University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: COMPUTER SCIENCE	Spring semester, 2015 Open access
Writer: Maryam Sadat Marashi	..... (Writer's signature)
Faculty supervisor: Dr. Krisztian Balog External supervisor(s):	
Title of thesis: Automatic Generation of Quiz Questions Using Freebase	
Credits (ECTS): 30	
Key words: Freebase, GoogleAPI, FreebaseAPI, Generating Question, Difficulty Level	Pages: 52 + enclosure: Source Code Stavanger, 15.06.2015 Date/year

Automatic Generation of Quiz Questions Using Freebase

Maryam Sadat Marashi

University of Stavanger

A thesis presented for the degree of
Master of Science

Department of Electrical Engineering and Computer Science
University of Stavanger
Norway
June 2015

Abstract

In this thesis an algorithm is designed to generate questions and answers automatically and randomly using Freebase to be used in an online knowledge game, called Game of Gums. In this game, a player chooses a topic based on his/her interest and answers to some questions in order to go further steps in the game.

Freebase is a knowledge base contains millions of entities in different domains. By accessing Freebase data by using Freebase APIs, we can read Freebase data and use entities and their properties for generating questions and answers. Questions in this game are in different category levels of : "Easy", "Medium", and "Hard". These categories are created based on the score of entity that are assigned to that during indexing. The score shows the popularity and number of incoming and/or outgoing links to each entity. Furthermore, a Java program was implemented for each question and corresponding answers using NetBeans IDE.

Finally, surveys are designed and have been sent to interested volunteers in each topic to evaluate quiz questions. The results of these studies were discussed afterward. The results show that most of the respondents could answer easy questions while least of them answer hard ones.

All these steps are explained in this thesis in great detail.

Acknowledgments

I would like to thank my supervisor, Dr. Krisztian Balog for his kind guidance and support. I am grateful for his valuable advises and feedback during development of this master Thesis.

I am very grateful to my beloved husband and my best friend, Ehsan, without whose love and encouragement, I would not have finished this thesis.

I would also like to thank my parents and my brothers, for their great support and love through my entire life.

A very special thanks goes out to my friends, specially Reza, Maziar and Patricia who helped me during my studying.

Contents

Chapter 1

Introduction

"Gumpaper is a platform to build social networks among people who share interests, activities, backgrounds or real-life connections. It consists of a representation of a person, company or organization, his or her social links, and a variety of additional services, such as news feeds, games and comments[1]".

They are going to design an online knowledge game call Game of Gums for their website. This game is a board game that in each cell player should answer to a question to move forward in the game. The questions are about the different topics that define in the website.

The goal of this project is to develop Automatic Generation of Quiz Questions or "AGQQ" system; a system to generate these question and answers.

1.1 Problem Statement and Overview

The task is as follows: Design an algorithm for generating questions and answers automatically and randomly for the online quiz game from a knowledge base.

In this online quiz game, players choose a topic based on their interest. They move around the gameboard by answering a question in chosen topic. For example if a user interested in art one of the possible questions can be: *Which one is not Edvard Munch painting?* And the relevant answers are: *The Elephant*, *The Scream*, *The Dance of Life* and *Madonna*. By choosing the correct answer, the player can move forward.

We use the Freebase[2] knowledge base, which currently includes more than 47 million topics and 2,9 billion facts. Also, it contains thousands of types and properties. Each topic in Freebase is linked to other related topics

and has some important properties like a person's name or an an artist's artwork.

For designing AGQQ Freebase APIs are used to retrieve data from Freebase. An API (application programming interface) is a set of routines and tools to build software applications. An API reveal component of a software in terms of its inputs, outputs, and its operations.

The Freebase API is a set of APIs that allow read and write access to the data that are stored in Freebase. The current Freebase API is designed to work with the Google API Client Libraries. Google APIs is a collection of APIs developed by Google, which allow communication with Google Services. These APIs are integrated into other services.

In this project, the AGQQ algorithm is designed for this online game based on the "topics", "types" and "properties". The quiz question types are multiple choice. So, for designing the answers we use the properties of topics. The correct answer is the one that has the property of the topic that has a relationship with the topic, and the other answers are the same property of other topics in the same type.

These questions are categorized into three levels of difficulty: "Easy", "Medium" and "Hard". These levels are determined based on the one of the properties of topics which is "score". A score of each topic was computed during indexing by Freebase based on the links count in Freebase and Wikipedia.

To reach AGQQ algorithm we designed some algorithms for different topics and various types of questions and then a generalized algorithm is developed which is a pattern for creating many quiz questions.

AGQQ algorithms are implemented in Java. Also, a User Interface is designed for testing the result of the program.

To evaluate the approach surveys for chosen topics are designed and send to some people. The results of these studies show that most of the respondents could answer to the easy questions while least of them answer the hard questions correctly. These results shows that questions are designed in a way that easy questions are not as much easy that they are answered by all of the respondents and hard questions are not that much hard that respondents could not answer to one of them.

1.2 Organization of the thesis

Chapter 2 contains some related works that we inspired from about question and answer systems, structured data, and Freebase.

In chapter 3, the data source of this work is explained which is Freebase. Freebase serves as the knowledge base and contains entities, type, and properties. This chapter described Freebase in significant details.

Chapter 4 talks about the algorithms that have been developed for AGQQ. Also, the other APIs that are used for implementing these algorithms are explained in details.

The results, as well as how the evaluation of the system has been done and the results of evaluations, are discussed in chapter 5.

Finally, Chapter 6 provides a conclusion with a short note on the future work.

Chapter 2

Related Works

Generating question from a knowledge base faces prominent challenges related to model and data source. These challenges involve finding the best representation of the question, converting it into a query and executing the query on the knowledge base. An approach followed in this project was inspired by some papers about information retrieval and semantic parsing for generating questions.

"Yao et al.,[3] proposed an automatic method for Question Answering from a structured data source (Freebase). They viewed a Knowledge base as an interlinked collection of topics. When given a question about one or several topics, a view of the KB concerning only involved topics can be selected, then every related node within a few hops of relations to the topic node will be inspected in order to extract the answer[3]". The view is called a topic graph by assuming that it is possible to find the answers within the graph. The goal of the study was to make the answer extraction process maximally automatic.

By having a topic, a topic graph should be formed. So, the Freebase graph will be searched by choosing the nodes that have a relationship with the topic node. Each node has incoming and outgoing relationships with the topic node. Also, nodes have properties: a string that explains the attribute of a node, for example, node type, artwork or artform for an artist. The most significant difference between relations and properties is that a relation is between two nodes, while only one side of a property is a node. Arguments of relations are usually interconnected, e.g., Mona Lisa can be the artwork for Leonardo Da Vinci. Arguments of properties are attributes that are only specified for individual nodes and have no outgoing edges.

To retrieve the answer, both relationship and property of a node are

necessary. The nodes are connected to the question with relationship and property, and they explain some unique characteristics of the nodes. As an example, without the properties type:artwork and artform:painting, we would not have known the node Mona Lisa represents a painting. These properties are important cues for answering the question. As a result, the relations and properties should be used as features for each node for the Freebase graph.

In that approach, question sentence was divided into following parts:

- Question Word (qword)

  Like what/who/how many. Nine common qwords were used.

- Question Focus (qfocus)

  a cue of expected answer types(e.g. name/money/time). Yao and his team keep their analysis simple. They simply find the noun dependent of qword as qfocus instead of using a question classifier.

- Question Verb (qverb)

  Like is/play/take, obtained from the primary verb of the question..

- Question Topic (qtopic)

  The topic of the question is helpful to find relevant Freebase pages. A named entity recognizer can be applied to find the question topic.

