KTH Computer Science
and Communication

Context modeling using a common sense database

GUSTAF LINDSTEDT
MARTIN RUNELÖV

Degree Project in Computer Science, First Level, DD143X
Supervisor: Gabriel Skantze
Examiner: Mårten Björkman

Royal Institute of Technology
School of Computer Science and Communication
KTH CSC 100 44 Stockholm URL: www.kth.se/csc

April 12, 2013

Abstract

Common sense databases are collections of human common sense knowledge. With these databases available, a lot of possibilities arise regarding understanding human language and putting it into a context. This essay aims to discuss how a common sense database could be used to enable a more human-like grasp of context in conversational agents. To try out the possibilities of context modeling using these databases we built a prototype that uses the knowledge base ConceptNet. Our tests showed that there are some issues with ConceptNet for our purposes regarding the quality of the data. However, we were still able to find some interesting previous research in the area, and were able to get some useful results using our prototype.

Referat

Kontextmodellering med en ”common sense”-databas

”Common sense”-databaser är samlingar av mänskligt sunt förnuft. Med dessa databaser dyker det upp en hel del möjligheter gällande förståelse av mänskligt språk och förmågan att sätta det i en kontext. Denna uppsats syftar till att diskutera hur en “common sense”-databas kan användas för att ge ett dialogsystem mer mänsklig uppfattning av kontext. För att testa möjligheterna med kontextmodellering med hjälp av dessa databaser byggde vi en prototyp som använder databasen ConceptNet. Våra tester visade att det finns vissa problem med att uppnå våra mål med hjälp av ConceptNet på grund av bristande kvalitet på viss data. Det finns däremot mycket mer att utforska och vi fann intressant forskning på området, samt fick en del användbara resultat med vår prototyp.

Foreword

This is a Bachelor thesis in Computer Science at the School of Computer Science and Communication at the Royal Institute of Technology the year 2013. Supervisor at CSC was Gabriel Skantze, and examiner was Mårten Björkman.

Statement of collaboration

Most of the work was done together, with close cooperation on all parts of the essay writing. The research was done in parallel, and when something interesting was found both Martin and Gustaf read it. Regarding the prototype used, Gustaf was responsible for the development of the API interface while Martin was responsible for the construction of the graph using this API. The natural language processing and user interface was mainly developed by Gustaf, with some supervision and input from Martin.

Glossary

Term	**Description**
N-gram	A sequence of n items from a given text. The items are typically single words or letters
Concept	A word or sentence in natural language, defining a single idea
Relation	A connection between concepts, describing how they relate to each other
Ontology	A collection of concepts and relations between these concepts
Conversational agent	A program that converses with humans in a coherent way.

Contents

Chapter 1

Introduction

It is notoriously difficult to construct a chatbot that is able to understand human language and respond in kind. Most often, dialogue systems are designed for a specific task which greatly reduces the amount of possible inputs from the user and thus limits what the agent is supposed to be able to understand. Examples of chatbots designed for specific tasks include online support chatbots and therapeutic chatbots like ELIZA [1]. However, creating a more general chatbot has proven to be much more difficult.

The famous Turing Test [2] is designed to test if a machine is intelligent enough to be indistinguishable from an actual human. The test, in its original form, is carried out as follows:

A human judge engages in regular conversations with both another human and the program to be tested. This is usually done in the form of a text-based chat. If the human cannot tell the program from the human, the program has passed the test.

Whether passing this test is sufficient enough to prove human-like behaviour in a computer is, however, questionable. The test was designed to be a challenge to overcome rather than a measurement of how human-like a program actually is. However, this does not mean that passing the test is negligible. For instance, for chatbots like ELIZA which want to encourage the human to keep talking in a therapy-like fashion it is sufficient if the chatbot can keep asking well-formed and relevant follow-up questions. But for more general purpose chatbots it does not suffice to simply print well-formed questions and answers. This is because they do not have any prior knowledge of the subject and context of the conversation, which therefore makes it almost impossible to use templates based on keywords such as in the case of ELIZA.

Template-based chatbots are ill suited for the general domain, since they rely on static templates which are often hand-written. It is immensely hard to cover all of the different things a person can say, although attempts have been made.

A popular approach to solving the problem with building a chatbot for the general domain is by using machine learning, where a computer tries to learn conversational intelligence by example [3]. However, this approach does not give you an intelligent chatbot that is actually “aware” of what is happening. It is simply mimicing human behaviour as best it can. It does not try to reason logically, but tries to respond with the statistically most probable answer to the specific question.

