|  |
| --- |
| COM3001 Final Year Project |
| Developing a Chatbot to Answer Questions Based on Wikipedia Articles |
| University of Surrey |

|  |
| --- |
| Alex Turner  10 February 2020 |

# Declaration of Originality

I confirm that the submitted work is my own work. No element has been previously submitted for assessment, or where it has, it has been correctly referenced. I have also clearly identified and fully acknowledged all material that is entitled to be attributed to others (whether published or unpublished) using the referencing system set out in the programme handbook.

I understand that the University may submit my work as a means of checking this, to the plagiarism detection service Turnitin® UK. I confirm that I understand that assessed work that has been shown to have been plagiarised will be penalised.

If in completing this work I have been assisted with its presentation by another person, I will state their name and contact details of the assistant in the ‘Comments’ text box below. In addition, if requested, I agree to submit the draft material that was completed solely by me prior to its presentational improvement.

© Copyright Alex Turner, January 2020

# Acknowledgments

# Abstract

Abstract here

Contents

[Declaration of Originality 1](#_Toc32263577)

[Acknowledgments 3](#_Toc32263578)

[Abstract 4](#_Toc32263579)

[List of Figures 7](#_Toc32263580)

[List of Tables 8](#_Toc32263581)

[Glossary 9](#_Toc32263582)

[Abbreviations 10](#_Toc32263583)

[1 Introduction 11](#_Toc32263584)

[1.1 Overview 11](#_Toc32263585)

[1.2 Objectives 11](#_Toc32263586)

[1.3 Report Structure 11](#_Toc32263587)

[2 Literature Review 12](#_Toc32263588)

[2.1 Chatbots 12](#_Toc32263589)

[2.2 Natural Language Processing 12](#_Toc32263590)

[2.3 Machine Learning 12](#_Toc32263591)

[2.4 Datasets 12](#_Toc32263592)

[2.5 Programming Languages 14](#_Toc32263593)

[2.6 Existing Solutions 14](#_Toc32263594)

[2.7 Conclusion 14](#_Toc32263595)

[3 Problem Analysis 15](#_Toc32263596)

[4 System Requirements 16](#_Toc32263597)

[4.1 Requirements Gathering 16](#_Toc32263598)

[4.2 Software Requirements 16](#_Toc32263599)

[Functional Requirements 16](#_Toc32263600)

[Performance Requirements 17](#_Toc32263601)

[5 Design 18](#_Toc32263602)

[5.1 Methodology 18](#_Toc32263603)

[6 Implementation 19](#_Toc32263604)

[7 Testing 20](#_Toc32263605)

[8 Project Evaluation 21](#_Toc32263606)

[9 Conclusion 22](#_Toc32263607)

[9.1 Overview 22](#_Toc32263608)

[9.2 Conclusions 22](#_Toc32263609)

[9.3 Future Work 22](#_Toc32263610)

[9.4 Final Statement 22](#_Toc32263611)

[10 Ethics 23](#_Toc32263612)

[11 References 25](#_Toc32263613)

[12 Appendix 27](#_Toc32263614)

# List of Figures

[Figure 1: DBpedia Ontology instances per class [13] 13](#_Toc32265715)

[Figure 2: DBpedia extraction framework [12] 14](#_Toc32265716)

# List of Tables

# Glossary

# Abbreviations

# Introduction

## Overview

## Objectives

## Report Structure

# Literature Review

In the past decade, conversational chatbots have seen a surge in popularity. The virtual assistant, such as offerings from Google Assistant and Amazon’s Alexa, are now entering our homes with Internet of Things devices. In 2017, Google Assistant was installed on over 400 million devices [1]. Furthermore, specialized chatbots have seen an influx within banking, retail, and healthcare. Chatbots represent a trend towards using natural language in the realm of human-computer interaction (HCI). This literature review will explore how chatbots are implemented, their benefits, and how this project can innovate within the chatbot space.

## Chatbots

When discussing chatbots, the beginning of their history is usually cited as Alan Turing’s 1950 article “Computing Machinery and Intelligence” [2], wherein Turing describes a test to determine whether a human evaluator can distinguish between a human and a machine during a natural language conversation. This test became known as the Turing test, and asks the question ‘Can machines think?’. However, the goal of many chatbots is not to create true artificial intelligence, but rather to using pattern matching and conversational responses to mimic the responses of a human.

