A TURING GAME FOR COMMONSENSE KNOWLEDGE EXTRACTION

BY

JUAN FERNANDO MANCILLA CACERES

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Adviser:

Professor Eyal Amir

Abstract

Commonsense is of primary interest to AI research since the inception of the field. Traditionally, commonsense knowledge is gathered by using humans to create and insert it in knowledge bases. Automating the collection of commonsense from text that is freely available can reduce the cost and effort of creating large knowledge bases and can enable systems that dynamically adapt to current relevant commonsense. In this thesis, we design, implement and evaluate an online game that classifies, with players' input, text extracted from the Web as commonsense knowledge, domain-specific knowledge or nonsense. We also create a knowledge base that includes commonsense facts in natural language and information on how common a given fact is. The game is currently released on the Web and on Facebook. It is open for play and under constant improvement. The creation of a continuous scale to classify commonsense helped during evaluation of the data by clearly identifying which knowledge is reliable and which needs further qualification. When comparing our results to other similar knowledge acquisition systems, our Turing Game performs better with respect to coverage/redundancy and reliability of the commonsense acquired.

To Papa, Mama, Carlitos, Tatiana, Pablo and Sara.

Acknowledgments

I would like to thank everybody who helped me and supported me throughout the process of creating this thesis. First of all, I would like to thank my advisor, Eyal Amir, for believing in me and giving me the opportunity to do research in this exciting topic on AI. I would also like to thank Fulbright(LASPAU) for supporting me during my two years of Master's studies at UIUC. I would like to give special thanks to my whole family and to Sara Estrada, who have always been present and available to provide support and valuable advice. Last, but not least, I want to thank all my graduate colleagues, professors and friends who collaborated with me on this project through discussions and exchanges of ideas: Karrie Karahalios, David Amores, Luis Mijangos, Leonardo Bobadilla, Oscar Sanchez, the KRR Group and the SigArt group of UIUC, and all those who played the Turing Game.

Table of Contents

Chapter 1

Introduction

The study of commonsensical agents is required for the development of human-level AI, and without it, there is no possibility for an artificial agent to be able to exist and be successful in the real world [Amir, 2009]. To be able to create an artificial commonsensical agent (i.e., an agent that is able to be cooperative, adaptive and capable of engaging in everyday life and reasoning), two kinds of commonsense must be studied: commonsense reasoning and commonsense knowledge.

Although commonsense reasoning is beyond the scope of this thesis, it is necessary to realize that in order to correctly apply any reasoning method, a vast amount of information about the world is needed [McCarthy, 1968]. This kind of knowledge about the world, such as facts about events, desires and object properties, is what we call commonsense knowledge.

Because of the nature of commonsense knowledge, creating a knowledge base with it is not an easy task. This is due to the fact that commonsense is dynamic and dependant on the context, which includes culture and other specific circumstances of the people expressing the commonsense [Valsiner, 1985]. This makes it impossible to generate commonsense randomly and to verify it automatically at the same time.

These considerations imply that humans are needed in order to either collect the commonsense knowledge or to verify it. Previous methods include hiring people to introduce facts into a knowledge base (See [Lenat et al., 1990]), or using volunteers to enter the data in natural language (See [Singh et al., 2002]). The first approach is expensive, and the use of volunteers results in unreliable and sporadic data.

These challenges point towards the solution of gathering commonsense knowledge via an automated process and, because the amount of commonsense is too large for a group of people to enter it manually (and continually), the Web is an excellent candidate as a source of this type of knowledge.

Humans are the standard bearers of commonsense [Elio, 2002]. Therefore, the evaluation and categorization of commonsense must agree with what humans consider commonsense. This framework entails the difficulty that commonsense may vary according to the group of people verifying it.

In order to deal with this difficulty, we will distinguish between two types of knowledge that any person has: *commonsense knowledge* and *domain-specific knowledge*. Domain-specific knowledge includes that which is relevant to a specific group about a specific subject (the carrier of such knowledge is regarded as an expert). Commonsense knowledge is what is known by everybody everywhere, regardless of group memberships.

Techniques to obtain parsed sentences from the web already exist, but none of them differentiates between domain-specific knowledge and commonsense knowledge as previously defined. Also, although these methods usually have high accuracy, they still generate nonsensical sentences that must be excluded from the knowledge base being created [Banko and Etzioni, 2007]. Therefore, the automation of the acquisition of commonsense knowledge must include a way of evaluating the knowledge so that it matches what humans regard as commonsense.

The fact that we need people to evaluate the data means that we need a way to encourage collaboration. As a way to motivate people to collaborate, in this work we introduce a game that will draft users into verifying if a sentence extracted automatically from the web is commonsense or not.

Specifically, we present the design and implementation of a game called **The Turing Game** that takes text extracted automatically from the Web and uses players' input to make distinctions among commonsense knowledge (most players know if the fact expressed by the sentence is either true of false), domain-specific knowledge (only a restricted amount of players know about the fact), and sentences without meaning (nonsense produced probably by an error of the parser used to extract the sentence). It also reports if more information about a given fact is needed in order to correctly classify it.

Although the Turing Game's design does not constrain the source of the original data, we are currently obtaining it from *Simple Wikipedia* [SigArt, 2009]. *Simple Wikipedia* is a smaller version of Wikipedia with 60,004 articles that uses simple English words and grammar oriented to people that are learning English [Wikipedia, 2010a].

Simple Wikipedia was chosen due to its accessibility and broad coverage of many topics about the world. In addition, some policies regarding the content of Wikipedia make it useful for commonsense. As an example of such policies, original research is not supposed to be part of any article. That policy is desirable for us because original research would clearly not be commonsense. Also, the final content of an article is usually achieved after several iterations of contributions and corrections from several editors, which reflects the agreement of different people about the specific subject [Wikipedia, 2010b]. This agreement (either tacit or explicit) among people has also been used as a definition of commonsense by [Custódio and Pinto-Ferreira, 1999].

A consequence of our definition of commonsense knowledge is that we cannot classify all facts as either commonsense or not, since not all commonsense is equally common (not all agreement is complete). Therefore, we propose the use of a continuous scale that will allow us to handle commonsense knowledge with greater precision and will also provide a way to report a degree of confidence in our results. Another advantage of this scale is that it allows us to clearly identify which knowledge needs revision or more qualification.

To evaluate the data received from the game, we will assume that each contribution and each player are independent of the others. This, together with the restricted set of actions and communication between players, allows us to do a hypothesis test for each sentence that has been verified in the game. The results indicate whether a sentence expresses commonsense or not, or whether more data is needed to reach a conclusion with a given confidence level. To increase the flexibility of our results, the confidence level used as threshold is a parameter that is selected according to the intended usage or application of the knowledge.

The commonsense knowledge acquired by the game is stored in natural language which, although ambiguous, makes the knowledge base readable by users and keeps the subtleties that make the fact commonsense. Working with natural language as a way to represent knowledge is a current line of research of interest to other fields such as Human-Computer Interaction and Natural Language Processing. Therefore, in this thesis we do not address the ambiguity of the language or its usage for formal reasoning.

Using games as a way to encourage collaboration from humans can be regarded as a type of human-based computation [Shahaf and Amir, 2007], in which the incentive for participation comes from the participant's desire to be entertained. The main difficulty of this approach lies in the fact that, in order for the game to work, it needs to be fun for people to play and at the same time supply the correct information to solve the problem at hand.

The combination of these two characteristics makes the design of the game much like designing an algorithm [von Ahn and Dabbish, 2008]. That is, the whole design must be focused on the correctness of the data and the features that increase the replay value of the game. In our case, correctness is ensured through the scoring policy and the limited communication among players; while fun is promoted through a list of high scores (including all players or including only friends), which provides a sense of competition. Players' feedback suggested that some of the enjoyment coming from the game was due to the intellectual stimulation obtained from verifying the sentences, although players also reported that the game lacks a definite end, which reduces the motivation to play.

The game was released on Facebook and on a University Website and is currently open to players[1]. Within a period of five weeks, more than 150 people played the game and more than 3,000 sentences were evaluated.

[1] http://commonsense.cs.illinois.edu/turing_game/index.php,
http://apps.facebook.com/turingrpg

Chapter 2

Related Work

The use of a continuous scale to reason about commonsense knowledge has been explored before in the context of logic. In [Custódio and Pinto-Ferreira, 1999] and [Druzdzel, 1993], the authors created logical systems that use either agreement of individuals or probabilistic models to reason about commonsense. Another probabilistic reasoner for commonsense is described in [Neufeld, 1990], but to the best of our knowledge no commonsense knowledge base with a continuous scale has been created. Although none of the previous papers discuss how to gather commonsense, they show the importance and relevance of using a continuous scale for commonsense.

Combining the computational power of machines and humans has been addressed and formalized before in [Shahaf and Amir, 2007]. According to the authors' description, our scenario fits the model where the player is considered a probabilistic oracle with utilities. In this context, our game can be considered as an interface for the players to act as *oracles* that may produce a correct answer with a given probability and expect some kind of reward. We modeled the player as probabilistic in order to be able to quantify the confidence we have in our results (and to consider the possibility of a player with malicious or random behavior). The reward to the player is the entertainment experienced while playing our game. This is important in order to encourage participation and efficiency in our system.