A significant challenge for general purpose chatbots is to be able to accurately keep track of the context of a conversation, relating to previously discussed topics and being able to bring up related subjects which makes sense in the context of the conversation. In the case of a machine learning based chatbot, the previous utterances can be saved and used to further narrow down which answer to the current question is the right one.

If we could make a chatbot more aware of the context of a conversation we could increase the probability that the answers it generates are within that context, thus improving its perceived intelligence.

One solution would be to grant the agent basic knowledge of words and their relations to each other as concepts, so that a method could be devised to read and create sentences based on these relations. This would theoretically eliminate any need for a-piori knowledge of the subject of the conversation. Thankfully, this kind of information regarding the relations of concepts already exist in so called common sense databases.

1.1 Problem statement

With the common sense databases that exist today, it should be possible to mimic a humans ability to keep track of a context and relate it to common sense. This would enable more dynamic conversational agents which are also less dependent of domain. The prime limiting factor would then be the common sense database, eliminating the need to write template-based responses or learn responses from training data. The goal of this paper is to give an overview of how a common sense database can be used to enhance chatbots ability to mimic an understanding of the context of a conversation. This goal is achieved using the following guiding questions:

- What previous attempts have been made?
- What is needed to represent the context of a conversation?
- How can common sense aid in the modeling of a context?

We will also outline a suggestion for how to build a context model with the aid of the common sense database ConceptNet, and discuss what information that can be extracted from the model and how best to use it to mimic an intelligent agent.

There exist many different approaches when constructing dialogue management systems. However, since the focus of this paper will be to examine the possibilities of extracting contexts from a conversation using a common sense database we will not be discussing the construction of well structured and coherent sentences, which is a vital part of successful communication. The relevance of the constructed context models will be judged subjectively since they are not inherently measurable due to their abstract nature and complexity. We will study known chatbots in the field and papers that discuss common sense databases in conjunction with dialogue systems to provide an overview of what work has been done in the field, and what is yet to be tested.

Chapter 2

Background

2.1 Context handling in existing chatbots

2.1.1 A.L.I.C.E.

A.L.I.C.E is a chatbot which was inspired by ELIZA, and has won the Loebner prize[4]. A.L.I.C.E. uses the AIML[5][6] XML schema. AIML is used to create response templates, matching input patterns to different responses and replacing keywords in the response with keywords extracted from the input.

A couple of variables exist in AIML to enable keeping track of the context. The most important variable is "that" which is used to keep track of previous bot outputs. It does not however keep track of previous user input, though this can be accomplished implicitly by providing multiple tiers of templates, effectively constructing a conversation tree based on what the user replied to the previous output. Also, "topic", "think" and "category" variables exists in order to adapt responses according to the topic of the conversation.

These variables can provide some of the functionality required to mimic a humans ability to relate statements to previously discussed topics and infer new ones from the context. However, this method is difficult to use for general purpose chatbots since the conversation templates are static and manually created by humans, making it almost impossible to come close to human-like conversational intelligence.

2.1.2 Cleverbot

Cleverbot is an online chatbot (publicly available at http://www.cleverbot.com) that uses past conversations to generate responses. The idea is to mimic human behaviour by using responses that humans have given to similar sentences. It also has a short-term memory that stores the current conversation, enabling it to filter its possible responses. What this means is that cleverbot can check which of the responses available are the best match for the current conversation. This could be as easy as checking what the most popular response to a certain sentence was when the previous input and response match the current conversations. Cleverbot has the ability to learn foreign languages, since it stores responses for future use, and since it uses the entire conversation to generate responses it can learn context as well. [21]

2.2 Using common sense

It is suggested by both [Tarau, Figa] and [Hadeel, Issa] that an ontology based approach would lead to a more scalable and dynamic agent, without the need for hand-tailored response templates and independent of domain. Tarau and Figa suggests a way to use a common sense knowledge-base in inference processing with a conversational agent. They do this in the context of interactive storytelling, a form of dynamic story influenced by a user's actions. The common sense knowledge base is used to give the program knowledge about concepts and their context. They have a so-called inference engine that dynamically accumulate facts that are related to the current context of a conversation, enabling them to make inferences. This also works as a user-specific short-term memory of sorts, making it possible to interpret ambiguous sentences.
They also mention that when approximating the context of a story, nouns give more meaningful relations than other lexical categories since they are given more reliable common sense classifications.