One of the first programs to attempt the Turing test was ELIZA, created by Joseph Weizenbaum between 1964 and 1966 [3]. ELIZA consisted of a language analyser and a set of rules by which the ‘chatterbot’ followed. ELIZA used a script called DOCTOR was designed to simulate responses of a psychotherapist during a psychiatric interview – predominantly achieved by the therapist mirroring the responses of the patient [3]. ELIZA may be considered rudimentary and narrow by today’s standards, it forms the basis of our understanding of chatbots and human computer interaction, and how we can teach machines to mimic human characteristics in dialogue.

ALICE

Google Assistant

Chatbots are ubiquitous in modern life, with most modern smartphones pre-equipped with a virtual assistant such as Google Assistant or Apple Siri.

Specific uses

## Natural Language Processing

## Machine Learning

## Datasets

Typically, chatbots are divided into two groups, open-domain and closed-domain [4]. In an open-domain system, the conversation can go in any direction, and the user can talk to the chatbot about any topic. A closed-domain system is restricted to a narrower topic area or set of function – these are the chatbots we see most in real-world applications such as customer service and banking. For this project, the focus will be on a closed-domain system as the goal is to create a chatbot that can achieve a goal – these are often called Goal-Oriented (GO) Chatbots [4]. However, to create a GO chatbot, one must have a goal the chatbot should achieve, and a dataset from which to learn.

The Ubuntu Dialogue Corpus (UDC), is one of the largest public dialogue datasets available [5], consisting of 1 million multi-turn dialogues from users receiving technical support for Ubuntu-related problems [6]. **continue**

In terms of knowledge bases that lend themselves to the question and answer format, Wikipedia is the world’s largest collaboratively edited source of encyclopaedic knowledge [7]. In terms of size, it eclipses the size of the Encyclopaedia Britannica, its nearest rival, by a factor of ten [8] – as of 12 November 2019, there are over 5.9 million articles in English, and over 51 million articles in the 306 languages officially covered by the Wikimedia Foundation [9]. However, Wikipedia’s content is only fit for human reading [7] and is hard to process computationally. Many attempts have been made to formalise and structure this data, as seen in [7], [8], [10], but this review will focus one of these being DBpedia.

DBpedia is a crowd-sourced effort to extract structured content from various Wikimedia projects [11], including Wikipedia. The English version of the DBpedia knowledge base describes 4.58 million things, out of which 4.22 million are classified in a consistent ontology [], consisting of 320 classes described by 1,650 different properties [12]. This structure enables programs to process this data effectively, including a chatbot application. The size of the DBpedia Ontology is shown in Figure 1, which demonstrates the scale of the project, and how this might be effective for the project.

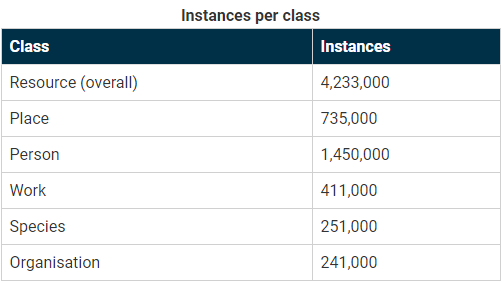


Figure : DBpedia Ontology instances per class [13]

The DBpedia extraction framework is responsible for extracting data from Wikipedia into a structured knowledge base, an overview of which is shown in Figure 2. This extraction is structured into four phases, as described in [12]:

**Input:** Wikipedia pages are read from an external source, either from a Wikipedia dump, or using the MediaWiki API.

**Parsing:** Each Wikipedia page is parsed, which transforms the source code of the Wikipedia page into an Abstract Syntax Tree (AST). An AST is a tree representation of the syntactic structure of the source code.

**Extraction:** The Abstract Syntax Tree of each page is forwarded to the extractors. There are many types of extractors, which will later be described, which extract data such as labels, images and infoboxes. Each extractor takes an AST as input and yields a set of Resource Description Framework (RDF) statements. These are XML statements which describe properties and values of resources.