2.1 Gathering Commonsense Knowledge

Current approaches for gathering commonsense knowledge involve either using humans to insert the data or mining the knowledge automatically. The best known two approaches that use humans are Cyc [Lenat et al., 1990] and OpenMind [Singh et al., 2002]. Cyc uses experts to code knowledge in a specific unambiguous language whereas OpenMind uses collaborators. Although these two approaches have been relatively successful, Cyc is often regarded as difficult to use while OpenMind lacks rewards for its collaborators (which limits the number of volunteers). On the other hand, [Matuszek et al., 2005] uses machine learning and Cyc to gather knowledge automatically from the Web. All these approaches tend to classify all commonsense knowledge as equally common. We address these difficulties through our continuous scale and the use of the game as motivation for our collaborators.

Another effort to collect common knowledge from contributors is LEARNER2 [Chklovski and Gil, 2005]. It was deployed for 3 years as an interactive kiosk at a science museum. It collected data about meronymy (*part-of*) relations. Knowledge was acquired using templates in the form of fill-in-the-blank. The main difference with our approach is the way contributors were encouraged to participate and the kind of knowledge collected. Also, our system improves on LEARNER2 by adding the possibility of directing the users to improve the reliability of certain data and by explicitly stating which data needs to be improved and which has already enough information to be classified correctly.

In [Schubert and Tong, 2003], the authors deal with extracting general world knowledge out of an annotated corpus, and subsequently evaluating their results with the help of judges. In their work, the authors show how judges are supposed to evaluate the data by indicating if a sentence make sense, if it has missing parts, if it doesn't make sense at all, and other similar questions. The instructions are somewhat complicated and, in fact, they reported low agreement among the judges. This problem is addressed in our game by making the rules accessible and by explicitly looking for the majority vote of the players.

2.2 Similar Games

Regarding the use of games to encourage contribution, von Ahn and his colleagues (See [von Ahn and Dabbish, 2008]) have developed what they call games with a purpose (GWAPs). These games focus on gathering data to solve hard problems like image tagging and text digitalization, among others. In their work, the authors describe three different kinds of games. The first kind is called output-agreement games, where two random players have to agree on the output (e.g. the ESP game [von Ahn and Dabbish, 2004] to annotate images where the input is an image and the output is a word). The second kind of game is the inversion-problem, where one player gets an input and gives clues to the other player, the latter must guess the input of the game (e.g. *Verbosity* [von Ahn et al., 2006], where the player gives common characteristics of a word and the other player must guess the word). The third type is the input-agreement kind. In this game, both players are given an input and they must guess if they both have the same input. By design, these games are different to the Turing Game.

The authors also describe how some design patterns have been useful to increase the enjoyment on all these games. Some of these characteristics are timed response, score keeping, player skill levels, high score list, etc. Our game incorporates some of these, but it is important to notice that our game does not try to gather information (as is the case for GWAPs) rather it evaluates previously collected data. In this sense our game does not fit any of the three kinds of games described previously but it has a new pattern in which a player judges the performance of the other player as correct.

In the ESP game, two players observe the same image and they must guess the word that the other person is thinking. Given that the only information they share is the image, the players are likely to introduce words that describe the image somehow. These words get attached to the image as labels.

Verbosity is an example of a game that collects commonsense. It is based on the game Taboo^{TM}. In this game, two players interact by filling templates that may be considered commonsense. It is an inverse-problem game in which one of the players gets an input in the form of a word and by filling templates (such as: ”It contains” or ”It is used for”) it gives clues to the other player so that he can guess the input word. Also, as with many other applications of commonsense, the data is considered commonsense or not in a binary decision, while in reality some facts may be more common than others. Although our game and *Verbosity* are related in the fact that both deal with commonsense knowledge, the main difference is that ours has the purpose of evaluating commonsense knowledge. In this sense we might as well see these games as complementary since the data collected by *Verbosity* could be evaluated by the Turing Game.

Common Consensus [Lieberman et al., 2007] is another game for commonsense knowledge acquisition. The game is based on the TV game show *Family Feud* and the data is restricted to knowledge about how to accomplish goals. It asks two participants about what is required to achieve a goal and the players may enter any amount of words. The more matches the players have at the end of the game, the more points they get. This game tries to solve the problem of people not contributing to OpenMind after a while because of a lack of reward for helping the project.

Chapter 3

The Turing Game

3.1 Design

To understand the ideas behind the design of the Turing Game we must observe it from two different perspectives: knowledge acquisition and the Turing Test. The classical approach to knowledge acquisition is to have people enter the data directly into a database [Singh et al., 2002]. Some of the current approaches try to automate the process by using learning techniques [Matuszek et al., 2005] or by encouraging people to collaborate, for example with games [von Ahn et al., 2006].

One of the weaknesses of these approaches is that they may produce unreliable data, and also that they classify knowledge as either commonsense or not. Verification of the data still needs to be done by humans, which in turn becomes expensive. Our proposed solution is to gather the knowledge automatically and use humans, who collaborate out of their desire of entertainment, only to verify the data.

Due to the unstructured nature of commonsense knowledge, games for collecting commonsense are hard to setup without the use of free-text (input text in natural language). Nevertheless, free-text games are easy to sabotage and therefore produce noisy data[1]. To overcome this difficulty, players in our game will have no communication among them and will only **classify** sentences in natural language.

Following the idea that the design of the game is much like the design of an algorithm, Figure 3.1 shows the input-output design of the game. The initial input information for the game comes from an off-the-shelf parser [SigArt, 2009] which extracts a sentence from an article in *Simple Wikipedia* and, together with the action of the user, the system produces as output an update to the knowledge base.

A naive implementation of the game would have a single user classifying sentences as commonsense or not. This implementation has clear difficulties. First, it would be impossible to evaluate the answers of the player as correct or incorrect making it impossible to reward the player with any score (since the game doesn't know what is commonsense and what is not). Also, the game

[1] When users are allowed to enter text in natural language, it is relatively easy for players to find creative ways to cheat and improve their score while not providing the correct information.

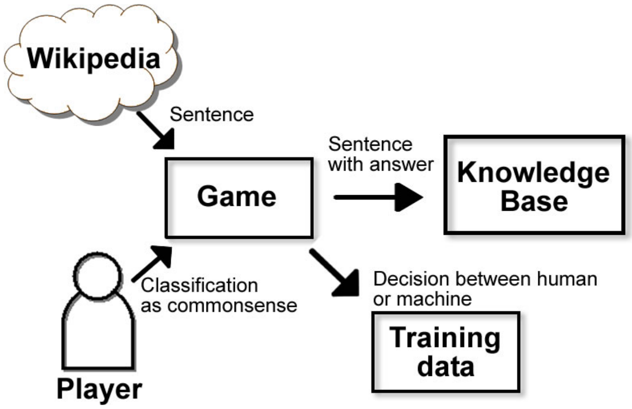


Figure 3.1: The input to the game is a sentence extracted from Wikipedia and the interaction of the player. The output is a knowledge base with information about each sentence commonsensical nature. Training data to distinguish between human and machine is also collected.

would be easily ”fooled” by a player answering randomly. This scenario would not produce the correct data and would be marginally fun.

Another more classical scenario (as shown in [von Ahn and Dabbish, 2004]) would have several players evaluating the same sentence and accepting the input only if they agree amongst each other. In principle, this approach would accurately mark a sentence as commonsense or not but, for our case, it is still not appropriate. The reason is that it would be easy for a group of players to agree on a strategy by entering the same answer (everybody marking the sentence as commonsense for example) which in turn would create noisy data.

The basic problem is that a yes/no question is not appropriate because an agent playing randomly can easily fool any system. Therefore we need to increase the set of possible answers so that it is harder to play randomly and still have a high score. Due to its nature, the input text may be classified in four different categories (See Figure 3.2). If the parser extracted an incomplete sentence or any other nonsensical data, the sentence in itself is nonsense, if the sentence was extracted correctly the content may either be known (regardless of the fact of actually being true or false), or unknown.

Therefore, the natural options to present to the player are four: True, False, Unknown, and Nonsense. These options are to be selected depending on whether the player knows if a sentence is true or not, if he/she doesn't know if the fact is true or if he/she is incapable of understanding the sentence.

Although this already improves the reliability of the data, only having multiple human players agree on one of those four answers still presents problems. Basically, they still can agree on a specific strategy and maximize their score while providing noisy data. Here is where the inspiration from the Turing Test comes into play.

In [Turing, 1950], Alan Turing proposed a famous test to demonstrate if

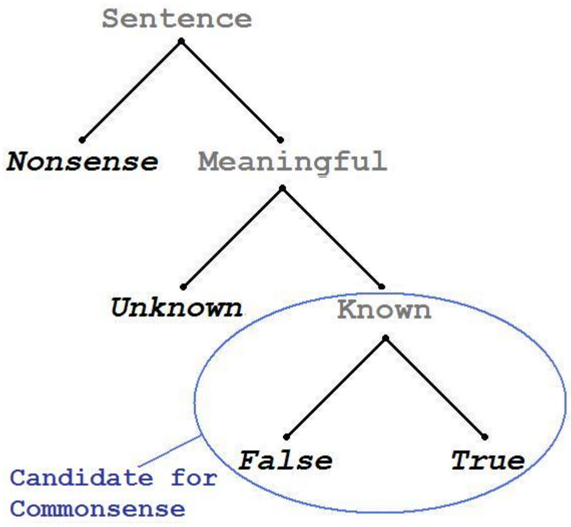


Figure 3.2: Possible values that a sentence may take. *Nonsense* if the parser made an error, *Unknown* if it is domain-specific or *Known* (either *True* or *False*) if it is commonsense. These values are the possible four answers that a player may give to any specific sentence.

a machine was intelligent or not. Although this test has been criticized as a correct test of intelligence, it is the source of inspiration for the scenario that is created in our game. In the test, Turing proposes a three agent scenario where there are two humans and one machine. The role of one human (the judge) is to determine which of the other two agents is the machine by virtue of their conversation skills.