2.3 Ontologies

An ontology is a formal way to represent a knowledge domain. It contains concepts and relationships between pairs of concepts. An ontology can therefore be used to enable reasoning regarding the concepts it covers. Ontologies can be built as hierarchies to provide information about concepts at different levels of granularity[9]. This is a way to further provide context to concepts, making it possible to, for example, infer the meaning of a concept at a certain moment in time, or within a certain culture. [8] [19]

2.4 Common sense databases

A common sense database is a collection of common sense knowledge. This means that the database contains concepts and relations between these concepts, in other words an ontology. A concept can be any n-gram. The n-grams can be anything from words, to objects, like "Fishing boat", or idioms like "Let the dust settle". Concepts are connected through (directed) semantic relations such as "IsA", "PartOf", "UsedFor", "AtLocation" etc. These connections is what makes a common sense database much more powerful than for example a database of synonyms. The semantic relation "synonym" is simply one possible connection between concepts. [10]

[14]

The semantic relations between concepts can provide information about the connection between concepts that otherwise would be impossible for a computer to infer. Humans know billions of these connections by heart since having such knowledge is a natural part of life. For instance, all humans know that water is a liquid. This information could be contained within a common sense database as follows:

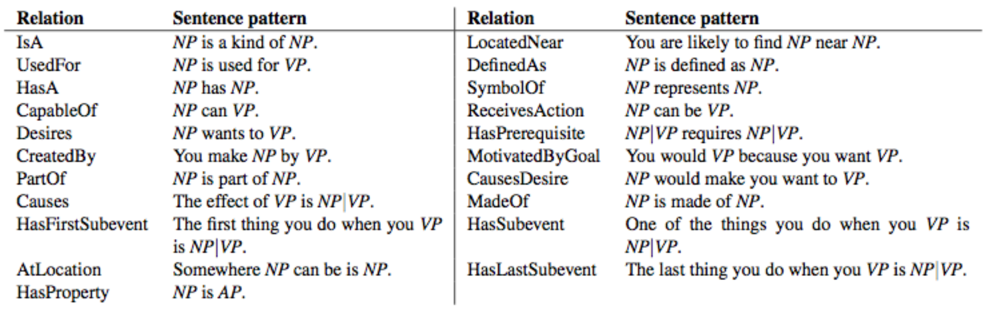
“Water” - “IsA” -> “Liquid”

In the above example we have the concepts “Water” and “Liquid” connected through the directed relation “IsA” that shows that water is a liquid.

2.5 ConceptNet

ConceptNet is a common sense knowledge database. It is built as a hypergraph (where edges can connect to any number of nodes) with about 12.5 million edges and 3.9 million nodes. The nodes are semi-structured word fragments (concepts) that are connected by an ontology of semantic relations (see figure 2.1). Rather than using a taxonomy to provide different levels of granularity, ConceptNet has a non-strict definition of what concepts are, allowing n-grams of different sizes, and a large set of relations (see figure 2.2). The data comes from various other projects such as Open Mind Common Sense[29], WordNet[30] and ReVerb[31] among others. The edges within conceptnet have a score, the higher the score the more it can be assumed that the relation is true. A negative score indicates that the statement is either false or nonsensical. [10] [11]

ConceptNet can be compared to the notable large-scale knowledge base Cyc, which is an AI project with the goal of reaching human-like reasoning. The key difference between these projects is that Cyc is optimized for formalised logical reasoning, with an inference engine that uses logical deduction to draw conclusions, while ConceptNet, with its set of relations and broad definition of concepts, is optimized for context-based inferences in human texts. [12] [14] [19]



Relation	Sentence pattern	Relation	Sentence pattern
IsA	*NP* is a kind of *NP*.	LocatedNear	You are likely to find *NP* near *NP*.
UsedFor	*NP* is used for *VP*.	DefinedAs	*NP* is defined as *NP*.
HasA	*NP* has *NP*.	SymbolOf	*NP* represents *NP*.
CapableOf	*NP* can *VP*.	ReceivesAction	*NP* can be *VP*.
Desires	*NP* wants to *VP*.	HasPrerequisite	*NP*|*VP* requires *NP*|*VP*.
CreatedBy	You make *NP* by *VP*.	MotivatedByGoal	You would *VP* because you want *VP*.
PartOf	*NP* is part of *NP*.	CausesDesire	*NP* would make you want to *VP*.
Causes	The effect of *VP* is *NP*|*VP*.	MadeOf	*NP* is made of *NP*.
HasFirstSubevent	The first thing you do when you *VP* is *NP*|*VP*.	HasSubevent	One of the things you do when you *VP* is *NP*|*VP*.
AtLocation	Somewhere *NP* can be is *NP*.	HasLastSubevent	The last thing you do when you *VP* is *NP*|*VP*.
HasProperty	*NP* is *AP*.		