WEBQUESTIONS dataset is an enormous dataset containing 3778 training questions and 2032 test questions collected from the Google Suggest API "Xuchen et al. [4], found that 85% of all questions (in the training set) can be directly answered via a single binary relation for the task of question answering (QA) over Freebase on the WEBQUESTIONS dataset. Thus, they turned this task into slot-filling for ¡question topic, relation, answer¿ tuples: predicting relations to get answers given a questions topic[4]". To identify question topics organically from millions of Freebase topic names, an efficient data structure was designed without using any NLP processing tools. Then the authors present a lean QA system running in real time.

In [5], a problem related to mapping natural language questions into structured queries was studied. "In this work, Yahya et al. tackle questions built from phrases corresponding to entities, classes and relations (collectively known as semantic items). The task is to disambiguate the question with respect to the data at hand by correctly segmenting the question, mapping segments to the appropriate entities, classes or relations, and grouping them together to capture the structure of the question[5]".

It should be mentioned that the referred papers above discuss the question answering problem. However, their approach to generating questions was only followed in this project, not the answering part.

Chapter 3

Data Set

3.1 Freebase

The data source that is used for this project is Freebase. Freebase is a massive knowledge base including over 47 million topics about well-known people, places, and things. The data are collected mainly by community members from many sources, including individual and user-submitted wiki contributions.

Freebase goal is to provide a global resource of data freely under an open license for commercial and non-commercial use. So, people and machines are allowed to have access common information effectively[6].

3.1.1 Freebase Graph

Freebase works by storing data as a graph. A graph is created on nodes connected by edges. The nodes are defined with ***/type/object*** and also edges are defined with ***/type/link***. In general, a graph represents a Freebase data and topics about real world entities correspond to the nodes in the graph. Although, it should be noted that every node is not a topic.

3.1.2 Topics

Storing the data as a graph allows Freebase to traverse arbitrary connections between topics quickly and add new schema easily without changing structure of the data[7].

Following types of topics can be found in Freebase as some examples:

- Physical entities

e.g., *Pablo Picasso*

- Media creations
  e.g., *A beautiful Mind (Movie)*
- Classifications
  e.g., *Cetacea*
- Abstract concepts
  e.g., *awesomeness*
- Schools of thoughts
  e.g., *Modernism*

Topics can be notable if they hold a lot of data or they link to many other topics, probably in other domains of information. *Football* is an example for a topic with lots of data. Topics like *humour* and *drama* are prominent because they are linked to other topics. For example, they often appear as book, poetry, and film subjects.

Any given topic can be found for different aspects for example:

- *Pablo Picasso* was a painter, sculptor, playwright, printmaker, ceramicist, poet, ...;
- *Shaquille O'Neal* is a basketball player, singer, TV actor, producer, ...;
- *Drama* is a book genre, film subject, theater genre, literature subject, ...;

3.1.3 Types

The concept of type is introduced in Freebase to capture different perspectives of a topic, Any number of types can be assigned to the topics. For instance, the topic about *Shaquille O'Neal* is assigned several types: the basketball player type, the music composer type, the musical artist type, the TV actor type, etc.

3.1.4 Properties

Each type has a different set of properties related to that. For instance:

- The person type contains much property for listing date of birth, place of birth, country of nationality, gender, profession, religion, etc.

- The visual artist type contains a property that lists all the art form that *Pablo Picasso* has produced as well as all of his artworks;
- The musical artist type contains a property that lists all the tracks *Shaquille O'Neal* has recorded, as well as all of his albums;

As a result, a type includes necessary properties to describe a particular aspect of information.

3.1.5 Domains

Types are grouped into domains in such a way that properties are grouped into types. Domains can be considered as the sections in a newspaper: Business, Life Style, Arts, Weather, Economics, Politics, Sports, etc.

3.1.6 IDs

Each domain has an ID (identifier), e.g.,

- `/music` is the ID for the Music domain
- `/sports` is the ID of the Sports domain
- `/visual_art`, the ID for Visual Art domain, and
- `/computer` is the ID of the Computers domain

The ID of a domain is similar to a file path, or a path to a web address. Also, each type has an ID according to the domain in which it belongs. For instance, the Musical genre type belongs in the Music domain, and it's given the ID `/music/genre`. Here are some other examples:

- `/sports/sports_team` is the ID of the Sports Team type, belonging in the Sports domain
- `/visual_art/artwork`, the Artwork type in the visual_art domain

A property inherits the beginning of its ID from the type it belongs to similar to a type. For example, the Artists property of the Musical genre type (used for listing artists a genre has) is given the ID `/music/genre/artists`. Here are some other examples:

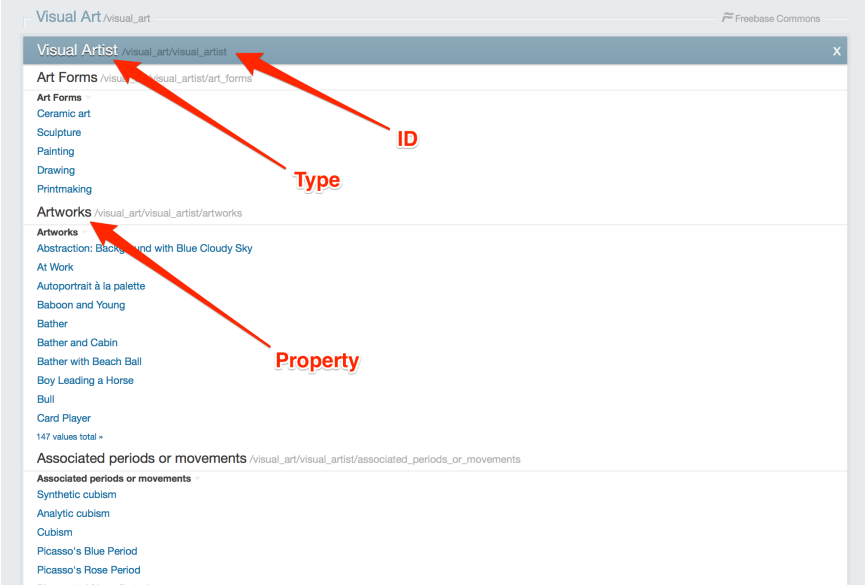

Figure 3.1: Type, Properties and IDs in Pablo Picasso's Page in Freebase

- `/sports/sports_team/championships` is the ID of the Championships property of the Sports Team type (used for specifying championships a sports team has)
- `/visual_art/artwork/art_form` is the ID of the Art form property of the Artwork type (used for listing the art form of an artwork)

Figure 3.1 shows type, property and their IDs for *Pablo Picasso* topic in Freebase.

Although in Freebase types are not arranged into hierarchies; IDs for domains, types, and properties are arranged in a file directory-like hierarchy.

3.1.7 MIDs

A topic may be identifiable by `namespace/key` IDs or not. However, it can always be identified with a Machine Identifier (MID), which is formed by `/m/` followed by a base-32 unique identifier. MIDs are assigned to topics for the topic's lifetime. They have a very important role in merging or splitting of topics. They allow external applications to track the logical topic even if the physical Freebase identity (the topic's GUID) may change. IDs generated by machines are different from other Freebase IDs, which can be read by

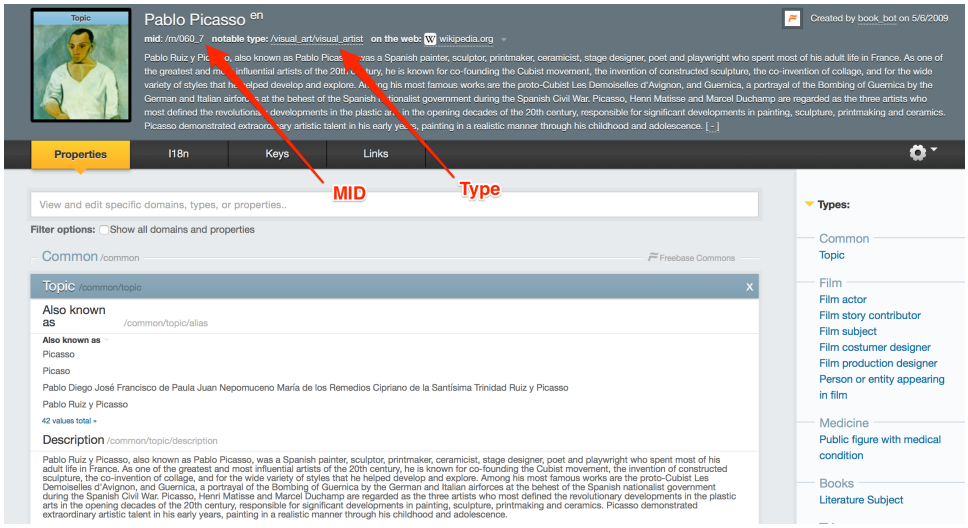

Figure 3.2: MID and Type in Pablo Picasso's Page in Freebase

human (returned by the "id" property). They have following specifications [7]:

- Guaranteed to exist
- Generated by machine
- Design to support offline comparison
- Short (with fixed length)

It is recommended to use MIDs as an identifier to address topics in Freebase. Figure 3.2 shows MID and type for *Pablo Picasso* topic in Freebase.