**Output:** These RDF statements are written into sinks, which receive the data.

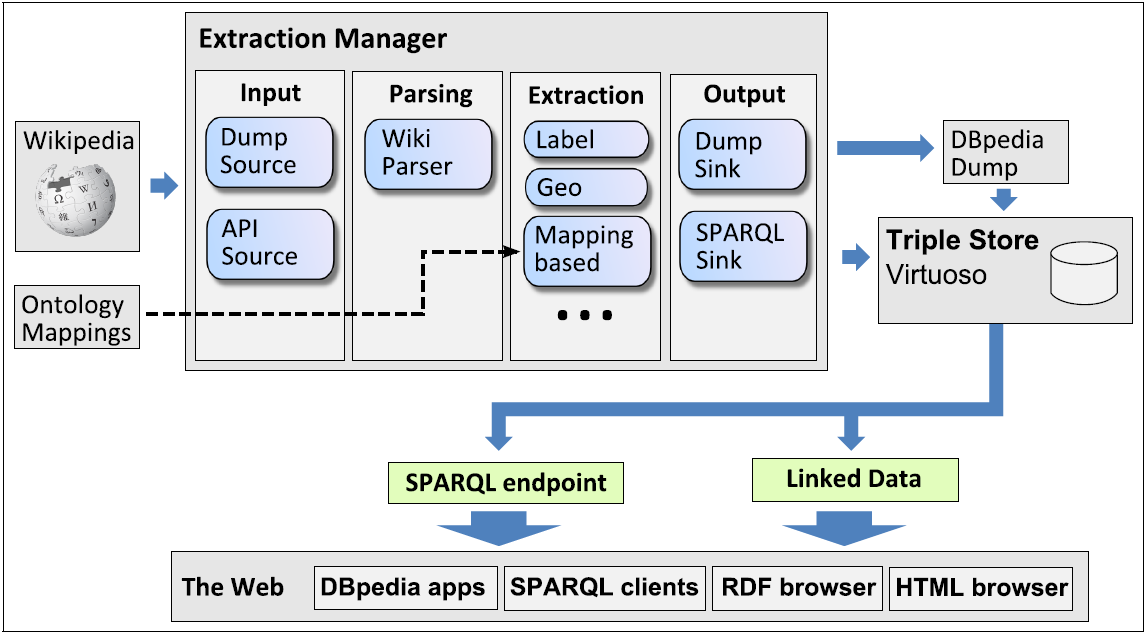


Figure : DBpedia extraction framework [12]

* Ontology
* Endpoint (SPARQL)

## Programming Languages

* AIML
  + ALICE framework
  + <https://ieeexplore.ieee.org/document/7810979>
* Java
* Python
* Libraries

## Existing Solutions

* Google Search/Assistant
* DBPedia chatbot

## Conclusion

# Problem Analysis

# System Requirements

## Requirements Gathering

## Software Requirements

### Functional Requirements

#### 1.0.0 User Interaction

* 1.1.0 The application should allow the user to interact with the chat bot
* 1.1.1 The user should be able to access the chatbot in a browser
* 1.1.2 The user should have a text box to type their query
* 1.1.3 The user should clearly see the response of the chatbot in the webpage
* 1.1.4 The user should be able to clearly see their conversation with the chatbot in the webpage

#### 2.0.0 Queries

* 2.1.0 The chatbot can answer basic questions about people:
  + 2.1.1 The application should take a user query about a person - ‘who is X’ - and respond with a description of that person.
  + 2.1.2 The application should take a user query about the birthdate of a person – ‘when was X born’ and return the birthday of the given person.
  + 2.1.3 The application should take a user query about the age of a person - ‘how old is X’ - and return the age of the given person.
  + 2.1.4 The application should take a user query about the birth place of a person – ‘where was X born’ - and return the birth place of the given person.
  + 2.1.5 The application should take a user query about the death date of a person – ‘when did X die’ and return the birth place of the given person.
  + 2.1.6 The application should take a user query about what a person is known for – ‘what is X known for’ and return a description of what the given person is known for.
  + 2.1.7 The application should take a user query about what a person looks like – ‘photo of X’ or ‘what does X look like’ - and return a photo of the person
  + 2.1.8 The application should take a user query about linking to the Wikipedia page of a person, and return a link to that page.
* 2.2.0 The chatbot can answer questions about countries:
  + 2.2.1 The application should take a query about a country, and return the description of that given country.
  + 2.2.2 The application should take a query about the population of a country, and return the population of that given country.
  + 2.2.3 The application should take a query about the capital of a country, and return the capital of the country.
* 2.2.3 The application should take a query about the description of a country, and return the description of that given country.
* 2.2.4 The application should take a query about the flag of a country, and return the flag image of that given country.
* 2.2.5 The application should take a query about where a country is on a map, and return a google maps location of that country.