Figure 2.1. Some key relations in ConceptNet with a description of each [10]. NP stands for Noun Phrase, and VP stands for Verb Phrase.

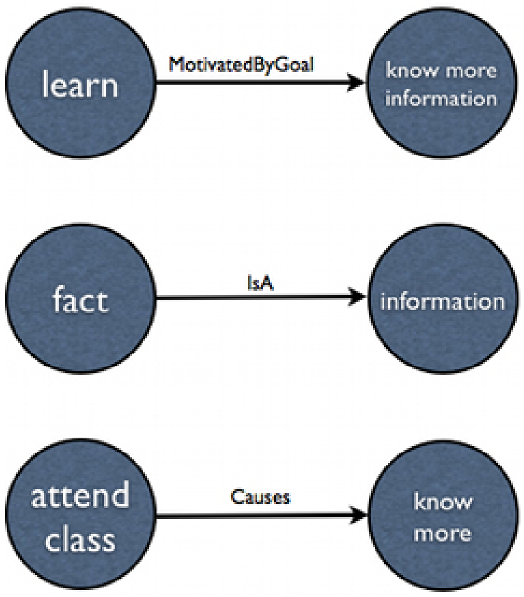


Figure 2.2. Examples of concepts and their relation in ConceptNet.

Chapter 3

Analysis

3.1 Context modeling

At its most basic core, a model of the context of a conversation should at least include a list of important words and phrases that have been brought up during the conversation. This is similar to what Cleverbot does as it keeps track of what has been said to better judge what to say next. Preferably it should also store how the words relate to each other. This is where common sense can play a huge role. For example, if the user mentions cats and dogs, the chatbot should be able to infer that they are both animals.

To prevent the chatbot from straying from the current topic, the context model must in some way represent the linearity of the conversation. In which order has the words appeared? Who said them? Access to this information also enables reasoning when the user omits the subject or object of a sentence. If the user first talks about how he or she loves ice cream, followed by complaints about the bad weather, and then states “It’s too cold.”, the chatbot should see that although both ice cream and the weather can be cold according to common sense, it is most likely the weather that is being referred to since it was the most recent topic.

An even better example of the power of common sense is if the conversation had turned to bicycles, and then the user complained about the cold weather. The common sense should then be able to override the linear representation, giving the topic weather a higher score than bicycle, since one seldom complains of a bicycle being too cold.

To really be able to mimic human-like intelligence, the context model should also store the state of the ‘world’ as described by the user. If the user says “My bike is red.”, the world model should store that the user owns a bike, and that this bike has the property of being the color red. This could prove a difficult task however, since the sentence has to be rigorously analyzed using NLP.

To be able to infer new knowledge from a context model, it needs to be able to expand on its own by finding concepts that have relations in common to concepts mentioned during a conversation. If the user talks about boats and fishing, the concept ocean could be inferred using common sense. Maybe the user lives by the ocean? Now the chatbot has a possibly relevant subject to say something about. And it is less likely that any future occurrence of "ocean" is metaphorical.

This is by no means a complete description of what is needed within a context model, but provides a starting point regarding how to approach the problem.

3.2 Natural language processing

In order to know what words refer to concepts that can be used to infer context, they need to be identified in some way. This is done by using natural language processing to classify the words in the input from the user. The analyzing of the sentence is done using a couple of techniques:

The input is tokenized, meaning that it is split into parts. The most trivial, but still very useful, tokenization is to split a sentence into a list of words. A more advanced tokenization procedure would be to extract all n-grams up to a certain size.

One way to analyze the tokens is to use a process called part of speech tagging, or POS-tagging for short. It involves tagging words in text based on their word class, their relative position to other words, the context and other aspects.

The input can be stemmed using a stemming algorithm. This means that the words are reduced to their base, or root, form. An example of trivial stemming is to remove known suffixes from a word, e.g. "Looking" -> "Look" where the popular suffix "ing" is removed, or "am tired" -> "be tired" where "am" is converted to its base form "be". This allows the database to only store words in their base form, making it easier to collect keyword data.

