Discovering Context Information and Parameters from a Natural Language query

Kamath Sanjay

▶ **To cite this version:**

Kamath Sanjay. Discovering Context Information and Parameters from a Natural Language query. [Research Report] Université de Grenoble et INP Grenoble. 2015. hal-01171194

HAL Id: hal-01171194
https://hal.science/hal-01171194v1

Submitted on 3 Jul 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Discovering Context Information and Parameters from a Natural Language query

Sanjay Kamath Ramachandra Rao
M1 Master of Science in Informatics at Grenoble - TER Report
Grenoble INP/ UJF
Grenoble, France
Sanjay-Kamath.Ramachandra-Rao@imag.fr

Supervised by: Marie-Christine Fauvet, Professor, UJF (LIG/MRIM)

I understand what plagiarism entails and I declare that this report is my own, original work.
Name, date and signature:

Abstract

The *E-tourism Project* deals with variety of use cases of the system for a user who is a tourist and would like to perform various activities like asking for driving directions, booking a hotel room etc. The proposed method focuses on extracting the context information and parameters required for the composition of services from a Natural Language query and perform the corresponding operation(s) requested by the user. To process the query, the system uses various external knowledge bases and Natural Language Processing tools to understand the named entities and proper context of the query.

1 Introduction and Background

The study reported in this paper is part of a broader project: E-Tourism Project funded by European Commission, which aims to design and implement a software system which provides mobile users with services according to their needs. Figure 1 depictes the whole architecture of this system. The roles of each module and data flow are rapidly introduced below.

1. User interaction and query management module: aims at managing user connections and getting natural language queries submitted by users using their mobile device. A user's query and her identification are received in this module in the data flow (1). This module extracts from the query information necessary for the choice and composition of the service components. With data flow (2) it sends single topic queries to the Discovery module and with data flow (3) the users' goals to the Composition and execution system. This paper focuses only on this module.
2. Discovery module: given the user's query received in the data flow (2), her profile and context, this module is responsible for retrieving services among a repository of service descriptions, that once composed can potentially meet the user's needs expressed in her query. The retrieved services are then sent, in the data flow (4), to the next module.
3. Composition and execution system: Eventually this module is in charge of automatically compose and execute services returned by the discovery phase. The automated composition is meant to satisfy users' goals. This module results in a BPM model whose each task refers to a service operation by the data flow (5) (for details see [Lumpoon *et al.*, 2013]). During the execution of the model some interactions with the user might be necessary (see data flow (6)).

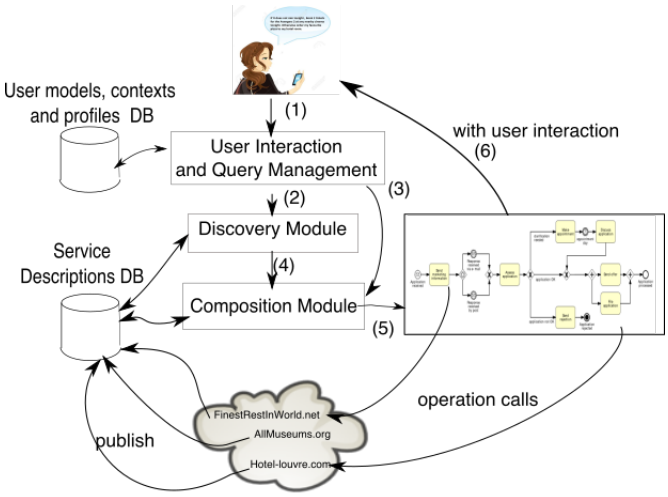

Figure 1: A context-aware discovering system for mobile users

We consider the following definitions for goal, context, profile and service. A *goal* refers to the task requested by the user. Examples: movie bookings, flat rentals, coffee shops nearby, events happening today, etc. A query might be about one or more goals. A *context* includes spatial, temporal, physical, and environmental properties that could be collected by sensors embedded on the devices used to submit the queries. Such properties are for example: GPS location, time stamp, external temperature, screen size, etc. A *profile* captures users' personal details, preferences and centres of interest (users' age, citizenship, gender, occupation, pre-

ferred recreational activities, etc). We adopt the definition of service given by the W3C[1]: *A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL[2]). Other systems interact with the web service in a manner prescribed by its description using SOAP[3] messages, typically conveyed using HTTP with an XML serialisation in conjunction with other Web-related standards.*

Existing technologies have solutions for various goals which a user might request from a system individually. A system which handles all the requests in the domain of tourism, cannot be seen in existence yet. Specifically speaking, it refers to an ideal system which takes input in the form of natural language and processes it to perform the tasks and also considers the user's profile and context information. The tasks in Tourism domain are of wide range and currently our aim is to focus on certain set of tasks which are commonly used by tourists.