3.1.8 Namespace and Key

"The file directory-like hierarchy of domain, type, and property IDs is just one application of a more general concept: *namespaces* and *keys*. A namespace is like a file directory, and a key is like a file name. Just as all file names within a particular file directory must be unique among themselves, all keys within a particular namespace must also be unique among themselves [7]".

For example, `/music` is the namespace connected to the Music domain.So, Music-related types have keys within the Music domain (e.g., *genre*). These

keys are unique among themselves. A type's ID is created by adding its key to the namespace's ID (e.g., `/music/genre`).

There are many namespaces correspond to domains and types. `/en` namespace is the most important one that is referred to English namespace. In this namespace most well-known topics have unique keys to form human-readable IDs. As an example, the *Mona Lisa* is so well known that its topic in Freebase is given the key *mona_lisa* in the `/en` namespace, and so the topic's ID is `/en/mona_lisa`. So, everyone can have access to its topic in the web client with the simple URL because of this ID.

Chapter 4

Design and Implementation

This project aims to develop AGQQ, a system for generating quiz questions and corresponding answers automatically by using a knowledge base for a given topic and a specified difficulty level of the questions.

First, an overview of some terms for this project is provided in the following:

- Data Sources

  There are many data sources that could be used for generating quiz questions. We can categorize these data sources to structured data, semi-structured data, or free text. The focus of this project is on the structured data, so, Freebase is used which belongs to this category.

- User

  The user of this project is defined as a person interested to play the game.

- Questions

  Multiple choice questions are designed for different domains in Freebase with three difficulty levels of "Easy", "Medium", and "Hard". These questions are developed based on entities and/or properties of entities.

- Answers

  There are four answers to each question, and only one of them is the correct answer. These responses are designed based on one or more properties of an entity.

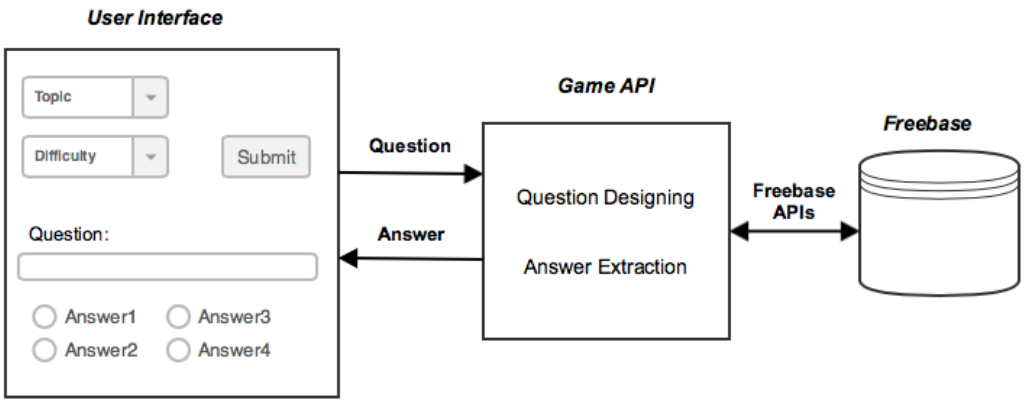

Figure 4.1: System's Architecture

- Difficulty

  The questions and answers should be designed in such a way that the user can simply answer to easy questions while finds it difficult to respond the hard questions.

4.1 Generating Questions

Figure 4.1 shows the architecture of the system. In this system, the user chooses a topic from a topic list and specifies the level of difficulty. By pressing the submit button, the interface will be interact with the program, which is connected to Freebase. Then, a question is generated randomly based on the entities of the chosen topic.

Answers are made based on the relation between entities and their properties. The correct answer is one of the properties of the entity in the question while the other answers are the same property but from different entities in the same topic area.

There are 19 different categories for different topics in this quiz game: *Animal*, *Art*, *Beauty*, *Education*, *Engine*, *Entertainment*, *Food*, *Hobby*, *Job*, *Media*, *Music*, *Nature*, *Philosophy*, *Politics*, *Royalties*, *Science*, *Society*, *Sports* and *Travel*.

In this project 3 topics, *Art*, *Music* and *Sports* are chosen and an algorithm is designed for these topics.

The difficulty level is determined based on the relevance score of entities

that will be explained later. The highest score in each difficulty category belongs to the most prominent entity, and the lowest score belongs to the least prominent one. We define three kinds of questions: "Easy", "Medium" and "Hard". The Easy questions are the ones about more prominent topics, so, they have the highest scores while hard questions are about the less famous topics with the lowest scores. The medium questions are neither easy nor hard with the moderate scores.

For generating questions and answers automatically from Freebase, some algorithms are designed based on entities and properties for each domain. First a simple and an advanced algorithm will be explained and then a general algorithm that is a pattern algorithm for designing many quiz questions.

4.1.1 Simple Algorithm

First step for designing the algorithm is to define some questions to be asked in the quiz game. For instance, in order to generate a question for *Music*) with the `/music` id and many types of entities like *rock and roll* as `/music/genre`, one of the generated questions can be: *Which genre is not a parenting genre for Rock and Roll?* with answers: *Rock*, *Country*, *Swing*, *Folk*. Another question can be *Which singer is the best known for Blues?* with the answers likes *Tom Jones*, *Chuck Berry*, *Bob Dylan* and *Johnny Cash*.

To generate the questions automatically, a simple algorithm is designed for a simple question first: *who is the artist of X?*

Figure 4.2 shows a simple flowchart for generating this question and corresponding answers:

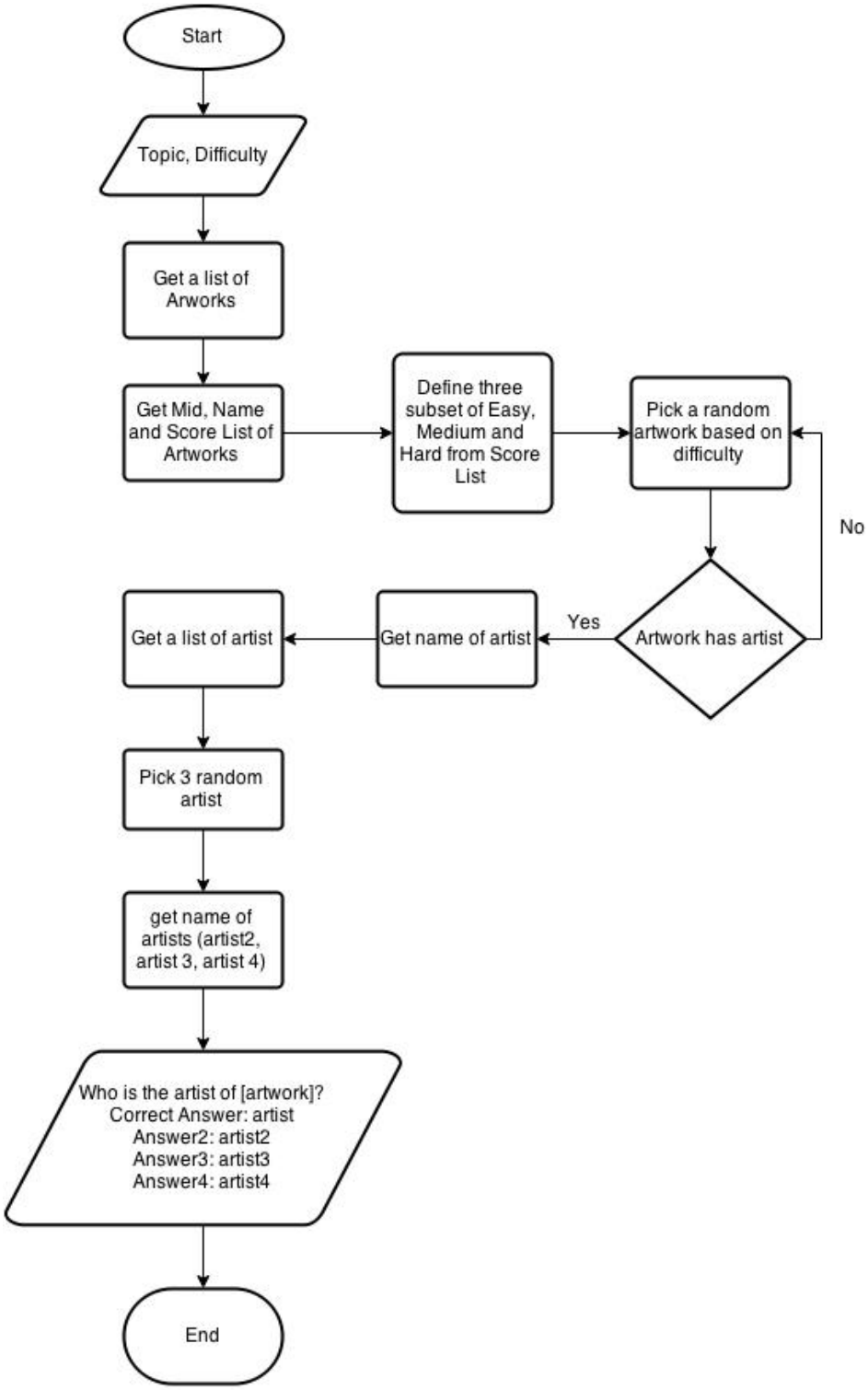

Figure 4.2: Simple Flowchart

The following algorithm shows how to generate this question:

Algorithm 1: Simple Algorithm

Data: topic, difficulty
Result: Who is the artist of [artwork]?, Correct answer, answer2, answer3, answer4

```
1  Get a list of artworks;
2  midList ← mid
3  nameList ← name
4  scoreList ← score
5  EasyList, MediumList and HardList are sublists of scoreList;
6  Select a random score from scoreList;
7  Specify TargetScoreList based on score;
8  Get mid and name;
9  if mid contains artist then
10 |  correctanswer ← artist
11 else
12 |  goto 6;
13 end
14 Get a list of artists;
15 Select three random artists from artists List;
16 answer2 ← artist1
17 answer3 ← artist2
18 answer4 ← artist3
```

The conditions in this algorithm are:

- Finding an artwork that has an artist in its properties
- Finding three artist for other options of answers

In this algorithm for specifying the difficulty level of question, the scoreList of artworks is divided into three sublists: "EasyList", "MediumList" and "HardList". This division is done based on the popularity of artworks for people who are interested in *Art* topic.

the correct answer is the name of the artist, and the other answers are the name of three other artists.

4.1.2 Advanced Algorithm

The following flowchart and algorithm is the most complex one that is designed for generating this question: *Which one is not a/an [artform] by [artist]?*:

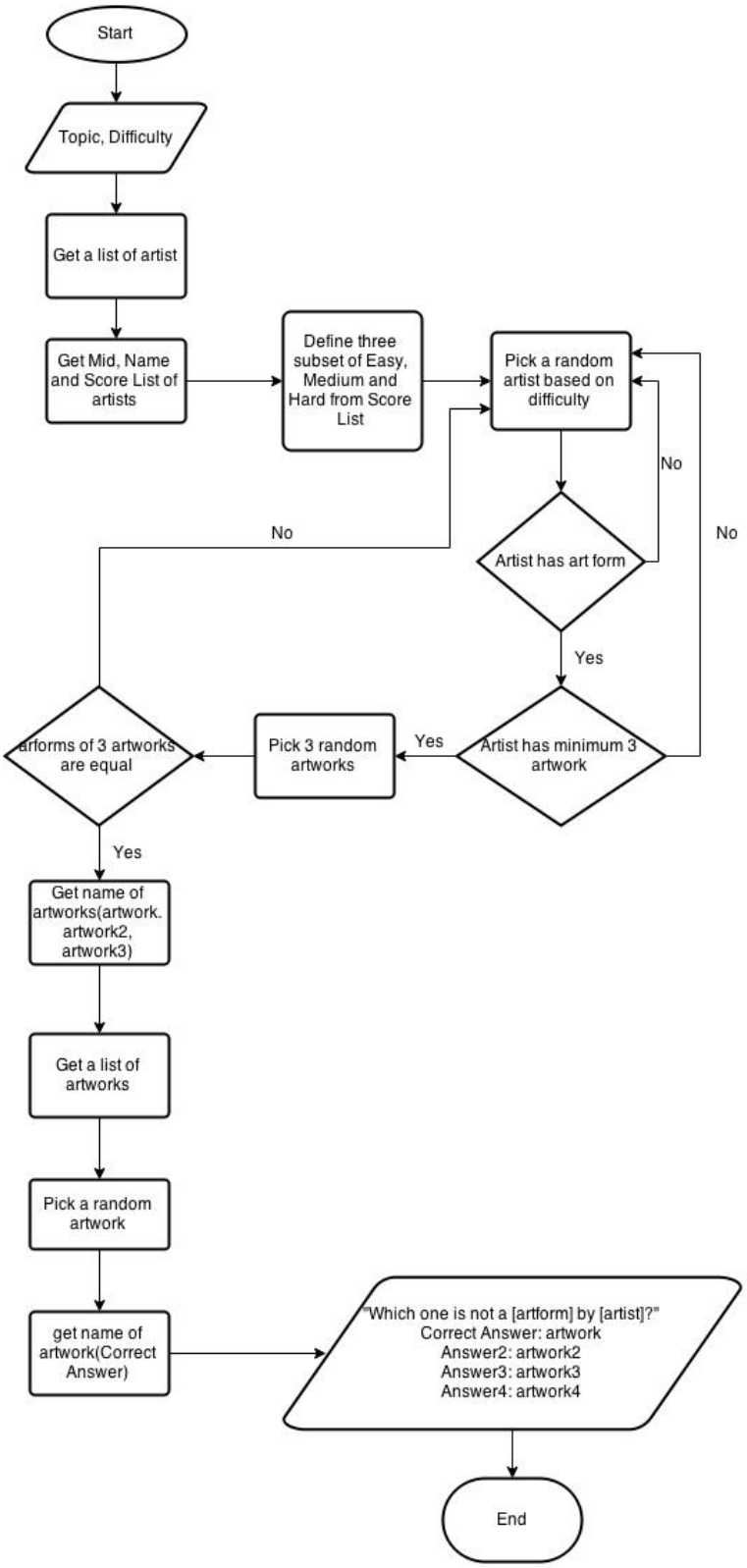

Figure 4.3: Advanced Flowchart

Algorithm 2: Advanced Algorithm

Data: topic, difficulty
Result: "Which one is not a/an [artform] by [artist]?", Correct answer, answer2, answer3, answer4

```
1  Get a list of artists;
2  midList ← mid
3  nameList ← name
4  scoreList ← score
5  EasyList, MediumList, HardList are sublist of scoreList;
6  Select a random score from scoreList;
7  Specify TargetScoreList based on score;
8  Get mid and name;
9  if mid contains artform then
10     if mid contains artwork then
11         if artworks ≥ 3 then
12             Select 3 random artworks;
13             if artforms of these artworks are equal then
14                 foreach artwork in artworkList do
                   answer2 ← artwork1
15                 answer3 ← artwork2
16                 answer4 ← artwork3 ;
17             else
18                 goto 12;
19             end
20         else
21             goto 6;
22         end
23     else
24         goto 6;
25     end
26 else
27     goto 6;
28 end
29 Get a list of artworks;
30 Select a random artwork from artworks List;
31 correctAnswer ← artwork
```