Chapter 4

Implementation and results

4.1 Implementation

While writing this paper, a prototype was developed to try out some of the techniques mentioned. It was developed in Python and uses the ConceptNet Web API [22] to interface with ConceptNet.

The program continuously takes input from the user, which should write natural language sentences. The program then attempts to create a context using the concepts it finds in the user input. We used basic natural language processing to analyze user input and then query the common sense database using the extracted data to try and make connections between found concepts.

ConceptNet has some built-in NLP such as stemming to make it easy to use. However, since we required more rigorous processing of the input we used the NLTK (Natural Language Toolkit) [24] [25] package to parse and POS-tag the input. We chose to only analyze nouns in our model, which simplified the NLP process. The POS-tagging was done on unigrams using a simplified tagger which uses a less descriptive set of POS-tags. The decision to only parse nouns was made based on the fact that they provide the most information about the context of a sentence. We wanted to narrow our model to minimize the overhead of complex NLP processing.

In this prototype, the contexts are locally stored graphs which contain concepts as nodes and relations between them as edges, similar to the ConceptNet graph. The relations between these concepts are a subset of the relations in conceptnet. The graph representation was chosen because it resembles the structure of ConceptNet and has an intuitive visual representation, and performance was not a priority. The graph was constructed using the library NetworkX [23]

4.1.1 Expanding the context graph

When nouns are discovered in the input from the user, we query ConceptNet using their web API. We make queries for each of the previously discovered concepts, trying to connect them to the new concept. This is complemented by a general query that is not dependent on the existing concepts and only returns the top scoring relations for the queried concept. All of these queries are filtered to use only a small subset of all possible relations to minimize the number of unwanted nodes.

Sample queries:
A query for any ''PartOf" relations between ''leg" and ''table"
```
http://conceptnet5.media.mit.edu/data/5.1/
search?filter=core&rel=/r/PartOf&start=/c/en/leg&end=/c/en/table&limit=2
```

A general query for the top three concepts related to the concept ''dog" through the relation ''IsA"
```
http://conceptnet5.media.mit.edu/data/5.1/
search?filter=core&rel=/r/IsA&start=/c/en/dog&limit=3
```

An example of the process:
The concept "table" is found within a sentence. We query conceptnet and find that a table is part of a kitchen (among other things). Then the concept "leg" is found. We query conceptnet for any connections between all previous concepts (in this instance there is one, "table") and find that a leg is part of a table. We also find that a leg is part of a chair and is a limb in the general query. The resulting graph can be seen in figure 4.1. Concepts that the user provided are green and all other concepts are red. This word-sense disambiguation is an example of how context ontologies can be used to infer the meaning of a sentence.

4.2 Results

While testing our prototype implementation, we found that not all resources that ConceptNet receives its data from are of equal quality. A lot of seemingly nonsensical relations have been gathered using automated procedures. This was partly mended by filtering queries to "core" resources, as ConceptNet calls them, which excludes "ShareAlike" [26] resources like Wikipedia and Wiktionary. These resources do contain lots of valuable information, but due to ConceptNet's loose definition of concepts, a lot of nonsense is produced while processing these resources.

The biggest challenge was to filter on specific relations. Some relations are stronger than others in specific contexts and we have no way of knowing how to prioritize these from concept to concept. Instead, we chose to only use general, strong, relations that always contain valuable connections for all nouns. We experimented with many variations of these, and finally decided to use the following relations:

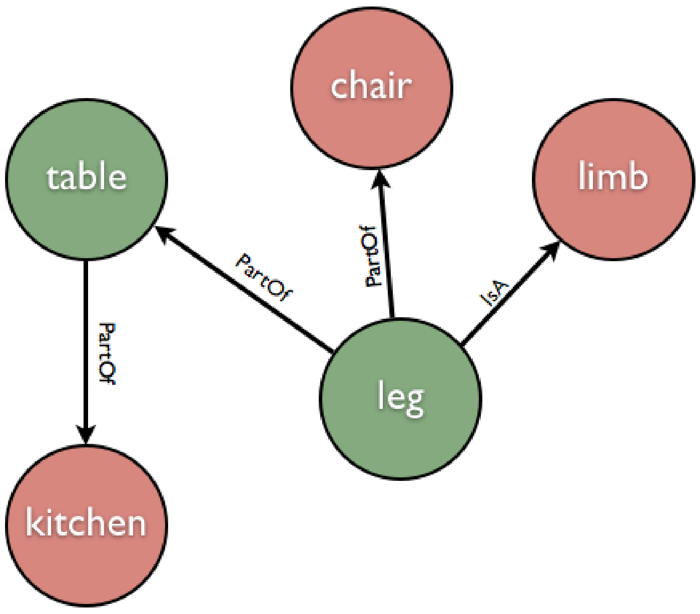


Figure 4.1. A context graph with two mentioned concepts

```
["IsA", "PartOf", "HasContext"]
```