We built a corpus of queries which consists of single goal oriented queries and complex goal oriented queries which involves multiple goals to be handled in a single query. We propose to structure this corpus as a set of cases, and for each case we show issues which are raised when trying to extract goals and parameters. The main contribution of our study is a proposal for a method to understand the query and generate topics it is about, and eventually its goals and parameters.

This paper is organized as follows: in the Section 2 we discuss about the research issues when processing natural language queries in order to understand them. While Section 3 discusses some related workThen, in Section 4 we discuss about proposed approaches to overcome these issues and in the Section 5 we discuss about the implementation of some of the approaches discussed below.

2 Research Issues

In this section we discuss issues raised by each case of queries according to the corpus we have built.

2.1 Named entity issues

Named Entity Recognition (NER), is a classic concern of Natural Language Processing (NLP) to determine the named entities in the text. There are many tools which can be used to determine the named entities in the text. Our focus is on the queries where the NER fails to recognize the inferred named entities.

Case 1: ***I would like to book a ticket for the grand budapest hotel nearby for tonight.***

Having no insight about the query, a human might interpret it as a hotel booking query in Budapest but, the real context of the query is a movie booking. The phrase "the grand budapest hotel" refers to a movie which was released in 2012. Having a keyword "ticket" and the phrase which is a movie name, refers to a movie ticket booking and has nothing do with a hotel in Budapest.

This problem exists when there are conflicting keywords in the same domain which refers to two different entities. like "Hotel" and "Movie" in our case.

Case 2: ***I would like to book a ticket for the grand budapest hotel in the city centre tonight.***

In the previous case, the query was inferred to be a movie booking query. In this case, presence of another keyword called "city centre", refers to a place in the city. Even though this query is the same as the previous one, existing systems which we discuss in upcoming sections, detect this query as a hotel booking query.

The issue here is that the relevance of certain keywords towards a certain task depends on the usage of appropriate keywords in the appropriate positions. As we see both queries have similar keywords except a few keywords which changes its goals.

2.2 Date and Time computations

Case 3: ***Book an apartment with 2 bedrooms on this New years eve for 2 days in Paris.***

The keywords present in the query infer that the user is looking forward to book an apartment for 2 days starting from 31st December, 2015.

Steps of shallow parsing for syntactic similarities between the goals will lead us to ask the user about the check-in date and check-out date. Though the query has the information about the check-in date as the new years eve that is 31st December, 2015, the program will not recognize the significance of these keywords in the query which signifies a popular date.

Ideally, the date 31st December should be inferred when the keyword *new year's eve* is used. The checkout date which is 2nd January shall later be calculated. Calculating this date requires certain services to be executed because the checkout time of the hotel depends on the hotel rules, some hotels may consider 12 AM to 12 AM as one day and some may consider other timings. So these details cannot be implicitly assumed, the service has to deliver the right information before we take decisions for its parameters.

Case 4: ***What are the events to look for in Paris on the Bastille Day this year?***

The service discovery module finds the right service for the query related to search of events for the day. But the service composition module awaits for the parameters required to execute the query. The parameter can be Bastille Day or sometimes it has to be explicitly mentioned that it is *14th of July*.

2.3 Location inference

Case 5: ***I want to book a restaurant table for 4 near by the Eiffel Tower.***

There are two different possibilities depending on the location of the user, if the user is currently in or nearby Las Vegas, Nevada, USA. When she enters the above query on her mobile device, she would be looking forward to book a table

[1] W3C glossary, `http://www.w3c.org/TR/ws-gloss`

[2] Web Service Description Language, `http://www.w3.org/TR/wsdl`.

[3] Former acronym for Simple Object Access Protocol, `http://msdn.microsoft.com/fr-fr/library/bb469912.aspx`

in a restaurant in the Las Vegas near by the Eiffel Tower and not the one in Paris, France.

In another case, if the user is in Paris, France, and enters the above query on her mobile device, she would be looking ahead to book a restaurant table near the Eiffel Tower in Paris and not the one in Las Vegas.