Since only humans are considered to have commonsense, the Turing Test provides a natural situation for humans to easily determine the identity of a computer which pretends to have commonsense. Using this as an inspiration, we devise a three player game where the purpose of the player is to distinguish between another human and a machine. Because of our definition of commonsense, we assume that both humans will agree (give the same answer) on a given sentence while the machine will guess its answer, making its identity obvious. This solves the problem outlined before: now the two humans cannot agree on any strategy because the robot might as well use such strategy. Also, the only communication between players is through their answers and therefore it would be difficult to distinguish the human by any other means than evaluating their answers.

Figure 3.3 shows the scenarios described above. Notice that the design of the game as a Turing Test is crucial to guarantee the correctness of the data.

Another important characteristic of the game is that the other human is also playing, therefore, if the answer of the player does not follow commonsense, the other human might erroneously identify the player as a machine, which results in a penalization on the player's score. Points are only awarded when recognizing

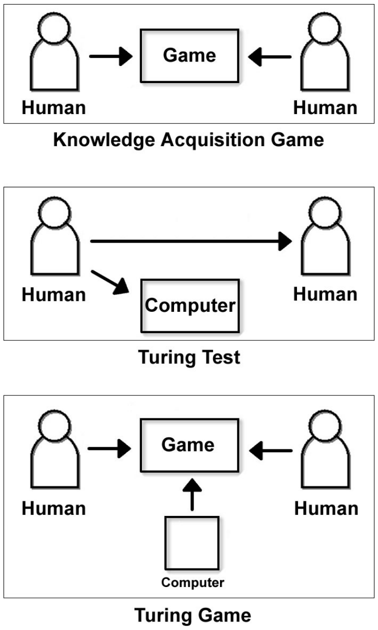


Figure 3.3: Scenarios of a knowledge acquisition game (top image), the Turing Test (middle image) and the Turing Game (bottom image).

the machine, not when evaluating the commonsense. This guarantees that the players will use their commonsense to classify the sentence (in order to avoid being penalized in the next step). Also, when awarding points, the information about the truth value of the sentence is not modified so even if the human is falsely identified, the data is not polluted.

Context is important for commonsense knowledge. Evidence discussed in [Elio, 2002] shows that people can fail to give correct answers on formal reasoning problems but give a correct one if the same problem is formulated with a certain contextualization (e.g. as a social behavior instance). This proves that, in the mind of people, the context validates some knowledge or gives priority to some ideas. In [McCarthy, 1990], the author proposed that in order to formalize commonsense, it is necessary to consider context as an object, in order to match the human ability to consider context explicitly. He proposes a formula *Holds(p,c)* to assert that the proposition p holds in context c.

Using this idea, the appropriate task for the player to solve is to answer that formula (expressed in natural language):

```
Is fact p true in the context c?
```

In our case, context is handled by the name of the Wikipedia article used as source for the sentence. For example, the sentence ”It means measure the land” comes from the article ”Geometry”. This would be presented in the game as follows:

```
Is the following fact about Geometry true?
```

```
It means measure the land.
```

In the initial testing phase, there was some concern regarding uninstantiated sentences such as: ”It is sold at stores”. The way that we used the source article as explicit context addresses this problem since it helps instantiating the sentence. For example, the previous sentence becomes commonsense when provided the fact that it refers to ”Shampoo”.

3.2 The Game

An instance of the game is defined as follows:

- First the player gets to choose a topic (which matches possible Wikipedia articles from where a sentence might be retrieved).

- Once the player chooses a topic, a sentence is randomly selected from the article with such title. Also, the system randomly chooses to return a new sentence (one that has not been seen by any player before) or a sentence that has been verified before. This is to balance between coverage and reliability by increasing the size of the knowledge base or increasing the amount of people that verify each sentence.

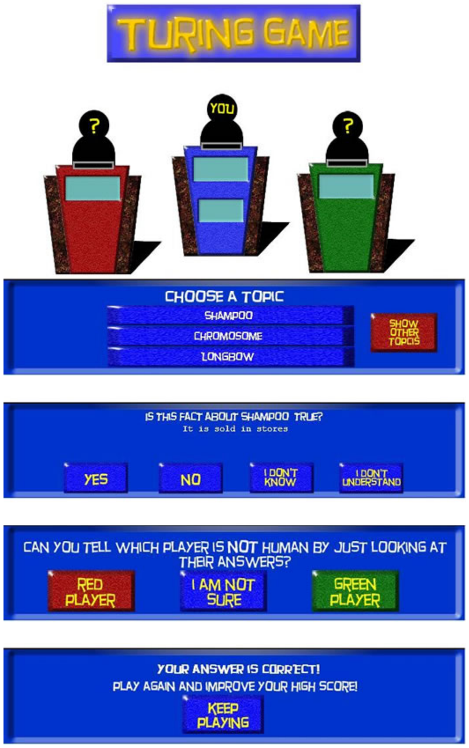
- The player is asked if the fact expressed by the sentence is true in the context of the article. The player has four possible answers: *”yes”*, *”no”*, *”I don’t know”* and *”I don’t understand”*, which respectively correspond to true, false[2], unknown and nonsense as described previously.

- Once the player answers, he is shown the answer of the other two players and he/she gets to identify which of the two is the machine that is answering randomly. The other answer comes from previously recorded games that has been provided by a human. If the sentence is new, the answer provided is *”I don’t know”*.

- If it is impossible to distinguish between the two answers, the player has the option to pass and avoid making a decision. No points are lost if this choice is made. If the player erroneously identifies the human as the machine, points are lost, if the player correctly identifies the machine, points are awarded.

- If the player provides an answer that apparently does not have commonsense, he may be penalized. This feature is implemented by learning how previous humans have correctly distinguished the machine. For example, if two players agree on a fact being true, and the third one answers nonsense, this indicates that the third player is the machine.

[2] Since data comes from Wikipedia, it is unlikely that a sentence will be false but to keep the possibility open for data to come from a different source, the option was kept.



IS THIS FACT ABOUT SHAMPOO TRUE?
It is sold in stores

CAN YOU TELL WHICH PLAYER IS NOT HUMAN BY JUST LOOKING AT THEIR ANSWERS?

YOUR ANSWER IS CORRECT!
PLAY AGAIN AND IMPROVE YOUR HIGH SCORE!

Figure 3.4: Snapshot of the game. The main interface is shown at the top along with the four possibilities of interaction with the player.

- Once the player has made a choice, he/she gets the opportunity to play again.

Figure 3.4 shows snapshots of the game showing all the stages of the game. The top part is an example of the interface and the rest show how the input of the player is collected.

3.3 Knowledge Base

After each instance of the game we obtain two outputs:

- Output 1: The sentence accompanied with the answer provided by the player
- Output 2: The decision of the player regarding the identity of the machine.

The knowledge base is implemented with a database that contains a unique id of the sentence, the complete sentence, the context (in this case, the source article) and 68 counters. These counters are divided into four groups according to the locale of the user (possible options are North America, Latin America, Europe, Other) and each of those groups is further divided into four subgroups according to the age of the player (children: younger than 15 years old, youth:

Sentence	Context	T	F	U	N
Toplessness being is	Toplessness	0	0	0	12
Gandalf Pippin go	The Lord of the Rings	0	0	0	8
trends are emerging to create hybrid models	Abiogenesis	1	0	2	1
Beaufou is a commune	Beaufou	3	0	3	0
Salmonella is a genus of bacteria	Salmonella	2	0	0	0
She was born on February	Regina Spektor	2	0	0	0
It means measure the land	Geometry	5	1	3	1
Frozen rain can be ice	Water	2	0	1	0
A werewolf is a mythical monster	Werewolf	4	0	0	0
It is sold in stores	Shampoo	10	0	0	0

Table 3.1: Example of the knowledge base. Each sentence is stored with a context (its article) and with counters that correspond to the times players have identified such sentence as true (T), false (F), unknown (U) or nonsense (N).

between 15 and 23, young adults: between 23 and 33, adults: older than 33). Each subgroup in turn has a counter for True, False, Unknown and Nonsense. These categories were chosen in the spirit of having a uniform distribution of the players. Four other counters are needed for the users of the game through the University Website which do not provide personal information, and also for some of the users of Facebook who choose not to share it. The information is stored in this way in order to enable an analysis of the distribution of the players. This is important to guarantee that the knowledge collected is in fact commonsense.

Output 1 is used to update the knowledge base by increasing the counter of the respective sentence. Output 2 is stored for training data to learn how to distinguish between human and machine given that we know our own answer.

Table 3.1 shows some of the sentences as stored in the knowledge base together with the counters obtained for each of them. For simplicity we do not show the 68 counters but the sum of all the counters that classify the sentence as true (T), false (F), unknown (U) or nonsense (N).