In this algorithm, we are looking for artwork that is not created by an artist in the question. We cannot easily extract the answer to this question by one Topic API. The number of conditions to be checked is higher in the

advanced algorithm compared to the simple one. To find the answer, we need to verify the following conditions:

- Finding an artist who has both art form and artwork in its properties
- The artist has at least three artworks
- The artform property of artworks is the same as the requested artform
- Finding an artwork from another artist for the wrong answer

For generating this advanced algorithm, it should be noted that property and its relations with an entity are essential to identifying the response to a question. They allow the entities in the question to connect to the correct answer. For instance, without the properties `/visual_art/visual_artist/artworks`:*Mona Lisa* and `/visual_art/artwork/art_form`:*Painting*, we would not have known that *Mona Lisa* is a painting by *Leonardo da Vinci*. So, these properties allow us to find the answer of these types of questions in Freebase.

4.1.3 Generalized Algorithm

For generating questions for different topics, we need to have a general algorithm. This algorithm is:

Algorithm 3: Generalized Algorithm

Data: topic, difficulty
Result: "Which one is not a/an [X] by [Y]?", Correct answer, answer2, answer3, answer4

Get a list of Y;
$midList \leftarrow mid$
$nameList \leftarrow name$
$scoreList \leftarrow score$
EasyList, MediumList, HardList are sublist of scoreList;
Select a random score from scoreList;
Specify TargetScoreList based on score;
Get mid and name;
if *mid contains X* **then**
    if $NumberofX \geq 3$ **then**
        Select 3 random X;
        $answer2 \leftarrow X1$
        $answer3 \leftarrow X2$
        $answer4 \leftarrow X3$
    else
        goto 6;
    end
else
    goto 6;
end
Get a list of X;
Select a random X from XList;
$correctAnswer \leftarrow X$

This algorithm is a pattern algorithm for AGQQ. So, for each question type we may need to change this algorithm as we explained for Simple and Advanced algorithm.

For implementing this algorithm, first we should access to Freebase data. For this purpose, Freebase Search API is used to extract a list of Ys with their relevant "mid", "name" and "score". Then a random Y is chosen its target Score sublist is determined. Moreover, for extracting the properties of Ys, we use Freebase Topic API, which gives us the corresponding Xs of

the artwork.

4.2 Freebase Search API

For implementing the above algorithms, we should access to Freebase data. For this purpose, Freebase Search API is used.

The Freebase Search API provides access to data stored in Freebase Using query or filter. The results of Search API have a relevancy score and are sorted decreasingly.

The following Java code shows how to perform a search to get a list of artworks:

```
GenericUrl url = new GenericUrl(
   "https://www.googleapis.com/freebase/v1/search");

url.put("filter", "(all type:/visual_art/artwork)");
url.put("scoring", "entity");
url.put("limit", "100");
url.put("key", apiKey);
```

As we can see from the above code, we can apply some parameters to search API. These parameters are explained below:

4.2.1 Filter

We can apply filters to Freebase Search API to force the search results to the specified types of data.

The Search API supports some filter constraints to get the correct entities as search result. For example, by using a "type" filter constraint, a list of the most notable artworks in Freebase is shown:

filter: `"(all type:/visual_art/artwork)"`

Filter constraints can accept a variety of inputs:

- Freebase IDs, for example:
  - `/visual_art/artwork` for a type constraint
  - `/sports` for a domain constraint
- Freebase MIDs, for instance:
  - `/m/01y2hcq` for the same `/visual_art/artwork` type constraint

  - `/m/02566hr` for the above `/sports` domain constraint

- Name of entities, for example:
  - `"artwork"` for a less precise `/visual_art/artwork` type constraint
  - `"sports"` for a less precise `/sports` domain constraint

By inserting a modifier between the operand and the text field we can specify a way that the match should occur.

- word: determine that the words in the string be matched with words in the text field in Freebase.
- phrase: specify that the words must be occurred next to each other in the same order in the text field.
- full: it is like phrase but also specify that the phrase must be completely matched with the text field, not just within the text field.

For example, to find the men's national basketball team, use a filter like the following:

Filter :

filter: `"(all name{phrase}:men's national basketball team type:/sports/sports_team)"`

Return :


```
{
   "result":[
      {
         "name":"United States men's national
            basketball team",
         "mid":"\/m\/0mbln",
         "notable":{
            "name":"Basketball Team",
            "id":"\/basketball\/basketball_team"
         },
         "id":"\/en\/united_states_national_
```

```
                basketball_team",
            "lang":"en"
        },
        {
            "name":"Lithuania men's national
               basketball team",
            "mid":"\/m\/07xz18",
            "notable":{
               "name":"Basketball Team",
               "id":"\/basketball\/basketball_team"
            },
            "id":"\/en\/
               lithuania_national_basketball_team",
            "lang":"en"
        }
    ]
}
```

4.2.2 Scoring and ranking

As we mentioned before, questions are categorized according to the level of difficulty. These categories are determined based on the score of entities.

Each Freebase entity has a relevance score that is computed during indexing. This score is based on its inbound and outbound link in Freebase and Wikipedia.

Moreover, some prominent Freebase entities have a popularity score that is computed by Google. Both scores are combined by default.

Scores of entities indicate the popularity of entities. They are prominent for ranking the result entities in the way that result entities are sorted by the scores and entities with the highest score first. As a consequence, the more people search for an entity, the more popular it is.

The scoring parameter controls that for computing the final score what relevance score components are used:

- freebase: Use the Freebase relevance score.

  filter : `/visual_art/artwork` scoring: freebase

- entity: Use the Freebase relevance score and Google score, and replaces missing Google scores with 1.0.

  filter : `/visual_art/artwork` scoring: entity

- schema: Use when we are looking for schema entities like domains, types or properties.

  filter : `/visual_art/artwork` scoring: schema

  HTTP Request :

  `https://www.googleapis.com/freebase/v1/search`
  `?filter=/visual_art/artwork&scoring=entity&key=APIKey`

  Parameters :

  filter : `/visual_art/artwork`
  scoring : entity
  key : API key

  Status Code :

  200 OK : valid key

  Return :

```
{
   "result":[
      {
         "score":24.442667,
         "name":"Statue of Liberty",
         "mid":"\/m\/072p8",
         "notable":{
            "name":"Neoclassical Structure",
            "id":"\/m\/07xnmw"
         },
         "id":"\/en\/statue_of_liberty",
         "lang":"en"
      },
      {
         "score":23.272909,
         "name":"Mona Lisa",
         "mid":"\/m\/0jbg2",
         "notable":{
            "name":"Renaissance Artwork",
            "id":"\/m\/06cvx"
         },
         "id":"\/en\/mona_lisa",
         "lang":"en"
```

```
      },
      {
         "score":22.351675,
         "name":"Rosetta Stone",
         "mid":"\/m\/06l5c",
         "notable":{
            "name":"Relief Artwork",
            "id":"\/m\/02nnps"
         },
         "id":"\/en\/rosetta_stone",
         "lang":"en"
      }
   ],
   "status":"200 OK"
}
```


For categorizing difficulty level of each topic, we asked two people who are interested in each of the topic to classify the entities. In a way that we ask them several questions based on the entities in list of scores retrieve from the Search API, and asked for the popularity of entities on each level. So, for example in a list of 100 entities retrieve from `https://www.googleapis.com/freebase/v1/search?filter=/visual_art/artwork&scoring=entity&key=APIKey` we obtain that first 30 entities with higher scores should be in EasyList, from 30 til 60 in MediumList and from 60 ta 100 in HardList.