Other relations that were considered were `ConceptuallyRelatedTo`, `AtLocation` and `DerivedFrom`, among others. These relations were searched for individually, with their own limit on the number of results to fetch. The original idea was that this limit, combined with the fact that all results were ordered by a score, would allow us to weigh the importance of each relation, giving us the most meaningful connections. However, we discovered that the score of the edges in ConceptNet were highly unreliable for our purposes. For example, when querying ConceptNet for concepts related to cat by the relation IsA, with cat as the start, the relation cat - IsA -> Animal receives a score of 10.987922; while cat - IsA -> woman_adult_female_person receives a score of 21.975843, about twice as high. And this is even when filtering to only use "core" resources. While it in some cases might be true that a 'cat' is actually referring to a woman, the assertion that a cat is an animal is a much more fundamental statement.

This became a problem since our prototype only took the top results in a query and added to the graph, often leaving out relations to relevant concepts but with lower scores.

We also found that the whole process of receiving input, tagging words, querying ConceptNet and updating the model was pretty slow, taking several seconds, even for a small graph (< 10 nodes). However, performance was not a priority in our prototype, and no steps to improve performance were taken.

Some nice results were achieved however, where relevant connections were made and mentioned concepts were able to connect.

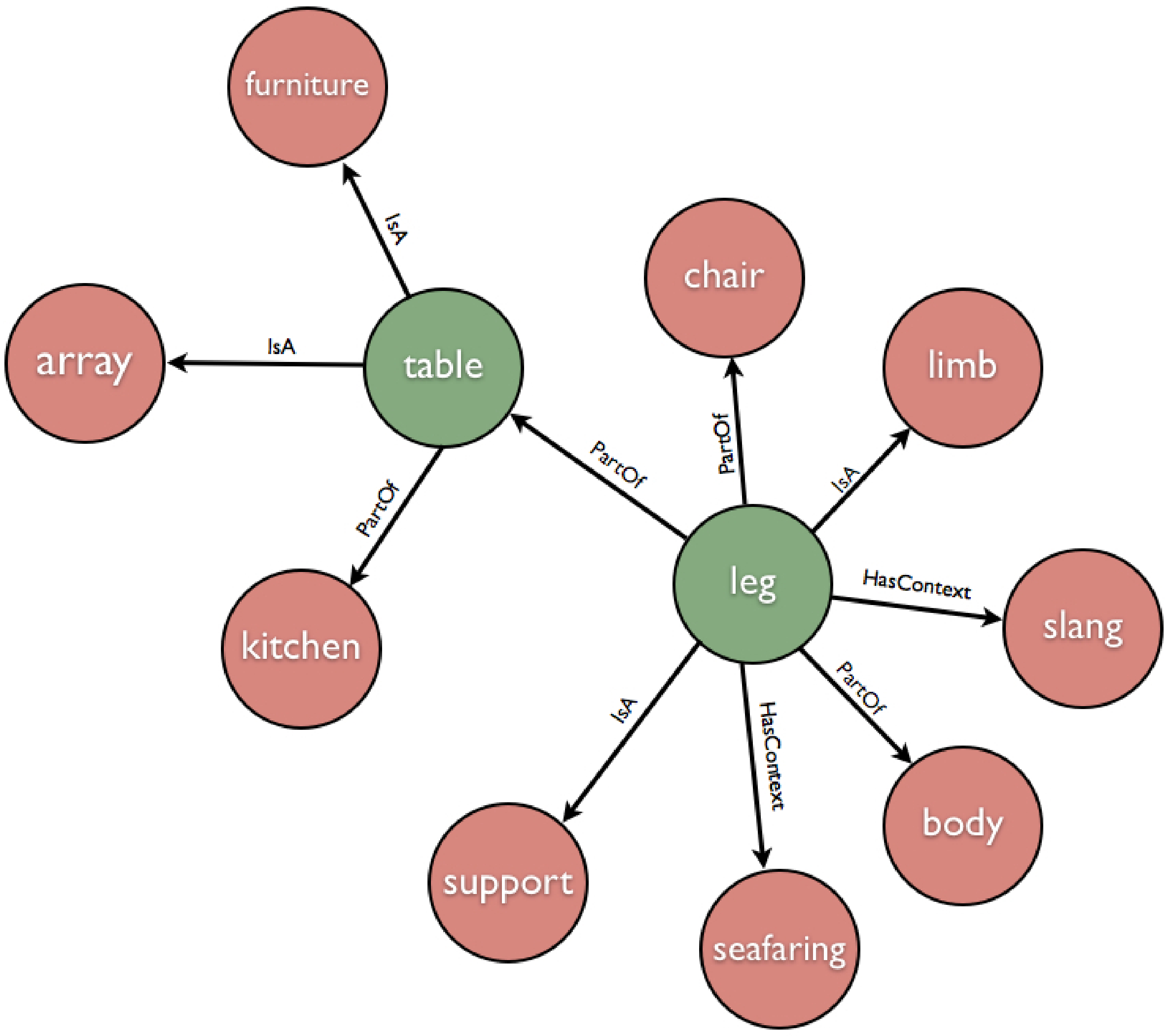


Figure 4.2. The same context graph as before, containing all found connections

Returning to our previous example with the concepts of "table" and "leg", here is the complete graph that was constructed by our prototype. If you are wondering why leg has the context seafaring it is because a leg is *"the distance traveled by a sailing vessel on a single tack"* [28].

Chapter 5

Discussion

5.1 Scoring

The fact that we only used such a small subset of all available relations made it difficult to infer new and unexpected relations. If scoring had been more consistent it would have enabled us to do a general search across all relations, making it possible to utilize ConceptNet's diverse set of relations in a much more powerful way. We were able to make these connections, but at the cost of contaminating the context with all the nonsensical connections that came along with it.

With the current state of the scores, one workaround to improve the graph would be to prioritize the connection of existing concepts by searching all relations for such a connection. However, this would require the existing concepts to already be relevant to the context. It is still difficult to do the initial filtering of what to include when adding a new concept without the help of some kind of score.

The real issue with using the edge score in this way is that the definition of score is that it should be based solely on how true the assertion is. We assumed that the highest scoring relations would also be the most fundamental ones. But since score is only based on trueness, which might have been calculated based on number of occurrences in a source text, this reasoning does not work.