Dealing with these two interpretations for the same query above, the deciding factor is the users location and not just the query keywords. Hence deciding the location the query is about, cannot be just made using the shallow parsing and knowledge bases. Even the users location should also be considered, if available. Sometimes, it might be necessary to loop back to the user to ask her to release the ambiguity.

2.4 Complex queries

Case 6: ***I want to book a cheap flight from Lyon to Berlin anytime tomorrow and a bus from Grenoble to Lyon airport an hour before the departure***

This sort of query deals with two bookings, one is a flight booking for Berlin from Lyon followed by a bus booking from Grenoble to Lyon. Noticeable point here is the time, the user wants to reach Lyon one hour before his flight departs to Berlin.

The booking details for the flight are known only after the booking service finishes its execution, and then passed on to the service which, in turn, handles the booking for a bus from Grenoble to Lyon at the right time.

In the cases like this, the result from a service is passed as the input to another service, the execution of the program tends to be in some kind of loop which has to be managed and not sequential. This type of issues deal with the causality of events. The temporal relation between events are discussed in detail by [Allen, 1984].

3 Some Related Work

Research in the field of Natural Language Processing is extensive and active.

Having a look towards existing systems which performs similar tasks, we came across one of the software which is developed by Google, named Google Now[4]. This application runs on Android smartphones (Above Android 4.3) which are capable of connecting to internet and work as a personal assistant which can help users with many tasks and recommend what's good for them. Another similar system is the Siri[5] developed by Apple Inc. in 2011. *Siri is a spin-out from the SRI International Artificial Intelligence Center, and is an offshoot of the DARPA-funded CALO project.*[6]

The above pointed out systems lacks the processing on NL queries when the query contains complex requirements or sophisticated input like checking weather and then ordering a pizza.

Most of the tourism related work are using ontologies for knowledge sources. *Ontology is the philosophical study of the nature of being, becoming, existence, or reality, as well as the basic categories of being and their relations*[7]. Dealing with an implementation of a planning system for tourist, [Tomai *et al.*, 2005] discusses about the case study using an ontology. One of the main drawback of ontology-based approach is the cost for creating the ontology which needs having a domain expert around. However, the planning system considers the time taken to perform each task and decides the right set of tasks a user can perform in his available limited time, this insight was a good idea for planning the travel itinerary. Building an ontology is expensive and time consuming because of this [Faria *et al.*, 2014] and [Ruiz-Martınez *et al.*, 2011] propose a domain-independent process for Automatic Ontology Population which is a good approach to avoid the domain experts involvement in the process of creating an ontology.

Building ontology either using a domain expert or automatically, raises certain concerns about handling certain specific cases in tourism domain which are not commonly used. Certain use cases might be local to a region in some country and the frequency of the usage of this might be very less. Including this type of relations in the Ontology makes the ontology very large and if the relations are not included, the system fails to understand that use case.

One of research groups from Italy, worked on building a DBpedia[8] for tourism, [Cresci *et al.*, 2014] which involves building a database for tourist attractions. *DBpedia* is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. *DBpedia* allows you to ask sophisticated queries against Wikipedia, and to link the different data sets on the Web to Wikipedia data.

Our system can make use of this knowledge source for extracting named entities for the places in Europe, but this system is limited to some of the locations in Europe and not yet complete for all the places around the world.

There is a need for an approach which does not use ontology but performs the task similar to one using ontology and be smart enough to handle cases out of the domain knowledge.

4 Proposed Approach

The Figure 2 sketches the overall architecture of the User Interactions and Query Management module. The role of each sub module is discussed in detail below.

Dataflow(1) represents the interactions with the user and the system with the help of a GUI. Dataflow(2) is the query input from the UI to the Query management module which processes the query. Dataflow(3) is the individual queries after splitting. Dataflow(4) is the information from external sources and the user's profile. Dataflow(5) is the output of the system to the other modules of the project.

The Figure 3 represents a flowchart of the system which signifies the working of the Query Processing module.

