeg, P.B.: Help! Is my Chatbot Falling into the Uncanny Valley? An Empirical Study of User Experience in Human-Chatbot Interaction. Human Technology (2019)
25. Vakulenko, S., Garcia, J.D.F., Polleres, A., de Rijke, M., Cochez, M.: Message passing for complex question answering over knowledge graphs (2019)
26. Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Ábrego, G.H., Yuan, S., Tar, C., Sung, Y., Strope, B., Kurzweil, R.: Multilingual universal sentence encoder for semantic retrieval. CoRR **abs/1907.04307** (2019)