#### 3.0.0 Advanced Queries

* + 1. The chatbot can perform advanced query searches and comparisons:
* 3.1.0 Advanced People queries
  + 3.1.1 The user can ask for a list of people born in a given year, e.g. ‘who was born in 1995’ and the chatbot returns a list of notable people born in that year
  + 3.1.2 The user can ask for a list of people born in a given place, e.g. ‘who was born in London’ and the chatbot returns a list of notable people born in that place
  + 3.1.3 The user can ask for winners of a given prize, e.g. ‘who won the Nobel Peace Prize [in 2009]’ and the chatbot returns a list of winners of that prize, or the winner of the prize in a given year.
* 3.3.0 Advanced Country queries
  + 3.2.1 The user can ask the chatbot for a list of countries with a given official language, e.g. ‘which countries speak Italian’, and the chatbot will return a list of countries which identify that language as their official language.
* 3.1.0 Combining queries with AND operator:
  + 3.1.1 The user should be able to combine queries using ‘AND’ to find people who satisfy two conditions. For example, ‘people who were born in 1980 AND were born in London’

#### 4.0 Conversation

* 4.1 The user should greet the chat bot and be returned with a similar greeting – e.g. Hello.
* 4.2 The user should be able to ask for example queries and the chat bot returns a number of working example queries
* 4.3 The user should be able to ask for help using the chatbot and be returned with a statement about how to use the chat bot.

### Performance Requirements

#### 5.1 Performance

* 5.1.1 The web page should load fully in less than 5 seconds
* 5.1.2 The chat bot should respond to each query within 5 seconds

#### 5.2 Reliability

* 5.2.1 The application should function without failure
* 5.2.2 Any errors that do occur during normal operation should be logged, and the user should be clearly informed that an error has occurred.

# Design

## Methodology

* Software development methodology with justification

# Implementation

# Testing

# Project Evaluation

# Conclusion

## Overview

## Conclusions

## Future Work

## Final Statement

# Ethics

# References

|  |  |
| --- | --- |
| [1] | R. Chandra, “How Google Home and the Google Assistant helped you get more done in 2017,” 5 Jan 2018. [Online]. Available: https://blog.google/products/assistant/how-google-home-and-google-assistant-helped-you-get-more-done-in-2017/. [Accessed 5 Nov 2019]. |
| [2] | A. M. Turing, “Computing machinery and intelligence,” *Mind,* vol. 59, no. 236, pp. 433-60, 1950. |
| [3] | J. Weizenbaum, Computer Power and Human Reason: From Judgment to Calculation, New York: W.H. Freeman and Company, 1976. |
| [4] | V. Ilievski, “Building Advanced Dialogue Managers for Goal-Oriented Dialogue Systems,” 16 Mar 2018. [Online]. Available: https://arxiv.org/pdf/1806.00780.pdf. [Accessed 13 Nov 2019]. |
| [5] | S. Kojouharov, “Ultimate Guide to Leveraging NLP & Machine Learning for your Chatbot,” Chatbots Life, 18 Sep 2016. [Online]. Available: https://chatbotslife.com/ultimate-guide-to-leveraging-nlp-machine-learning-for-you-chatbot-531ff2dd870c. [Accessed 13 Nov 2019]. |
| [6] | R. Lowe, N. Pow, J. Pineau and I. V. Serban, “The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructed Multi-Turn Dialog Systems,” 4 Feb 2016. [Online]. Available: https://arxiv.org/pdf/1506.08909.pdf. [Accessed 13 Nov 2019]. |
| [7] | M. Völkel, M. Krötzch, D. Vrandecic, H. Haller and R. Studer, “Semantic Wikipedia,” in *Proceedings of the 15th international conference on World Wide Web*, Edinburgh, 2006. |
| [8] | O. Medelyan, D. Milne, C. Legg and I. H. Witten, “Mining meaning from Wikipedia,” *International Journal of Human-Computer Studies,* vol. 67, no. 9, pp. 716-754, 2009. |
| [9] | Wikimedia Foundation, “List of Wikipedias,” 17 Oct 2019. [Online]. Available: https://meta.wikimedia.org/wiki/List\_of\_Wikipedias. [Accessed 13 Nov 2019]. |
| [10] | F. Wu and D. S. Weld, “Autonomously semantifying Wikipedia,” in *CIKM '07 Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, Lisbon, 2007. |
| [11] | DBpedia, “About | DBpedia,” 2019. [Online]. Available: https://wiki.dbpedia.org/about. [Accessed 13 Nov 2019]. |
| [12] | J. Lehmann and et al., “DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia,” *Semantic Web,* vol. 6, no. 2, pp. 167-195, 2015. |
| [13] | DBpedia, “Ontology | DBpedia,” 2019. [Online]. Available: https://wiki.dbpedia.org/services-resources/ontology. [Accessed 13 Nov 2019]. |

# Appendix