[4] https://www.google.com/landing/now, last access:12/05/2015

[5] https://www.apple.com/ios/siri, last access:12/05/2015

[6] http://en.wikipedia.org/wiki/Siri

[7] http://en.wikipedia.org/wiki/Ontology

[8] http://wiki.dbpedia.org

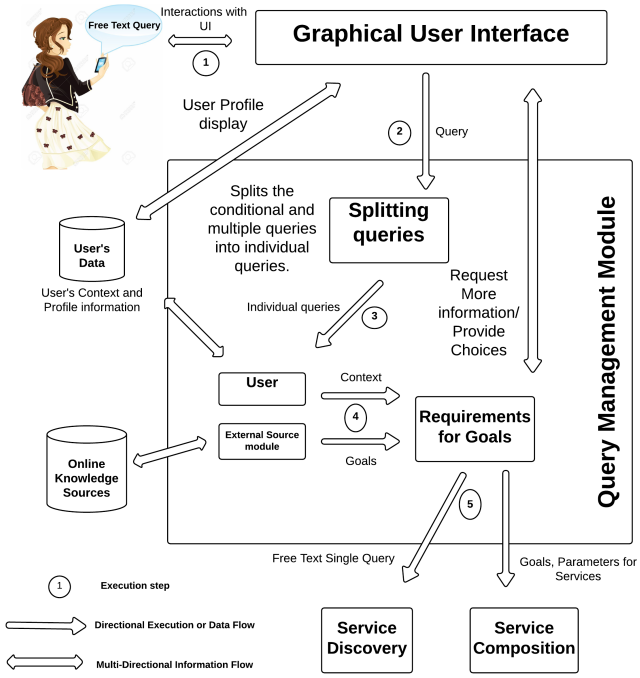

Figure 2: Query Processing Module architecture

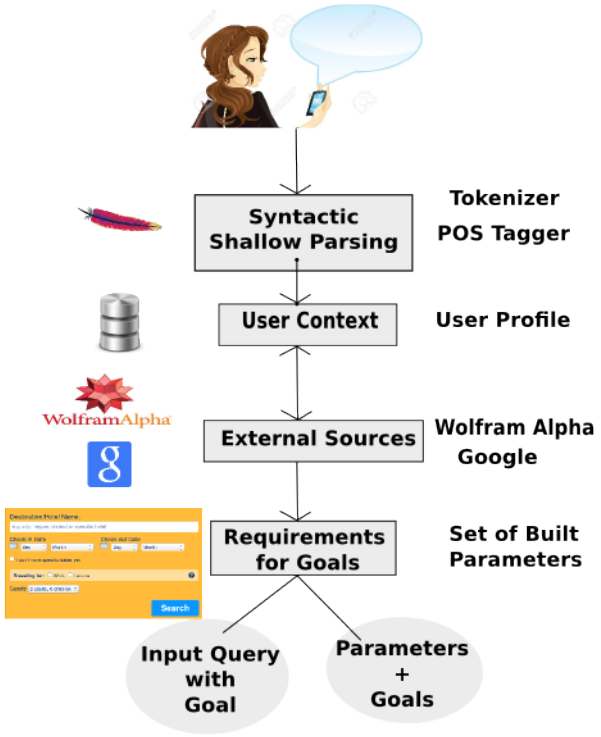

Figure 3: Execution flow

4.1 Query Processing

The user inputs the free query in a textbox and submits the query.

The system inputs the query to the program which deals with query management to check what is the user's goal in the query.

We define a set of sub modules to perform different tasks.

Splitting Queries

In certain scenarios, the queries might have conditions in them. The conditions in our corpus of queries are simple keywords like **if, then, else, otherwise, if not**. And in certain cases the queries might have multiple goals in them, these type of queries will have keywords like **and, along with, and also** etc.

Our proposed approach does the shallow parsing on the queries for searching these keywords and split the queries into separate smaller queries.

Shallow parsing refers to reading the input of the natural language query to match the Part of the Speech values for each keyword and the position of the keywords along with numerical elements in the query to extract the needed information without understanding the semantics of the query.

User Context

User's context refers to information which has to be retrieved from the User's device and User's profile. Queries which require location, date, time etc. shall access the device metadata to obtain these information. Location is of major concern for the privacy of the user.

Some queries might need the location of the user and some do not. This module analyzes the query and determines whether the location is necessary or not.

For other information like date, time, user name, nationality, choice of food, etc. will be accessible from the user's profile which can be used to filter the results.

External sources

The queries with named entities like famous places, or a festival which takes place every year, should have to undergo certain phases of processing to understand the correct named entities in them. There should be a knowledge base which is updated. We use various sources like Wikipedia, Google Applications etc. But we decided to use Wolfram Alpha as it is a computational engine which learns by a large corpus of data submitted by users.

Wolfram Alpha[9] is a computational knowledge engine or answer engine developed by Wolfram Research. It is an online service that answers factual queries directly by computing the answer from externally sourced "curated data", rather than providing a list of documents or web pages that might contain the answer as a search engine might.