4.2.3 Limit

It shows how many result should be returned in the result. By default, 20 matches are returned in decreasing order of relevancy score. Fewer or more matches can be requested by using the `limit` parameter.

4.2.4 API Key

To use Freebase APIs, an API key is needed. It is a unique key that is generated by Google API Console. When our application needs to call an API, this key is passed by the application to all API requests as a `key=API_key` parameter. There is not any user action requires for using this key and also it does not allocate access to any account information or using for authentication.

For accessing to all the properties and relation related to a topic, Freebase Topic API is used.

4.3 Freebase Topic API

The Freebase Topic API is a web service for returning all the facts for a given entity in Freebase likes images or any other property. Like Freebase Search API, filters can be applied to the Topic API for only returning the property values that we're interested in.

Here are some examples of how we would use the Freebase Topic API in our project:

- Getting the text or image for a topic.
- Getting the corresponding property of the topic that is required in the question.

In the below example, we filtered the response to only show property values from the /visual_art domain for the topic "Mona Lisa".

HTTP Request :

`https://www.googleapis.com/freebase/v1/topic/mona_lisa`
`?filter=/visual_art&key=APIKey`

Parameters :

filter : `/visual_art/`
key : API key

Return :

```
{
   "id":"/m/0jbg2",
   "property":{
      "/visual_art/artwork/art_form":{ },
      "/visual_art/artwork/art_genre":{ },
      "/visual_art/artwork/artist":{ },
      "/visual_art/art_subject/
         artwork_on_the_subject":{ },
      "/visual_art/artwork/media":{ },
      "/visual_art/artwork/support":{ },
```

```
      "/visual_art/artwork/date_completed":{ },
      "/visual_art/artwork/dimensions_meters":{
         },
      "/visual_art/artwork/date_begun":{  },
      "/visual_art/artwork/period_or_movement":{
          },
      "/visual_art/artwork/owners":{ },
      "/visual_art/artwork/art_subject":{ },
      "/visual_art/artwork/locations":{ }
   }
}
```

4.4 Freebase Image API

For designing some questions, we need to access to the image of an entity. So we need to access to Freebase Image API.

An image is a particular type of object value in Freebase because the it is not stored in the Freebase graph. A Freebase ID was assigned to the image to retrieve it from the Freebase Image API.

By using the image ID "id": `/m/0jbg2` which is MID for Mona Lisa, an Image API request is created by appending the ID of the image to the Freebase Image API.

By default, Freebase Image API returns a minuscule image. We add maxwidth and maxheight parameters to make it larger:

HTTP Request :

`https://googleapis.com/freebase/v1/image/m/0jbg2&key=APIKey&maxwidth=250&maxheight=250`

Parameters :

key : API key
maxwidth : 250
maxheight : 250

Return :

Figure 4.4

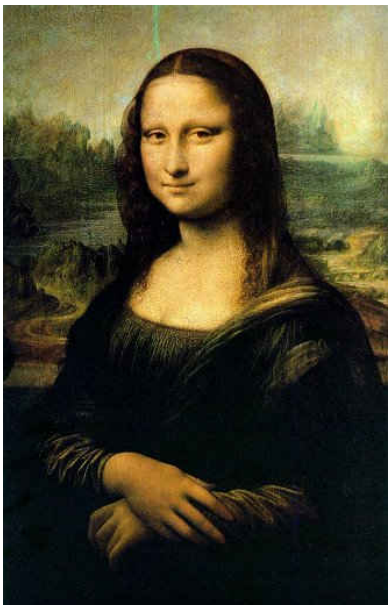

Figure 4.4: The Mona Lisa requested from Freebase Image API

Chapter 5

Results

In this section, we describe the results of designing AGQQ system and explain how the system was evaluated. We developed this system using Java and NetBeans IDE.

In this project, an algorithm is designed for generating questions and answers. Three domains have been chosen for generating questions, which are Art, Music, and Sport. Also, an interface is designed for testing this algorithm and also for getting the results of the queries with an image. This interface is shown in Figure 5.1. User Interface works in such a way that the user can choose a topic, and a level of difficulty and by clicking the Submit button, the question will display. In the current interface by clicking the submit button user will get the next question.

To evaluate this system, three surveys for three domains of Art, Sport and Music were designed in `https://www.surveymonkey.com/` which is an online survey development cloud base website. It provides customizable surveys that include question designing tools, data analysis, and data representation tools. Each survey contains 12 questions in defined difficulty levels, and each level has four questions. Each question has a list of four answers. Example questions and answers includes:

1. Who is the painter of The Scream?
   (a) Donatello
   (b) Pablo Picasso
   (c) Francisco Goya
   (d) Edvard Munch
2. Which country was the champion of 1958 FIFA World Cup?

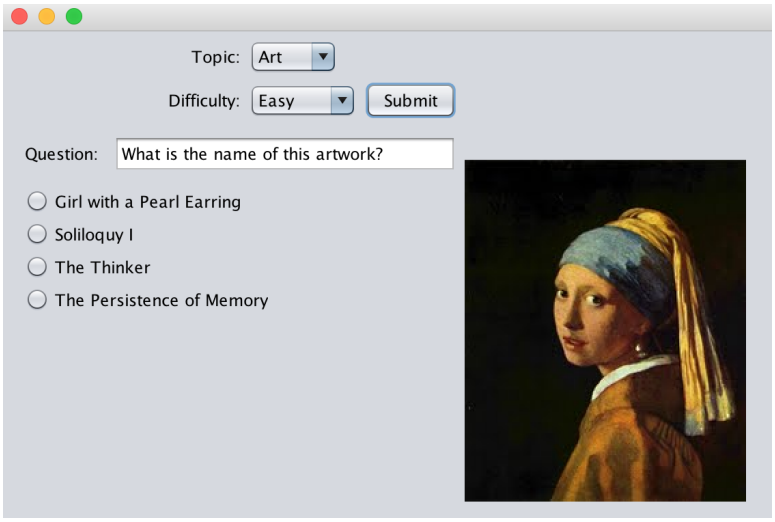

Figure 5.1: User Interface

(a) Brazil national football team
(b) Mexico national football team
(c) France national football team
(d) Germany national football team

3. Which song was not recorded by Pink Floyd?

(a) Bring the Boys Back Home
(b) Vera
(c) I Will Survive
(d) The Show Must Go On

Questions with different difficulty levels were lined up without any structure in each survey and sent to 10 persons who are interested in each of these three topics to answer the questions.

The following charts are the results of this evaluation:

Figure 5.2 shows that on average, seven from 10 respondents could answer easy questions, more than five people could respond to the medium question and less than 4 of them answer hard questions.

In Figure 5.3 for Music topic in average, more than six people could answer easy questions, more than four persons could respond to the medium question and less than 3 of them answer hard questions.