3.4 Correctness and Efficiency

The correctness of the data is due to the scoring policy and limited communication among the players. All players are anonymous and the only way to distinguish between a human and a machine is through their answers. Therefore, in order to maximize the score, each player will answer with as much commonsense as possible or he would face the risk of being misclassified as the machine. In the case of ill-intentioned players, or players that are simply answering randomly, we regard such answers as noise. The effect of that noise will be minimized according to the amount of people that have evaluated the sentence. Therefore, the more a sentence is evaluated, the more confidence we have about each individual answer. In Chapter 4, we will do an statistical analysis of the data and show the need of having each sentence being evaluated several times, and the confidence that we have on each sentence. This confidence will then be converted to a scale of commonsense which expresses how common is a fact expressed by a sentence.

The effectiveness of the game is directly related to how much the players enjoy the game and how often they play it. This allows us to process a larger amount of information. The performance of the game is evaluated and compared to similar approaches in Chapter 5.

The data obtained from the game will reflect if a sentence is commonsense or only domain-specific dependant on the diversity of the users. If users come from only one area, it may be possible that the knowledge is biased. Actually, the knowledge will be biased by the simple fact that currently it is only shown in English. Nevertheless, a simple analysis of the data shows that users come from a diverse background (considering age range and locale), which gives some confidence about the fact that the knowledge extracted is commonsense and not domain-specific.

Chapter 4

Commonsense in the Knowledge Base

As previously noted, the nature of commonsense knowledge and domain-specific knowledge is such that a specific world fact cannot be strictly classified as only one of those two options.

Also, because we are using anonymous contributors from the Web, it is crucial to consider the possibility of malicious players, automated robots, or simply players that did not understand the instructions. In a basic scenario, a simple majority vote can deal with the presence of these kind of players [Shahaf and Amir, 2007] but in our case, this is not enough. The main reason is that if a sentence has been evaluated by few players, we have much less confidence about the result than if a sentence has been evaluated by a significant amount of players.

Therefore, we create a scale of commonsense that describes how common a specific world-fact is. The values assigned to each sentence in the scale need to be proportional to the ratio of people who know the given fact and it must also contain information about the confidence of such ratio.

To do this, we construct the scale with information of the p-value from a hypothesis test on each sentence. This value depends on the number of people that reported that they actually know if the fact is true in comparison to the total number of people that have evaluated such fact.

As explained in a previous chapter, each sentence in the knowledge base is accompanied by 68 counters, which are increased according to the locale, age and answer of the player. Let us first define four quantities $t_{count}(s)$, $f_{count}(s)$, $u_{count}(s)$ and $n_{count}(s)$, which hold the total number of times that a sentence s has been classified as true, false, unknown and nonsense respectively:

$$\begin{aligned}
t_{count}(s) &= \sum_{i,j} t_{ij}(s) \\
f_{count}(s) &= \sum_{i,j} f_{ij}(s) \\
u_{count}(s) &= \sum_{i,j} u_{ij}(s) \\
n_{count}(s) &= \sum_{i,j} n_{ij}(s)
\end{aligned} \tag{4.1}$$

Where i and j stand for all possible locales and all possible ages.

For each sentence s, we will analyze it in two ways. First we will analyze if it is meaningful or not (if it makes sense) and then we will analyze if the knowledge expressed by s is common. If a sentence is both meaningful and common, it will be classified as commonsense. If it is meaningful but generally unknown, it will be classified as domain-specific.

We begin by creating a scale that represents how meaningful a sentence s is, and then we will also create a scale to represent how common s is. At the end of the chapter we will combine these two scales in order to make a final judgment regarding the commonsensical nature of the fact expressed in the sentence.

4.1 Identifying a Sentence as Meaningful or Nonsense

We begin by analyzing if a sentence has some meaning attached to it (i.e., it is not nonsense or an incomplete sentence).

Definition 4.1.1 Let $P_\sigma(s)$ be the ratio of people that have answered true, false and unknown (i.e., they understood the sentence regardless of its truth value) over the total number of instances the sentence s has been verified, $m(s)$.

$$\begin{array}{c} m(s) = t_{count}(s) + f_{count}(s) + u_{count}(s) + n_{count}(s) \\ P_\sigma(s) = \frac{t_{count}(s)+f_{count}(s)+u_{count}(s)}{m(s)} \end{array} \tag{4.2}$$

Assuming that each instance of the game is independent, we can consider each of them as a Bernoulli trial. Then, $P_\sigma(s)$ is an estimator of the real proportion of people that would understand such sentence and it has a binomial distribution. Because each sentence was evaluated at maximum 15 times, we cannot approximate the Binomial Distribution with a Normal Distribution and must perform a Binomial Hypothesis Test.

If all players played the game according to the rules, we assume that they will agree on the fact that a sentence is meaningful or nonsense. Therefore, our null hypothesis is that all players are playing randomly and therefore the ratio of people classifying the sentence as nonsense should be 0.5. The alternative hypothesis will be that the ratio is different to 0.5 which means that we will do a two-tail test. We choose this as our alternative hypothesis because rejecting the null hypothesis gives us enough information to determine if the ratio of people was significantly above 0.5 (which would imply that most people believe the sentence is meaningful) or significantly below 0.5 (which would indicate that the sentence is probably nonsense).

$$\begin{array}{c} H_0 : P_\sigma(s) = 0.5 \\ H_1 : P_\sigma(s) \neq 0.5 \end{array} \tag{4.3}$$

If we fail to reject the null hypothesis, we must conclude that we don't have enough information to make the claim that the sentence is meaningful or nonsense, in other words, we must conclude that we need more people to evaluate the sentence s in order to be able classify it. This will happen whenever the sentence has not been evaluated by enough people, or when the sentence is specially ambiguous and needs refinement.

Assuming that all players "tossed a coin" in order to decide if the sentence was nonsense or not, we would expect $n_{count}(s) = \frac{m(s)}{2}$. Therefore, the *effect size* we want to detect is the difference between the actual number of times the sentences has been marked as nonsense and the expected one.

Definition 4.1.2 Let $e_n(s)$ be the *effect size* of the Binomial Hypothesis Test which we want to detect.

$$e_n(s) = \left| n_{count}(s) - \frac{m(s)}{2} \right|. \tag{4.4}$$

We are interested in obtaining the p-value $p_n(s)$ of this test. That is, we want the probability of observing the current counters given that we suspect that the players were acting randomly. This means that the lower the value of $p_n(s)$ the more confident we are about the value of the counters. The p-value is obtained by calculating the probability of having an $n_{count}(s)$ that differs from $\frac{m(s)}{2}$ by at least e_n considering a binomial distribution with parameters $m(s)$ and 0.5.

Definition 4.1.3 Let $p_n(s)$ be the p-value of the Binomial Hypothesis Test. It is defined as the probability of observing a difference in the value of a random variable of at least the size of the effect size $e_n(s)$.

$$p_n(s) = P\left(X < \frac{m(s)}{2} - e_n(s)\right) + P\left(X > \frac{m(s)}{2} + e_n(s)\right) \tag{4.5}$$

where

$$\begin{aligned} P(X < x) &= \sum_{i=0}^{x} \tbinom{m(s)}{i} 0.5^i 0.5^{m(s)-i} \\ P(X > x) &= \sum_{i=x}^{m(s)} \tbinom{m(s)}{i} 0.5^i 0.5^{m(s)-i} \end{aligned} \tag{4.6}$$

Notice that $p_n(s)$ is **not** the probability of s being meaningful, it is the probability of the value of the counters being produced by players acting randomly. A better estimation of that probability of s being meaningful is $P_\sigma(s)$, $p_n(s)$ provides a way to quantify the confidence about $P_\sigma(s)$.

Table 4.1 and Table 4.2 show some sentences with their corresponding $P_\sigma(s)$ and $p_n(s)$. Notice that $P_\sigma(s)$ is equal to 1 in both the cases where all players evaluated the sentence as meaningful regardless of the fact that only one person evaluated sentence *1* and six evaluated sentence *2*. That is, a lot of the information is lost if $P_\sigma(s)$ is used to classify the sentence as meaningful, whereas $p_n(s)$ includes the information about how many people have verified s. Notice the case for sentence *1*, the p-value equals 1 which means that we have not

Sentence Id	Sentence	Article
1	*"People are known acting in comedies are comedians"*	Comedy
2	*"Computers can use many bits"*	Computer
3	*"For example, some languages (e.g. Chinese, Indonesian)"*	Verb

Table 4.1: Examples of sentences played on the game. The sentence Id is used for reference in this chapter. See Table 4.2 for relevant information about these sentences.

Sentence Id	Times played	$P_\sigma(s)$	p-value	Degree assigned by scale	Meaningful
1	1	1	1	0.5	Unknown
2	6	1	0.0313	0.9844	**Yes**
3	6	0.1667	0.0313	0.0156	**No**

Table 4.2: $P_\sigma(s)$ is the proportion of people that didn't answer *nonsense*. The p-value, $p_n(s)$, corresponds to the one obtained by the Binomial Hypothesis Test. The **degree assigned by our scale**, $\pi_s(s)$, represents the confidence that we have when classifying the sentence as meaningful. The last column is the decision made regarding the meaning of the sentence with a significance of 0.1

enough information to identify the sentence as meaningful, i.e., we cannot guarantee that the count observed was not produced by a player playing randomly, whereas for sentence *2* the p-value is 0.0313 which means that it is very unlikely for the counters to take those values if the players were playing randomly.