We pass the query to Wolfram Alpha API to determine the goal of the query. However, the output provided by Wolfram Alpha is not totally correct for all the cases. To validate this, we take results from Google and compare it with Wolfram alpha.

[9] http://www.wolframalpha.com/

Ideally, if both the results match, we consider the results and proceed with it, else there can be mismatch in both the results.

For example, booking tickets for **The Tomatina festival**, Wolfram Alpha fails to recognize the keyword, **Tomatina** but Google understands the right goal.

In the cases like this, the system asks the user for the correct choice of the goal and then proceed with the further actions.

The result thus obtained signifies the main goal of the query.

Some of the responses from Wolfram Alpha are shown in the Figure 4 and Figure 5

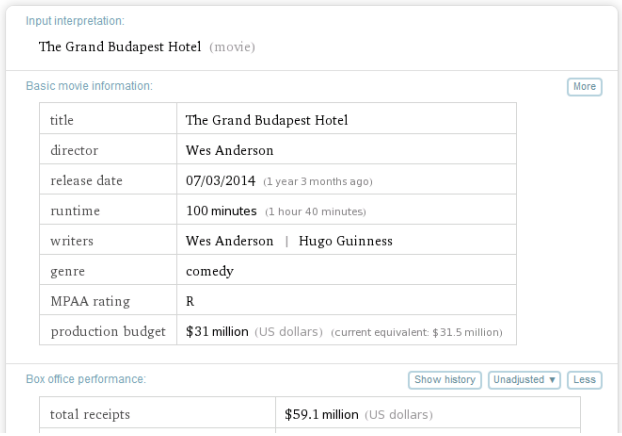

Figure 4: Wolfram Alpha's interpretation for Movie context.

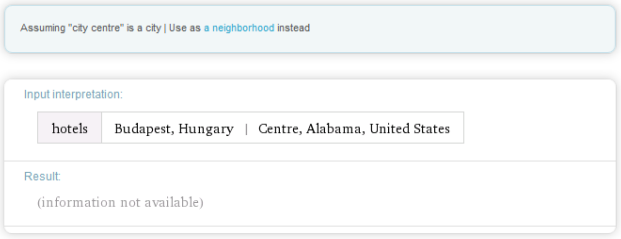

Figure 5: Wolfram Alpha's interpretation for Hotel context.

Requirements for goals

Each goal will have certain specific parameters. A movie booking has the movie name, number of people, location of the user as parameters. Instead of having ontology relations for each goal, we simply assume certain mandatory parameters for specific goals which are common in the domain of tourism.

Parameters which are assumed for common cases are minimal and don't conflict while interacting with the user.

The assumptions made by this method does not follow any relationships like in ontology, but this set of assumptions help in obtaining the parameters required for the execution of composition of services module.

The query will have certain parameters as shown in the Figure 6 which have to be directly substituted in the assumptions what our system makes for that particular goal, and the parameters like location can be obtained from the sensors on the device.

For example, if the query is about hotel booking, the source location of the user is not needed but if the user asks for driving directions to that hotel, the system needs the source location.

Query Objectives	Parameters
Movie Booking	Src Location, Dest Loc, Movie name, No. of people, Time
Hotel Booking	Src Location, Dest Loc, No. of people, Check In time, Check out time, Price range
Museum Booking	Src Location, Dest Loc, No. of people
Driving Directions	Src Location, Dest Loc
Taxi Booking	Src Location, Dest Loc, price range
Restaurant Booking	Src Location, Dest Loc, No. of people, Time
Search for a specific place	Src Location, Dest Loc, Place name
Search for eatery nearby	Src Location, Dest Loc, type of search
Search for popular things around	Src Location

Figure 6: Assumed parameters for certain objectives.

In cases where the system has conflicts in assuming the parameters, the interaction module asks the user to input that particular parameter again rather than making wrong assumptions.

We encounter Synonymy problem in this phase which involves one or more keywords having the same meaning. Words like 'Book' and 'Reserve' both refer to the same verb.

To tackle this problem, We use the Wordnet[10]. *Wordnet is a lexical database for the English language.It groups English words into sets of synonyms called synsets, provides short definitions and usage examples, and records a number of relations among these synonym sets or their members. WordNet can thus be seen as a combination of dictionary and thesaurus*[11].

The results from the query processing are of two types:

1. Original input query with the goal appended to the tail of the query.
2. Parameters from the query along with the Goal(s).

Appending the goal to the input query facilitates the Service discovery module to function without errors. If the goal of the query is not present in the query, Service discovery module fails to discover suitable services for the operation.