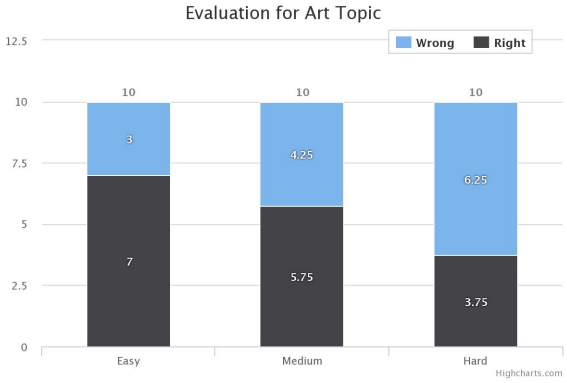

Figure 5.2: Evaluation for Art Topic

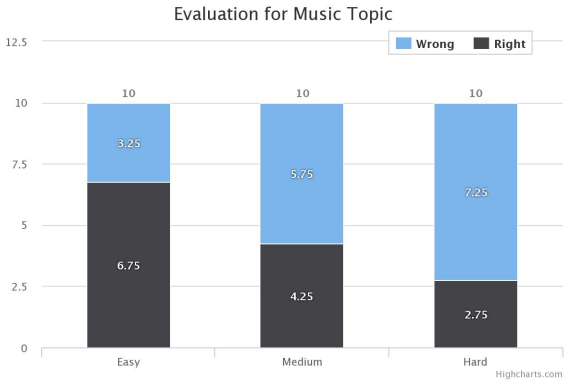

Figure 5.3: Evaluation for Music Topic

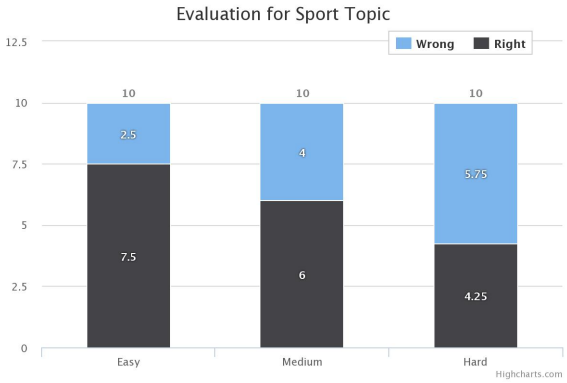

Figure 5.4: Evaluation for Sport Topic

Figure 5.4 shows that on average, more than seven from 10 respondents could answer easy questions, six people could answer the medium question and less than 5 of them answer hard questions.

As we can see from the above charts, the questions are designed in such a way that not all the respondents could answer the easy questions, and not none of them answers to hard questions.

These charts show then how accurate the system is. Based on these results we see that most of the people could answer to easy questions while some of them could answer to hard questions.

Chapter 6

Conclusion

In summary, we have presented AGQQ an automatic generating questions system for an online quiz game with Freebase Knowledge base and implemented this system in Java. Freebase contains many domains, entities, and properties. In this project we have chosen three domains which are Art, Music ans Sport and design an algorithm for generating questions and answers automatically and randomly. These questions have been designed to different levels of difficulty that are Easy, Medium and Hard.

For each topic, some questions are designed based on the various entities and properties and their relations. For instance, in case of painting, which the question is about to find the artist of a painting, we can find the name of artist in `/visual_art/artwork/artist` property of the painting. The other answers for this questions would be the name of three random artists. For example:

- Who is the artist of The Last Supper?

    1. Leonardo da Vinci
    2. Haroldo Gonzlez
    3. Michelangelo
    4. Pablo Picasso

We should notice that this question is in the difficulty level of easy, because of its popularity in Freebase.

The result of this project shows that easy questions can be answered quickly by the most of the people who are interested in that particular topic while just some of them could answer hard questions.

6.1 Future work

There are always improvements that can be made to the program. Here we are listed a few of them:

- Generate questions and answers for all the topics in the quiz game
- Design an algorithm to find the relations between words in question and generate it automatically
- possibility to make an API for this application and publish on the Internet

Appendices

Appendix A

Deployment

This application has been developed in Java. To run the application Java, virtual machine is needed on your computer. We used Freebase API as a data source, so you need to be connected to the Internet.

Appendix B

Java Classes

The following Java classes have been developed in the program:

- Main Class: This is the primary class and starts point of application and also main interface of the application. In this form, you can choose category and difficulty. And the system gives u random question based on your options.
- clsQuestion: This class has question information such as question text, the right answer, and wrong answers. All other questions use object of this class to make a question.
- clsArtwork: This class has been designed to generate question: *"What is the name of this artwork?"*. It gets a list of artworks, choose a random artwork, find the image property of the artwork and if there is any, show the image in the interface. The answers are the name of four artworks.
- clsArtist: This class has been designed to generate question: *"Who is the artist of [artwork]?"*. It gets a list of artworks, choose a random artwork, find the artist property of the artwork and if there is any, get the artist of artwork as the correct answer with three other artists for wrong answers.
- clsVisualArtist: This class has been designed to generate question: *"Which one is not by [artist]?"*. It gets a list of artists, choose a random artist, find the artwork property of the artist and if there is any, get three artworks of artist as the wrong answers with a random artwork from another artist as correct answer.

- clsArtform: This class has been designed to generate question: *"Which one is not a [artform] by [artist]?"*. It gets a list of artists, choose a random artist,find the artform property of the artist and if there is any find the artwork property of the artist and if there is any, get three artworks of artist as the wrong answers with a random artwork from another artist as correct answer.

- clsTrack: This class has been designed to generate question: *"Which song was not recorded by [musicArtist]?"*. It gets a list of musicArtists, choose a random musicArtist, find the track property of the musicArtist and if there is any, get three tracks of musicArtist as the wrong answers with another track from another musicArtist as correct answer.

- clsAlbum: This class has been designed to generate question: *"Which album was not recorded by [musicArtist]?"*. It gets a list of musicArtists, choose a random musicArtist, find the album property of the musicArtist and if there is any, get three albums of musicArtist as the wrong answers with another album from another musicArtist as correct answer.

- clsParentGenre: This class has been designed to generate question: *"Which genre is not parenting genre for [genre]?"*. It gets a list of genres, choose a random genre, find the parent_genre property of the genre and if there is any, get three parent_genre of genre as the wrong answers with another parent_genre from another genre as correct answer.

- clsGenre: This class has been designed to generate question: *"Which singer is best known for [genre]?"*. It gets a list of genres, choose a random genre, find the artists property of the genre and if there is any, get the artist of artwork as the correct answer with three other artists for wrong answers.

- clsFootball: This class has been designed to generate question: *"Which country was the champion of [FIFA World Cup]"*. It gets a list of FIFA World Cups, choose a random FIFA World Cup, find the champion property of the FIFA World Cup and if there is any, get the champion team as the correct answer with three other champion teams of other FIFA World Cups as wrong answers.

- clsVolleyBall: This class has been designed to generate question: *"Which country was the champion of [sports_championship_event]"*, [sports_championship_event] which contains "Volleyball Men's World" in its name. It gets a list of Volleyball World Cups, choose a random Volleyball World Cup, find the champion property of the Volleyball World Cup and if there is any, get the champion team as the correct answer with three other champion teams of other Volleyball World Cups as wrong answers.

- clsBasketBall: This class has been designed to generate question: *"Which country was the champion of [sports_championship_event]"*, [sports_championship_event] which contains "FIBA World" in its name. It gets a list of FIBA World Cups, choose a random FIBA World Cup, find the champion property of the FIBA World Cup and if there is any, get the champion team as the correct answer with three other champion teams of other FIBA World Cups as wrong answers.

- clsTennis: This class has been designed to generate question: *"where is [tennisPlayer] from?"*. It gets a list of tennisPlayers, choose a random tennisPlayer, find the nationality property of the tennisPlayer and if there is any, get the nationality of tennisPlayer as the correct answer with the name of three other countries for wrong answers.

Appendix C

User Study

The List of all the questions in survey: (To find the answer of all the questions go to Page 50)

- Art:
  1. Who is the artist of The Last Supper?
     (a) Leonardo da Vinci
     (b) Haroldo Gonzlez
     (c) Michelangelo
     (d) Pablo Picasso
  2. What is the name of this artwork?