On the other hand, an obvious difficulty with using only p_n as a way to classify s, is that it is not proportional to the real probability of s being meaningful, that is, by only looking at the p-value, $p_n(s)$, we cannot distinguish between sentence *2* and sentence *3*, although by looking at $P_\sigma(s)$ we can clearly tell that one is very likely nonsense while the other one is not. Therefore, it is desirable to assign a number to s that tells us, just by looking at it, information about the ratio of people that classified the sentence as nonsense (proportional to $P_\sigma(s)$), that also includes the information about p_n.

With this in mind we define $\pi_s(s)$, which is a degree of how much *sense* a sentence has, including all the properties mentioned above. It allows us to easily classify sentences as meaningful or nonsense and it tells us if we need more information about s in order to be able to make a claim. Table 4.2 shows each of the sentences with its corresponding $\pi_s(s)$. Notice how we are now able to clearly differentiate *1* from *2* (*2* has a much higher value of $\pi_s(s)$) and *2* from *3* (sentence *2* is close to 1 while sentence *3* is close to 0).

Definition 4.1.4 Let $\pi_s(s)$ be the value that represents how much confidence there is on a sentence s being meaningful. $\pi_s(s) \in [0, 1]$, a value of 1 represents a lot of confidence, 0 represents none.

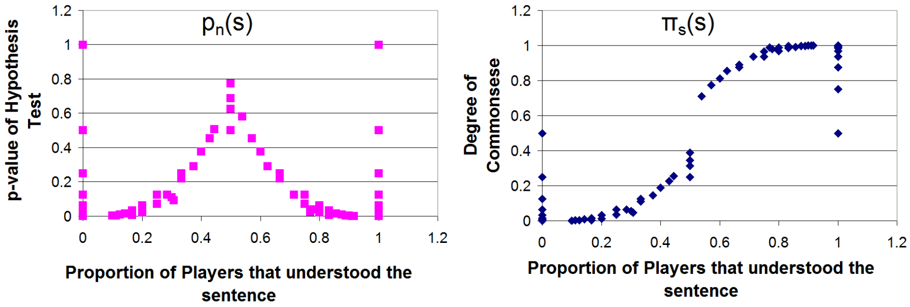


Figure 4.1: Left figure: The p-value of the Binomial Hypothesis Test, $p_n(s)$, with respect to the ratio of players that understood the sentence, $P_\sigma(s)$. Right figure: The value assigned by our scale, $\pi_s(s)$, with respect to $P_\sigma(s)$. Notice that our scale assigns a value to each sentence that is proportional to the proportion of players that understood the sentence while the p-value doesn't.

$$\pi_s(s) = \begin{cases} 1 - {}^{p_n(s)}/_2 & \text{if } P_\sigma(s) > 0.5 \\ {}^{p_n(s)}/_2 & \text{if } P_\sigma(s) \leq 0.5 \end{cases} \tag{4.7}$$

By dividing $p_n(s)$ by 2, we keep $\pi_s(s) \in [0, 1]$ and by inverting its value when $P_\sigma(s) > 0.5$, we get the proportionality we desired. If we want to classify the sentence as meaningful or not, we only need to define a threshold α to which we can compare $\pi_s(s)$. If $\pi_s(s) < \alpha$ we have a confidence of $1 - \alpha$ that the sentence is nonsense (this confidence is inherited from $p_n(s)$) and if $\pi_s(s) > 1 - \alpha$ then we have a confidence of $1 - \alpha$ that the sentence is meaningful. For all other cases we can only claim that we need more players to evaluate the sentence. Figure 4.1 shows $p_n(s)$ and $\pi_s(s)$ with respect to $P_\sigma(s)$.

4.2 Identifying a Sentence as Known or Unknown

In order to be able to fully classify a sentence as commonsense we still need to decide if a given sentence s is known or not by most of the players. Also, if a sentence has been determined to be meaningful as described in the previous section but it is unknown, then the sentence is probably domain-specific (only a selected group of people know about the fact expressed in the sentence). We will do a similar analysis to the one described previously but with the appropriate modifications.

First, we need to find the ratio of players that classify s as *known*. This amount will be represented as $P_\gamma(s)$ and it is the sum of all players that know the truth value of the sentences (either *true* or *false*) divided by all the players that understood the sentence (i.e., all players without counting those that marked the sentence as *nonsense*). This is equivalent to assuming that s is meaningful. $P_\gamma(s)$ is an estimation of the probability of s being commonsense given that it makes sense.

Definition 4.2.1 Let $P_\gamma(s)$ be the ratio of people that know the truth value of s over the total number of people that understood the sentence $k(s)$. Let $k(s) = m(s) - n_{count}(s)$.

$$P_\gamma(s) = \frac{t_{count}(s) + f_{count}(s)}{k(s)} \tag{4.8}$$

Using the same rationale as before, we do a Binomial Hypothesis Test to determine if a sentence s is known or not. Again, we do a two-tail test and the hypothesis are

$$\begin{aligned} H_0 &: P_\gamma(s) = 0.5 \\ H_1 &: P_\gamma(s) \neq 0.5 \end{aligned} \tag{4.9}$$

If all players were playing randomly, we would expect $P_\gamma(s) = \frac{k(s)}{2}$. Therefore,

Definition 4.2.2 Let $e_n(s)$ be the *effect size* of the Binomial Hypothesis Test which we want to detect.

$$e_c(s) = \left| \frac{k(s)}{2} - (t_{count}(s) + f_{count}(s)) \right| \tag{4.10}$$

We define $p_c(s)$ as the p-value of the test which is calculated in a similar fashion as $p_n(s)$.

Definition 4.2.3 Let $p_c(s)$ be the p-value of the Binomial Hypothesis Test. It is defined as the probability of observing a difference in the value of a random variable of at least the size of the effect size $e_c(s)$.

$$p_c(s) = P\left(X < \frac{k(s)}{2} - e_c(s)\right) + P\left(X > \frac{k(s)}{2} + e_c(s)\right) \tag{4.11}$$

Also, for the same reasons as before, we need a scale to represent the fact that a given sentence s is known. This scale should include the confidence of $p_c(s)$ and at the same time be proportional to $P_\gamma(s)$.

Definition 4.2.4 Let $\pi_c(s)$ be the value that represents how much confidence there is on a sentence s being known by most players. $\pi_c(s) \in [0, 1]$, a value of 1 represents a lot of confidence, 0 represents none.

$$\pi_c(s) = \begin{cases} 1 - {}^{p_c(s)}/_2 & \text{if } P_\gamma(s) > 0.5 \\ {}^{p_c(s)}/_2 & \text{if } P_\gamma(s) \leq 0.5 \end{cases} \tag{4.12}$$

where

Sentence Id	Sentence	Article
4	*”It is a county in the U.S. state of North Carolina”*	Anson County
5	*”the level experience is needed to level”*	Diablo II
6	*”Chess is a very complex game”*	Chess

Table 4.3: Examples of sentences played on the game. The sentence Id is used for reference in this chapter. See Table 4.4 for relevant information about these sentences.

Sentence Id	Times known	$P_\gamma(s)$	p-value	Degree assigned by scale	**Known**
4	9	0	0.0039	0.002	**No**
5	1	1	1	0.5	Unable
6	9	1	0.0039	0.998	**Yes**

Table 4.4: Times known is the total number of times the sentence has been played (without counting nonsense votes). $P_\gamma(s)$ is the proportion of people that answered *true* or *false*. The p-value, $p_c(s)$, is the one obtained by the Binomial Hypothesis Test. The **degree assigned by our scale**, $\pi_c(s)$, represents the confidence we have regarding the decision to identify the sentence as known. The last column is the decision made regarding how known is the sentence with a significance of 0.1

$$\begin{aligned} P(X < x) &= \sum_{i=0}^{x} \binom{k(s)}{i} 0.5^i 0.5^{k(s)-i} \\ P(X > x) &= \sum_{i=x}^{k(s)} \binom{k(s)}{i} 0.5^i 0.5^{k(s)-i} \end{aligned} \tag{4.13}$$

Table 4.3 and Table 4.4 shows some examples of sentences with their corresponding $k(s)$, $P_\gamma(s)$, $p_c(s)$ and $\pi_c(s)$. Also, a classification as *known/unknown* is done with $\alpha = 0.1$.

4.3 Identifying a Sentence as Commonsense

To complete our analysis of each sentence s, we need to classify the sentence as commonsense or not. This is accomplished by combining both values $\pi_s(s)$ and $\pi_c(s)$.

Definition 4.3.1 Let $\pi(s)$ be the value that represents how much confidence there is on a sentence being commonsense. $\pi_c(s) \in [0, 1]$, a value of 1 represents a lot of confidence, 0 represents none.

$$\pi(s) = \pi_s(s)\pi_c(s) \tag{4.14}$$

Notice that a sentence s will be commonsense if it is both meaningful (players are able to understand its content) and it is known by most players. Equation (4.14) adequately reflects the behavior that we expect. It is both proportional to $P_\sigma(s)$ and to $P_\gamma(s)$ and with the confidence information of $p_n(s)$ and $p_c(s)$.