5.2 Possible improvements

There were a lot of functionality discussed in the analysis which we did not implement and try out. This would be a good starting point for further development.

The first thing to improve when building a context is to expand the NLP processing to include more word classes like verbs and adjectives. To fully utilize ConceptNet we should also go beyond unigrams and try to find matches for n-grams of all sizes.

An n-gram of a larger size is also less likely to be ambiguous, with stronger relations to fewer concepts.

An alternative way to infer new concepts from the context, instead of relying on a score giving appropriate answers, is by querying for nodes that are related to more than one concept in the context. If two concepts in the context both relate to a third concept in some way, that concept is likely to be relevant.

Although it was not a priority for our prototype, there are a lot of ways that performance could be enhanced. The first action would be to use a local copy of ConceptNet instead of using the web API. This might also open up the possibility of running more advanced queries on the database.

What was not implemented in our prototype was a representation of the world state as described by the user. This would be a fundamental part in a real conversational agent in order to keep a meaningful conversation.

Another key feature is to keep track of the order in which each concept is mentioned, making it possible to judge relevance based on when the concept was mentioned.

Even if there existed good explicit scores on edges some kind of local weights could be calculated. One such weight could be the total sum of the scores to mentioned concepts, or simply the number of connections if the score is omitted. This weight could be used to grade how relevant each concept is within the context.

Chapter 6

Conclusion

We have examined the possibility of how chatbots can mimic an understanding of context with the aid of a common sense database. We found that common sense databases offer a lot of possibilities to provide a more logical, reasoning approach to conversational agents. Using common sense data and user input to construct a model representing the context however, is no simple task. The text still has to be rigorously analyzed using NLP to enable a fully fledged context model. Furthermore, populating databases with common sense knowledge is still an ongoing effort, with much work to be done. ConceptNet, with its 3.9 million concepts, contains only a microscopical fraction of all the common-sense knowledge an ordinary human knows.

We believe that the work that has been made using common sense knowledge shows great promise, and that the usefulness of these knowledge bases will increase as they grow and improve while we get better at utilizing them. We believe that there is great potential in creating conversational agents with the aid of common sense databases, and expect to see more research concerning it in the future.