     (a) Cutting the Stone
     (b) Oath of the Tennis Court
     (c) Primavera
     (d) Women of Algiers

3. Who is the painter of La maja desnuda?
   (a) Donatello
   (b) Francisco Goya
   (c) Pablo Picasso
   (d) Francisco Pradilla
4. Which one is not an artwork by Ai Weiwei?
   (a) Circle of Animals / Zodiac Heads
   (b) Template
   (c) Coca-Cola vase
   (d) Still Life
5. Which one is not a painting by Caravaggio?
   (a) Salome with the Head of John the Baptist
   (b) Sleeping Cupid
   (c) Martha and Mary Magdalene
   (d) Death Dealer
6. What is the name of this artwork?

   (a) Annunciation
   (b) Irises
   (c) Amor Vincit Omnia
   (d) Grande Odalisque
7. Which one is not a painting by Eugne Delacroix?
   (a) The Death of Sardanapalus
   (b) Liberty Leading the People
   (c) Clorinda Rescues Olindo und Sophroni
   (d) The Fighting Temeraire
8. Which one is not a sculpture by Auguste Rodin?
   (a) Balzac in a Frock Coat

(b) Bust of George Wyndham
(c) Sans II
(d) Bust

9. Which one is not by Banksy?
   (a) Cardinal Sin
   (b) Balloon Girl
   (c) Bomb Hugger
   (d) Guardians of the Secret
10. Who is the artist of The Scream?
   (a) Lyn Ott
   (b) Edvard Munch
   (c) Bob Ross
   (d) Jeff Koons
11. Who is the artist of Mount Rushmore National Memorial ?
   (a) Gutzon Borglum
   (b) Francis Derwent Wood
   (c) Charles O. Perry
   (d) tefan Luchian
12. Which one is not a sculpture by Donatello?
   (a) David
   (b) Virgin and Child
   (c) Judith and Holofernes
   (d) Apollo Belvedere

- Music:
  1. Which song was not recorded by Pink Floyd?
     (a) Bring the Boys Back Home
     (b) Vera
     (c) I Will Survive
     (d) The Show Must Go On
  2. Which genre is not parenting genre for Pop music?
     (a) Rhythm and blues
     (b) Folk rock
     (c) Dance music
     (d) Rock music
  3. Which singer is best known for Rock music ?
     (a) The Police
     (b) Edward Elgar
     (c) Ani DiFranco
     (d) Charles Mingus
  4. Which Album was not recorded by Linkin Park?
     (a) Breaking the Habit
     (b) Bleed It Out
     (c) A Thousand Suns
     (d) Embracing the Sunshine
  5. Which Album was not recorded by Rihanna?
     (a) Talk That Talk
     (b) California King Bed
     (c) Gorilla
     (d) Unapologetic
  6. Which singer is best known for Bolero ?
     (a) David Miller
     (b) Chet Atkins
     (c) Alex Lifeson
     (d) Charles Mingus
  7. Which song was not recorded by Bruce Springsteen?
     (a) Secret Garden

(b) My City of Ruins
(c) Chicken Lips and Lizard Hips
(d) This Is Your Life

8. Which genre is not parenting genre for Punk rock?
   (a) Samba
   (b) Glam rock
   (c) Rockabilly
   (d) Mod
9. Which song was not recorded by Celine Dion?
   (a) The Reason
   (b) Rasputin
   (c) Here There & Everywhere
   (d) My Heart Will Go On
10. Which genre is not parenting genre for Alternative rock?
   (a) Heavy metal
   (b) Rock music
   (c) Soul music
   (d) Post-punk
11. Which singer is best known for Country ?
   (a) Frank Sinatra
   (b) Cold Chisel
   (c) Cowboy Slim Rinehart
   (d) Billy Bob Thornton
12. Which genre is not parenting genre for Heavy metal?
   (a) Blues rock
   (b) Hard rock
   (c) Emo
   (d) Trance music

- Sports:
  1. where is Andy Murray from?
     (a) Austria
     (b) Australia
     (c) United Kingdom
     (d) United States of America
  2. Which country was the champion of 1958 FIFA World Cup?
     (a) Brazil national football team
     (b) Mexico national football team
     (c) France national football team
     (d) Germany national football team
  3. Which country was the champion of 2002 FIVB Volleyball Men's World Championship?
     (a) Poland men's national volleyball team
     (b) France men's national volleyball team
     (c) Brazil men's national volleyball team
     (d) Serbia men's national volleyball team
  4. Which country was the champion of 2010 FIBA World Championship?
     (a) Puerto Rico men's national basketball team
     (b) United States men's national basketball team
     (c) China men's national basketball team
     (d) Croatia men's national basketball team
  5. Which country was the champion of 1986 FIFA World Cup?
     (a) Brazil national football team
     (b) Argentina national football team
     (c) Spain national football team
     (d) Germany national football team
  6. where is Lindsay Davenport from?
     (a) United States of America
     (b) Canada
     (c) Russia
     (d) Republic of Ireland

7. Which country was the champion of 2014 FIVB Volleyball Men's World Championship?
   (a) Russia men's national volleyball team
   (b) France men's national volleyball team
   (c) United States men's national volleyball team
   (d) Poland men's national volleyball team
8. where is Victoria Azarenka from?
   (a) Belarus
   (b) England
   (c) Republic of Ireland
   (d) Thailand
9. Which country was the champion of 1962 FIVB Volleyball Men's World Championship?
   (a) Italy men's national volleyball team
   (b) France men's national volleyball team
   (c) Netherlands men's national volleyball team
   (d) Soviet Union men's national volleyball team
10. Which country was the champion of 1994 FIFA World Cup?
    (a) Brazil national football team
    (b) England national football team
    (c) Italy national football team
    (d) France national football team
11. Which country was the champion of 1990 FIVB Volleyball Men's World Championship?
    (a) Serbia men's national volleyball team
    (b) Italy men's national volleyball team
    (c) Finland men's national volleyball team
    (d) Russia men's national volleyball team
12. Which country was the champion of 1970 FIFA World Cup?
    (a) Brazil national football team
    (b) Germany national football team
    (c) Mexico national football team
    (d) Netherlands national football team

Answers of above questions:

- Art:
  1. a
  2. c
  3. b
  4. d
  5. d
  6. a
  7. d
  8. c
  9. d
  10. b
  11. a
  12. d
- Music:
  1. c
  2. b
  3. a
  4. d
  5. c
  6. a
  7. d
  8. a
  9. b
  10. c
  11. c
  12. d
- Sports:
  1. c

2. a
3. c
4. b
5. b
6. a
7. d
8. a
9. d
10. a
11. b
12. a

Bibliography

[1] http://www.gumpaper.com

[2] http://www.freebase.com

[3] Xuchen Yao, Benjamin Van Durme, *Information Extraction over Structured Data: Question Answering with Freebase*, 2014: Johns Hopkins University, Baltimore, MD, USA.

[4] Xuchen Yao, *Lean Question Answering over Freebase from Scratch*, 2015: Johns Hopkins University Baltimore, MD, USA.

[5] Mohamed Yahya, Klaus Berberich, Maya Ramanath, Gerhard Weikum, *On the SPOT: Question Answering over Temporally Enhanced Structured Data*, 2013: Proc. of TAIA (in conjunction with SIGIR 2013)

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

[7] https://developers.google.com/freebase/guide/basic_concepts

[8] Xuchen Yao, Benjamin Van Durme, Jonathan Berant *Freebase QA: Information Extraction or Semantic Parsing?*, 2015: Johns Hopkins University, Stanford University, USA.

[9] Jonathan Berant, Andrew Chou, Roy Frostig, Percy Liang, *Semantic Parsing on Freebase from Question-Answer Pairs*, 2013: Computer Science Department, Stanford University

[10] A. Kalyanpur, B. K. Boguraev, S. Patwardhan, J. W. Murdock, A. Lally, C. Welty, J. M. Prager, B. Coppola, A. Fokoue-Nkoutche, L. Zhang, Y. Pan, Z. M. Qiu, *Structured data and inference in DeepQA*, 2012: IBM Journal of Research and Development