Sentence Id	Times Played	$\pi_s(s)$	$\pi_c(s)$	Commonsense Score	**Commonsense**
1	1	0.5	0.5	0.25	Unknown
2	6	0.9844	0.984	0.96899	**Yes**
3	6	0.016	0.5	0.00781	**No**
4	9	0.998	0.002	0.00195	**Domain-specific**
5	8	0.004	0.5	0.00195	**No**
6	9	0.998	0.998	0.9961	**Yes**

Table 4.5: The Sentence Id refers to the sentences in Tables 4.1 and 4.3. **Commonsense Score** is the score obtained by the sentence when evaluated for commonsense, $\pi(s)$. It represents the confidence that we have on identifying each sentence as commonsense. $\pi_s(s)$, and $\pi_c(s)$ are the scores obtained by the sentence when evaluated for *meaning* and *known*, respectively. Times Played is the total number of times the sentence has been played. The last column is the decision made regarding if the sentence is commonsense or not with a significance of 0.1

It will be high (i.e., we will regard s as commonsense) only if s scores high in both $\pi_s(s)$ and $\pi_c(s)$. It will be lower if it scores low in at least in one of the two scales. If s is domain-specific it will only score high in $\pi_s(s)$ but low in $\pi_c(s)$.

Notice that $\pi(s)$ is not linear, this means that if a sentence has double the value of another, it does not necessarily mean that a double amount of people know about it. This scale just provides a ranking in the confidence of a given sentence being commonsense or not.

Table 4.5 show the sentences from the previous tables with their corresponding $\pi(s)$ value. Notice the value of $\pi(s)$ for sentence *1*. Although $\pi(s)$ is not a exactly a probability, its value clearly reflects the fact that, since only one player has evaluated that sentence, the probability of having that count if the player is playing randomly is exactly 0.25 which means that we cannot say for sure if that sentence is commonsense or not. This will be the case for all sentences that have been evaluated only once.

Sentence *2* and *6* are both classified as *commonsense*. Nevertheless, since sentence *6* has been evaluated more times than sentence *2* we have more confidence about the values in the counters which generates a higher value of $\pi(s)$. For sentence *3* and *5* we could not decide if they were known or not but since they were clearly classified as nonsense, $\pi(s)$ takes this into consideration and correctly classifies them as *not commonsense*. As expected, sentence *4* was not classified as commonsense but as domain-specific because $\pi_s(s)$ is high while $\pi_c(s)$ is low.

It is noteworthy that if a sentence s is classified as *nonsense* it will be impossible to also classify it as *commonsense* because $p_c(s) \leq 1$. Although this is actually correct (commonsense must have a meaning attached to it), it is important to remember that meaningless sentences occur in the knowledge base due to an error in the parser. A better parsing algorithm would improve on these results.

As mentioned early, in order to justify that the knowledge collected is not actually domain-specific, a simple analysis on the origin of the players can be done. Our data shows that 1% comes from Europe, 40% from North America and 59% from the rest of the continent. Regarding the age of the players 16% are 15 to 23 years old, 63% are 23 to 33 years old, and 21% are older than 33. It can be observed that the evaluation of the knowledge in our database has information from different backgrounds, increasing the likelihood of our data being commonsense.

Chapter 5

Evaluation

Evaluating knowledge acquisition systems is difficult and there is no consensus on how it should be done, especially when the knowledge comes from volunteer contributors. This adds complexity to any comparative analysis.

Probably the most detailed evaluation of such a system was done for the LEARNER2 in [Chklovski and Gil, 2005]. The authors emphasized the importance of an analysis on coverage and reliability, as well as the presence of knowledge that needs to be identified, discarded or that needs further classification.

For a brief description of the systems that will be compared in this section, the reader is referred to the Related Work chapter.

5.1 Classification of Knowledge as Reliable

None of the systems reviewed offers an explicit way to detect knowledge that should be discarded due to errors or noise in the input of contributors. Also, all of them classify the knowledge as either correct or incorrect, and they do not provide any way to distinguish the data that need further qualification.

Our system provides both features with the help of our scale $\pi(s)$. If $\pi_s(s)$ is less than a given α, the sentence should be erased. α represents the probability of making a Type I error, i.e., discarding a meaningful sentences as nonsense. Also, if a sentence is not clearly nonsense, commonsense or domain-specific, then the game can be directed so that it shows such sentences to players more often until enough data has been collected to make a decision regarding such sentence.

To the best of our knowledge, this is a feature that is not explicitly available in any of the previous related systems.

5.2 Coverage and Redundancy

Because the system depends on contributors (volunteers or players) the knowledge collected is typically ”spontaneous”, which might result in higher redundancy.

Different systems deal with this in different ways. Systems like OpenMind [Singh et al., 2002] and LEARNER2 suggest inputs to the player by showing examples of interesting data. Nevertheless we could not find any analysis about the effectiveness of such a strategy.

Other systems like *Verbosity* [von Ahn et al., 2006], *Common Consensus* [Lieberman et al., 2007], and our own system, *Turing Game*, redirect the input of the player in order to balance the coverage of data. Both *Verbosity* and *Common Consensus* do so by providing varied input to the players. In our case, coverage is handled in two ways. First, the user is suggested three different topics (which the player can refuse and choose from other three options). Once the player chooses a topic, a sentence is randomly selected from the already evaluated sentences and new sentences with a 0.5 probability. This means that we expect half the sentences to be evaluated more than once. This is motivated by the desire to balance coverage and reliability (the more people that evaluate the sentence the more reliable our data).

Although most of the reviewed systems claim that some of their knowledge was entered several times, only LEARNER2 reports data on this (others only casually mention high reliability when compared to judges). Out of 6658 entries, the creators of LEARNER2 claim that only 2088 are different statements, some of them were entered as high as 136 times, and actually 4416 entries yielded only 350 distinct statements. This means that 66.3% of their data is highly redundant. This also means that 1738 statements share 2242 entries which give an average of 1.29 entries per statement. As we showed earlier, these few entries per statement produce unreliable data which means that only 350 statements can actually be trusted, since on average they received 12.6 entries per statement (notice that this average is not a good representative since several statements have more than 100 entries).

On the other hand, our game collected 6763 entries (instances of the game) and generated 3011 evaluated sentences, where the most often evaluated sentence received 15 entries. Also, 2116 sentences have only 1 or 2 entries which make those sentences unreliable (we can easily improve this by directing the efforts of new players to increase these numbers while free input interfaces like those of OpenMind and LEARNER2 are incapable of this). This leaves 4232 entries shared in 1221 statements, giving an average of 3.46 entries per statement which already shows that our data is more reliable than that of LEARNER2. Also, this average is more significant since only a few sentences have more than 10 entries.

Figure 5.1 shows the comparison of coverage and reliability between LEARNER2 and the Turing Game.

5.3 Evaluation of Data by Human Judges

Although there is no general agreement regarding a way of evaluating the quality of collected knowledge, the most common procedure is asking judges to evaluate a sample of the data. The size of the sample and number of judges vary in the

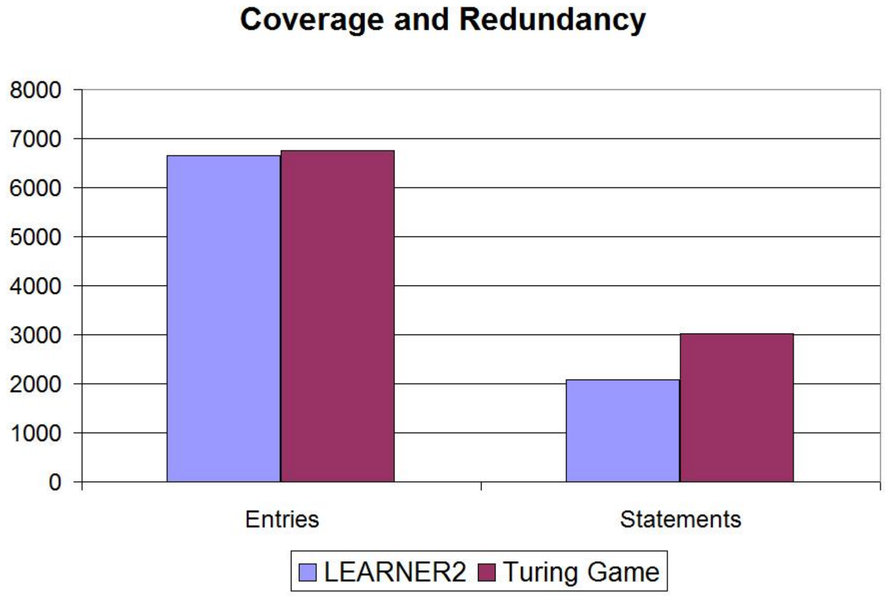


Figure 5.1: Comparison between LEARNER2 and The Turing Game. Although the amount of entries in both systems is similar, the amount of unique statements by those entries is different. The Turing Game provides both more reliable data and more coverage.

literature as well as exactly how to evaluate the data. In some cases, the judges only marked the knowledge as correct/incorrect while in other cases they were allowed to classify the data in a more refined way.

First, we will show an evaluation of our data and then we will compare these results with those from similar systems. For the evaluation, we asked 4 judges to classify a randomized sample of 50 sentences from our knowledge base. The judges were asked to evaluate the knowledge by classifying it in one of four categories: "*Generally/Definitively True*", "*Sometimes/Probably True*", "*Unknown*" and "*Nonsense/Incomplete*". These categories correspond to the 4 possible ways that our game can classify the knowledge (Commonsense, Domain-Specific, Unknown and Nonsense respectively).