Parameters and Goal(s) are passed to the Composition module which performs the operations.

[10] https://wordnet.princeton.edu
[11] http://en.wikipedia.org/wiki/WordNet

4.2 Interactions - Select from the choices or provide more information

After the input query is processed by the system. Some of the queries whose output would be a set of suggestions would invoke an interaction with the user.

The user has to choose one of the choices to go ahead with booking , so the interaction system will provide a method to choose from different options provided by the services.

And in cases where the user query is missing a vital information like the checkout Date for the hotel booking, the specific field will be asked to the user by the service which handles it with the help of this module.

5 Implementation

The idea behind this project is to implement a system which helps the user to interact with the system with any mobile device which has internet connectivity or any computer device connected to internet. So we chose to implement the system as a web application which can be accessible from any device, and also create a mobile version of the same.

For processing the queries, we use Tokenizer and POS Tagger - Part of the Speech Tagger provided by Apache OPENNLP[12] for the purpose of programming the approach discussed above.

Tokenizer split the queries into tokens and the POS Tagger appends the tokens with the part of the speech corresponding to that word in the query.

This part of the speech value can be used to determine the goals, as goals are always a verb in our domain.

The online knowledge sources we use are Google and Wolfram Alpha. System parses the output from the sources to determine the goals.

After we perform the POS Tagging and Wolfram Alpha computation to understand the goal, we use Wordnet data to compare the results with the Goal.

We classify the requirements mentioned above using a key for representing each goal. This key is stored along with the set of requirements. For example, 'Book' is a key for all 'Bookings' type of queries. Whenever a **Verb** is encountered in the query, the Wordnet is accessed to get the Verbs which are synonyms to the input verb. The data obtained from Wordnet is matched with the keys to determine the key of the set of requirements which it matched.

We compare the Wordnet synonyms with the keys of the requirement class to understand the goal requested by the user when the user uses synonyms.

Once we obtain the goal, we substitute the data present in the query for the parameters we assumed.

Some of the parameters present in the query match the parameters assumed in the system, and some parameters might be missing. The system signifies it as missing parameter and passes it to the service composition module.

This module does not invoke an interaction with the user for the missing parameters, because the service operations might need few parameters and some of the parameters obtained from the user might not be used.

[12] `http://opennlp.apache.org`

6 Future Work

Further work can be focused on processing complex scenarios involving previous booking history. Recommendations can be tailored for the user using his profile information and history of searches. The subset of requirements can be made dynamic by computing the service requirements for each instance. Taking decisions on the results of the query is made blindly on the data of Wolfram Alpha and Google, this can made more accurate by considering the history and user's profile on a social media network.

More knowledge sources can be integrated easily in the system but work needs to be done on validating the results from these knowledge sources.

To improve the decision making, we can apply machine learning algorithms to classify the data which consists of user's choice, user's history, Wolfram alpha data and search results by Google and provide better learning for the system.

References

[Allen, 1984] J. Allen. Towards a general theory of action and time. *Artificial intelligence*, 23(2):123–154, 1984.

[Cresci *et al.*, 2014] S. Cresci, A. DErrico, D. Gazzé, A Lo Duca, A. Marchetti, and M. Tesconi. Towards a dbpedia of tourism: the case of tourpedia. In *Proceedings of the 2014 International Conference on Semantic Web-Poster and Demo Track, ISWC2014*, 2014.

[Faria *et al.*, 2014] C. Faria, I. Serra, and R. Girardi. A domain-independent process for automatic ontology population from text. *Science of Computer Programming*, 95:26–43, 2014.

[Lumpoon *et al.*, 2013] P. Lumpoon, M. Lei, I. Caicedo-Castro, MC Fauvet, and A. Lbath. Context-aware service discovering system for nomad users. In *7th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2013)*, 2013.

[Ruiz-Martınez *et al.*, 2011] J. Ruiz-Martınez, J. Minarro-Giménez, D. Castellanos-Nieves, F. Garcıa-Sánchez, and R Valencia-Garcia. Ontology population: an application for the e-tourism domain. *International Journal of Innovative Computing, Information and Control (IJICIC)*, 7(11):6115–6134, 2011.

[Tomai *et al.*, 2005] E. Tomai, M. Spanaki, P. Prastacos, and M. Kavouras. Ontology assisted decision making–a case study in trip planning for tourism. In *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops*, pages 1137–1146. Springer, 2005.