Bibliography

[1] Weizenbaum, Joseph. ”ELIZA—a computer program for the study of natural language communication between man and machine.” Communications of the ACM 9.1, 1966.

[2] Turing, Alan M. ”Computing machinery and intelligence.” Mind 59.236, 1950.

[3] Paek, Tim, and Roberto Pieraccini. ”Automating spoken dialogue management design using machine learning: An industry perspective.” Speech Communication 50.8 (2008): 716-729.

[4] Loebner Prize - Home Page. http://www.loebner.net/Prizef/loebner-prize.html (2013-04-11)

[5] Wallace, Richard. ”The elements of AIML style.” Alice AI Foundation (2003).

[6] Artificial Intelligence Markup Language. http://www.alicebot.org/TR/2005/WD-aiml/WD-aiml-1.0.1-008.html (2013-04-11)

[7] Tarau, Paul, and Elizabeth Figa. ”Knowledge-based conversational agents and virtual storytelling.” Proceedings of the 2004 ACM symposium on Applied computing. ACM, 2004.

[8] Gruber, Thomas R. ”A translation approach to portable ontology specifications.” Knowledge acquisition 5.2, 1993.

[9] Van Rees, Reinout. ”Clarity in the usage of the terms ontology, taxonomy and classification.” CIB REPORT 284, 2003.

[10] Speer, Robert, and Catherine Havasi. ”Representing general relational knowledge in conceptnet 5.” Proceedings of Eighth International Conference on Language Resources and Evaluation, LREC. 2012.

[11] ConceptNet - Home Page. http://conceptnet5.media.mit.edu (2013-04-11)

[12] Liu, Hugo, and Push Singh. ”ConceptNet—a practical commonsense reasoning tool-kit.” BT technology journal 22.4, 2004.

[13] Liu, Daphne, and Lenhart Schubert. ”Combining self-motivation with logical planning and inference in a reward-seeking agent.” Proc. of the Int. Conf. on Agents and Artificial Intelligence (ICAART 2010). Vol. 2. 2010.

[14] Panton, Kathy, et al. ”Common sense reasoning–from Cyc to intelligent assistant.” Ambient Intelligence in Everyday Life. Springer Berlin Heidelberg, 2006.

[15] Al-Zubaide, Hadeel, and Ayman A. Issa. ”Ontbot: Ontology based chatbot.” Innovation in Information & Communication Technology (ISIICT), 2011 Fourth International Symposium on. IEEE, 2011.

[16] Leuski, Anton, et al. ”Building effective question answering characters.” Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue. Association for Computational Linguistics, 2009.

[17] Morbini, Fabrizio, et al. ”A mixed-initiative conversational dialogue system for healthcare.” Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue. Association for Computational Linguistics, 2012.

[18] Rao, Sanjay, et al. ”Commonsense via Instant Messaging.” Massachusetts Institute of Technology 20 (2007).

[19] Computers versus Common Sense. Douglas Lenat. - Google TechTalks. http://www.youtube.com/watch?v=gAtn-4fhuWA (2013-04-11)

[20] Natural Language Processing - Wikipedia. http://en.wikipedia.org/wiki/NLP (2013-04-11)

[21] How the Cleverbot computer chats like a human - NBC News. http://www.nbcnews.com/id/44434584/ns/technology_and_science-science/t/how-cleverbot-computer-chats-human (2013-04-11)

[22] ConceptNet Wiki - GitHub. https://github.com/commonsense/conceptnet5/wiki (2013-04-11)

[23] NetworkX - GitHub. http://networkx.github.io (2013-04-11)

[24] Natural Language Toolkit - Home page. http://nltk.org (2013-04-11)

[25] Bird, Steven, Edward Loper and Ewan Klein. ”Natural Language Processing with Python”. O’Reilly Media Inc. (2009)

[26] ShareAlike - creative commons. http://wiki.creativecommons.org/Share_Alike (2013-04-11)

[27] Yampolskiy, Roman V. ”AI-complete CAPTCHAs as zero knowledge proofs of access to an artificially intelligent system.” ISRN Artificial Intelligence 2012 (2011).

[28] Definition of the word ”leg” - The Free Dictionary. http://www.thefreedictionary.com/leg (2013-04-11)

[29] Open Mind Common Sense - Home page. http://commons.media.mit.edu/en/ (2013-04-11)

[30] WordNet - Home page. http://wordnet.princeton.edu (2013-04-11)

[31] ReVerb - Home page. http://reverb.cs.washington.edu (2013-04-11)