In order to compare the results from our knowledge base with those of the judges we chose a significance level of 0.1, i.e., if $\pi(s) > 0.9$ it was considered commonsense, if $\pi_s(s) < 0.1$ it was considered nonsense, if $\pi_s(s) > 0.9$ and $\pi_c(s) < 0.9$ it was considered domain-specific and all other cases unknown. Notice that whenever our game considers a sentence as unknown it is not stating that the fact is unknown, it is just claiming that it does not have enough data in order to correctly classify it. Therefore, during evaluation, it was not considered a mistake if the game accepts that it cannot say anything about a given sentence, regardless of the answer of the judge. Also, if the judge classified a sentence as unknown while the game classifies it as domain-specific, there was no mistake recorded because this only means that the judge does not belong to the restricted set of people that know that fact. For all other cases, if the judge and the game differ, it was counted as a mistake of the game. Notice that this is still probably too hard on the game since the judge might ignore or know something that the rest of the players didn't and therefore it is possible that there is no mistake if the game and the judge differ with respect to commonsense or domain-specific knowledge.

When comparing the answers of the judges to the ones from the game, the average agreement between players and judges was 94%. The disagreement usually occurred in sentences that were slightly misspelled or incomplete. This is because some players and judges disagree on whether to classify such sentences as nonsense or not due to subjective judgements.

5.4 Comparison to Similar Systems

A direct comparison with other approaches is difficult, mainly because each one has a different evaluation procedure according to their own goals. Also, not all of them offered a detailed description of their data. Therefore, we will make comparisons in 4 different categories in which not all of the related work is comparable.

First, we will analyze data obtained over a period of one week. Since *Common Consensus* provide information only about a test-run, their data is not comparable when it comes to data obtained across time. *Verbosity* also provides information of a test-run that lasted one week, which means that the results are not comparable to results obtained through real players. Therefore we will only compare results against OpenMind and LEARNER2.

OpenMind has been online for approximately 8 years and according to [Speer et al., 2008] it has around 700,000 statements. LEARNER2 was available for 3 years and collected 100,000 statements. The data evaluated in the *Turing Game* comes from a 5 week period, and consists of 3011 sentences. Figure 5.2 shows the ratio of data acquired per week by these three systems. OpenMind is clearly larger than the other two methods, nevertheless it is important to notice that this is mainly due to the origin of the project and the amount of time it has been available. In the last years, the interface has suffered from attrition, which is the reason *Common Consensus* has been created to keep encouraging contributors to provide data [Lieberman et al., 2007].

With respect to the data provided by users, LEARNER2 does not provide information about how many users contributed. Although *Verbosity* only reports data based on an experiment, it does include how much data was gathered by a fixed amount of users. We compared the amount of data per user per week provided by each system (See Table 5.1). Notice that the Turing Game performs much better than OpenMind which is also released in the real-world while it performs lower than *Verbosity*, although the comparison might not be fair since the data of *Verbosity* comes from a controlled test-run.

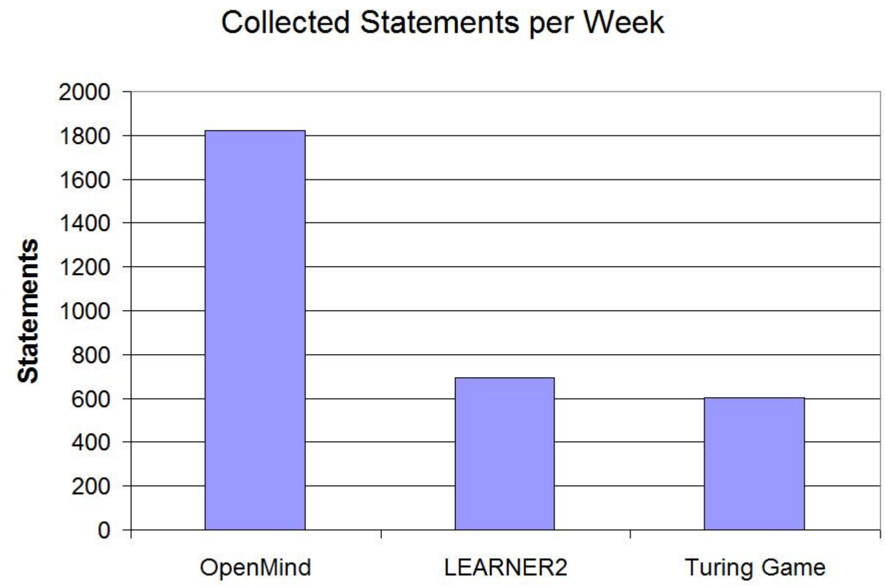


Figure 5.2: Comparison between OpenMind, LEARNER2 and The Turing Game with respect to data acquired per week. Both the Turing Game and LEARNER2 perform similarly while OpenMind has a clear advantage in amount of knowledge collected per week.

When comparing Users per unit of time, we can only compare against OpenMind. We found the performance to be comparable since OpenMind reports an average of 39 users per week while the Turing Game reports 32 users per week.

The last comparison concerns the score reported about the correctness of the data. This is the hardest to compare because each system used a different scale, criteria and number of judges. The results are shown in Table 5.1. Notice that OpenMind used a scale from 1 to 5 that the judges used to rate each entry as General Knowledge. The average is 3.36 which we can interpret as showing that a little over half of the data is real commonsense. *Common Consensus* did not provide any numerical value but commented that all entries that were entered by at least 4 players were regarded as true by the judges; how many of the statements were mentioned by at least 4 players is unknown. *Verbosity* asked the judges to rate each input as correct or incorrect; the judges reported 0.85 of the data to be correct. LEARNER2 used a scale similar to ours and reported that 89.8% of the data that was entered by at least 2 people was correctly common knowledge. The judges of the *Turing Game* reported an average score of 94% agreement with the results of the game, showing that the sample was mostly correctly classified, clearly above all previous systems.

When comparing these results to those from other systems that extract knowledge automatically from text, our system also perform better. The authors of [Berland and Charniak, 1999] reports 0.83 accuracy on a set of 119 extracted relations while [Etzioni et al., 2004] used different rules to extract different kind

	OpenMind	*Verbosity*	LEARNER2	The Turing Game
Judges	7	6	3	4
Data per Week	1822.91	29.48^+	694.44	602.2
Data per User per Week	0.12	29.48^+	N/A	3.76
Users per Week	39.06	267^+	N/A	32
Correctness*	3.26/5	85%	89.8%	**94%**

Table 5.1: Comparison of the Turing Game with Similar Systems. Although each system was evaluated differently, the Turing Game provides **more correct data** and performs similarly to LEARNER2 with respect to data obtained per week, and similar to OpenMind when it comes to users per week. N/A is shown whenever that is the case. *The scale used in each system is different so a direct comparison is not possible. $^+$Data available from a one-week test-run of the game.

of relations; their best performance rule had a precision of 0.8.

Although it is clear that most of these systems are hardly comparable, it is our intention to show that the performance of the *Turing Game* is high and that it provides solutions to problems not addressed by similar systems, such as good coverage, low redundancy and high reliability.

Chapter 6

Conclusions and Future Work

We have presented the design of a game that evaluates and classifies sentences extracted automatically from the Web. This is a mechanism that can also be used in other domains. Although the game can still be improved, we have already collected enough data to show the relevance and reliability of this approach to automate the acquisition of commonsense knowledge from the Web.

One of the main advantages of our design is that it classifies commonsense knowledge in a continuous scale, which allows us to talk about how common a commonsense fact is. Also, data from the game allowed us to obtain training data on how humans distinguish between machine and other humans by using commonsense knowledge. This information is useful for the advancement of human-level AI, and will eventually provide information on how to automatically distinguish commonsense and domain-specific knowledge. This information will allow us to eventually remove humans from the loop of gathering commonsense knowledge.

Errors in the evaluation may be accounted for by the amount of people that played the game. Our statistical analysis gives us confidence about the results in the data even when some of the players disregard the rules and create noisy data. Our analysis identifies which data needs to be evaluated further to improve the confidence and which data has been classified with certainty.

A weakness of our current approach is the use of uninstantiated sentences (such as the example about Geometry "*It means measure the land*"). This is currently solved by making explicit the source of the sentence (in this case it is clear that *it* refers to Geometry) but further work to improve how to present the sentence must be done.

In order to improve these results two things are needed: improving the design of the game so that it is more enjoyable to players (this would increase the number of players that play the game, thus increasing the confidence in the data) and making the rules and scoring policy more accessible so that the players behave in the expected manner.

Although the game has already provided data that shows that our approach is viable, improvements in the design and deployment of the game are possible. One feature that will be further explored in future work is the demographics of the players. Each answer given by the player is stored accordingly to range of age and locale available from Facebook (without storing any personal information). This will prove useful in the sense that we will not only be able to classify commonsense knowledge but we will also be able to cluster commonsense knowledge according to such information.

6.1 Application on Semantic Relatedness

A possible application of the data is to use our scale, along with the concepts available in the sentence as a Semantic Relatedness Measure. These measures have been proven useful in many Natural Language Processing applications such as wordsense disambiguation, information retrieval and spelling correction (See [Budanitsky and Hirst, 2006] for a review on current methods and applications).

A Semantic Relatedness Measure is a way to describe how related two concepts are, regardless of what kind of relation exists between them (in opposition to Semantic Similarity which only cares about how similar two concepts are). Computational applications typically require relatedness rather than just similarity. This is because relatedness not only takes similarity into account (usually described by hyperonyms/hyponyms) but also relations as opposite (antonyms), part-of (meronym) and other ad hoc relations.

Some of the current methods used to measure semantic relatedness use lexical resources such as WordNet [Budanitsky, 2001], others use corpus-based techniques [Gorrell, 2006], and others [Gabrilovich and Markovitch, 2007] use statistical methods.

We propose the use of commonsense knowledge as a way to augment and improve on these methods by using concepts that co-occur in the sentences that are classified as commonsense. This would provide interesting relations between concepts that are not usually available in lexical resources, which would help to better understand the results provided by some of the statistical and corpus-based methods.

As a preliminary evaluation of this idea, we used a set of free association norms [Nelson and Schreiber, 1998] to determine if the concepts expressed in the sentences evaluated as commonsense are indeed related according to human judgement. From a set of 146 sentences, all nouns were compared to pairs of free association. Thirty-seven matches were found which constitutes one fourth of the original data.

These are preliminary results, but they indicate that the concepts expressed in the commonsense sentences are indeed related in a meaningful way. This may be used to infer relatedness between texts.

Table 6.1 shows some of the pairs found as meaningful. The concepts are some of the nouns that appear in the sentences classified as commonsense.

Pairs of Related Concepts	
Checkers	Square
Drop	Water
Body	Head
Coach	sport
Cocaine	Drug
Poker	Player
Steak	Cow
Chess	Game

Table 6.1: Some examples of related concepts according to our commonsense knowledge that match free association norms. Notice that although all concepts are clearly related, the relation is not restricted to hyperonymy or hyponymy.

6.2 Other Possible Applications

Many applications can be improved if they are enhanced with the commonsense stored in our knowledge base. One of these possible applications is automatically annotating images. For example, from a previously annotated image, keywords may be obtained and used in conjunction with our commonsense knowledge base, where we can look for related concepts, and then iteratively improve the keywords associated with the respective image.

Notice that we have restricted the input data to commonsense knowledge, but the basic setting may be used for any kind of input, which means that the Turing Game could be used to verify knowledge bases in general by just changing the input.

Further study in a correct and useful representation of the data is required in order for it to be useful in *commonsense reasoning.* A mapping from natural language to a formal representation is not easy, but it is required if we wish to use this data in more complex tasks.

After enough data has been recorded, a study on how to automate the process of commonsense evaluation will be the focus of our research, with the main goal of fully automating the process of acquiring commonsense knowledge.

As shown, the Turing Game addresses many problems commonly encountered in commonsense knowledge acquisition and provides a sound solution. The possibility of evaluating large commonsense databases is one of the major contributions of this work, the sense that it does not directly compete with previous approaches, but can be added to help evaluate previously collected information.

Appendix A

Glossary of Symbols

Symbol	Description	Definition
$P_\sigma(s)$	Proportion of players that understand a sentence	4.1.1
$e_n(s)$	Effect size to detect if a sentence is meaningful	4.1.2
$p_n(s)$	p-value of Hypothesis Test to detect if a sentence is meaningful	4.1.3
$\pi_s(s)$	Scale that represents the confidence of a sentence being meaningful	4.1.4
$P_\gamma(s)$	Proportion of players that know the truth value a sentence	4.2.1
$e_c(s)$	Effect size to detect if a sentence is known	4.2.2
$p_c(s)$	p-value of Hypothesis Test to detect if a sentence is known	4.2.3
$\pi_c(s)$	Scale that represents the confidence of a sentence being known	4.2.4
$\pi(s)$	Scale that represents the confidence of a sentence being commonsense	4.3.1

Glossary of symbols with their description and reference to where they are defined in the text.

References

[Amir, 2009] Amir, E. (2009). We want more from computers but not too much. Forbes.com, The AI Report.

[Banko and Etzioni, 2007] Banko, M. and Etzioni, O. (2007). Strategies for lifelong knowledge extraction from the web. In *K-CAP '07: Proceedings of the 4th international conference on Knowledge capture*, pages 95–102, New York, NY, USA. ACM.

[Berland and Charniak, 1999] Berland, M. and Charniak, E. (1999). Finding parts in very large corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 57–64, Morristown, NJ, USA. Association for Computational Linguistics.

[Budanitsky, 2001] Budanitsky, A. (2001). Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures.

[Budanitsky and Hirst, 2006] Budanitsky, A. and Hirst, G. (2006). Evaluating wordnet-based measures of lexical semantic relatedness. *Comput. Linguist.*, 32(1):13–47.

[Chklovski and Gil, 2005] Chklovski, T. and Gil, Y. (2005). An analysis of knowledge collected from volunteer contributors. In *AAAI'05: Proceedings of the 20th national conference on Artificial intelligence*, pages 564–570. AAAI Press.

[Custódio and Pinto-Ferreira, 1999] Custódio, L. M. M. and Pinto-Ferreira, C. (1999). Logic of agreement: Foundations, semantic system and proof theory. *Int. J. Approx. Reasoning*, 20(1):47–78.

[Druzdzel, 1993] Druzdzel, M. J. (1993). *Probabilistic reasoning in decision support systems: from computation to common sense.* PhD thesis, Pittsburgh, PA, USA.

[Elio, 2002] Elio, R. (2002). Issues in commonsense reasoning. In Elio, R., editor, *Commonsense, Reasoning and Rationality*, pages 3–36. Oxford University Press, New York.

[Etzioni et al., 2004] Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2004). Methods for domain-independent information extraction from the web: an experimental comparison. In *AAAI'04: Proceedings of the 19th national conference on Artifical intelligence*, pages 391–398. AAAI Press / The MIT Press.

[Gabrilovich and Markovitch, 2007] Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1606–1611.

[Gorrell, 2006] Gorrell, G. (2006). Generalized hebbian algorithm for incremental latent semantic analysis. In *Proceedings of Interspeech*.

[Lenat et al., 1990] Lenat, D. B., Guha, R. V., Pittman, K., Pratt, D., and Shepherd, M. (1990). Cyc: toward programs with common sense. *Commun. ACM*, 33(8):30–49.

[Lieberman et al., 2007] Lieberman, H., Smith, D., and Teeters, A. (2007). Common Consensus: a web-based game for collecting commonsense goals. In *In Proceedings of the Workshop on Common Sense and Intelligent User Interfaces held in conjunction with the 2007 International Conference on Intelligent User Interfaces (IUI 2007)*.

[Matuszek et al., 2005] Matuszek, C., Witbrock, M., Kahlert, R. C., Cabral, J., Schneider, D., Shah, P., and Lenat, D. (2005). Searching for common sense: Populating cyc from the web. In *In Proceedings of the Twentieth National Conference on Artificial Intelligence*, pages 1430–1435.

[McCarthy, 1968] McCarthy, J. (1968). Programs with common sense. In *Semantic Information Processing*, pages 403–418. MIT Press.

[McCarthy, 1990] McCarthy, J. (1990). Artificial intelligence, logic and formalizing common sense. In *Philosophical Logic and Artificial Intelligence*, pages 161–190. Kluwer Academic.

[Nelson and Schreiber, 1998] Nelson, D.L., M. C. and Schreiber, T. (1998). The university of south florida word association, rhyme, and word fragment norms.

[Neufeld, 1990] Neufeld, E. (1990). A probabilistic commonsense reasoner. *International Journal of Intelligent Systems*, 5(5):565–594.

[Schubert and Tong, 2003] Schubert, L. and Tong, M. (2003). Extracting and evaluating general world knowledge from the brown corpus. In *Proceedings of the HLT-NAACL 2003 workshop on Text meaning*, pages 7–13, Morristown, NJ, USA. Association for Computational Linguistics.

[Shahaf and Amir, 2007] Shahaf, D. and Amir, E. (2007). Towards a theory of ai completeness. In *8th International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense'07)*.

[SigArt, 2009] SigArt (2009). Wikipedia knowledge extractor. Special Interest Group on Artificial Intelligence (SIGArt), Association of Computing Machinery (ACM), University of Illinois at Urbana-Champaign (UIUC).

[Singh et al., 2002] Singh, P., Lin, T., Mueller, E. T., Lim, G., Perkins, T., and Zhu, W. L. (2002). Open mind common sense: Knowledge acquisition from the general public. pages 1223–1237. Springer-Verlag.

[Speer et al., 2008] Speer, R., Havasi, C., and Lieberman, H. (2008). Analogyspace: reducing the dimensionality of common sense knowledge. In *AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence*, pages 548–553. AAAI Press.

[Turing, 1950] Turing, A. M. (1950). Computing machinery and intelligence. In *Mind*, pages 433–460.

[Valsiner, 1985] Valsiner, J. (1985). Common sense and psychological theories: The historical nature of logical necessity. *Scandinavian Journal of Psychology*, 26(1):97–109.

[von Ahn and Dabbish, 2004] von Ahn, L. and Dabbish, L. (2004). Labeling images with a computer game. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326, New York, NY, USA. ACM.

[von Ahn and Dabbish, 2008] von Ahn, L. and Dabbish, L. (2008). Designing games with a purpose. *Commun. ACM*, 51(8):58–67.

[von Ahn et al., 2006] von Ahn, L., Kedia, M., and Blum, M. (2006). Verbosity: a game for collecting common-sense facts. In *In Proceedings of ACM CHI 2006 Conference on Human Factors in Computing Systems, volume 1 of Games*, pages 75–78. ACM Press.

[Wikipedia, 2010a] Wikipedia (2010a). Simple wikipedia.

[Wikipedia, 2010b] Wikipedia (2010b). Wikipedia:list of policies and guidelines.