Quantifiers Types Resolution in NL Software Requirements

Mehreen Saba[1], Imran Sarwar Bajwa[2]
[1]Department of Computer Science & IT
The Islamia University of Bahawalpur
Pakistan
[2]Schol of Computer Science
University of Birmingham, UK
mehreen.sabba@gmail.com, i.s.bajwa@cs.bham.ac.uk

***ABSTRACT*:** *Natural language quantifiers can be classified according to their semantic type in addition to their syntactic expression. Quantification in Natural language (NL) has two types, ambiguous quantification and Unambiguous quantification. Unambiguous quantification is very simple and also called exact quantification, but ambiguous quantification is complex and also called inexact quantification. Inexact quantifiers include "many, much, a lot of, several, some, any, a few, little, fewer, fewest, Less, greater, at least, at most, more, exactly". To identify the problems of Natural language Quantification, convert these Natural Language sentences into First order logic by attaching weights and classify these complex sentences by using Markov Logic.*

Keywords: NL Quantifiers, Markov Logic, Quantifier Type Resolution

Received: 28 December 2013, Revised 20 February 2014, Accepted 27 February 2014

1. Introduction

Natural languages are inherently ambiguous and informal. In natural languages, English has four main sentence types, Assertive sentences, affirmative sentences, Interrogative sentences and Imperative sentences. Exact quantification of these sentences is simple and easy except affirmative sentences and interrogative sentences. An affirmative sentence gives a positive or negative sense. An interrogative sentence is based on answers either true or false. That s why Quantification of affirmative sentences and interrogative sentences is complex. However inexact quantification of these sentences is ambiguous and complex. Quantifiers bound the range of indication of things. They provide information about "*how much entity*", or "*how many*"entities, not about "*which*" entities. They indicate a quantity somewhere between '*all*' and '*none*'. This is not concerned in reducing natural language expressions to a restricted number of logical concepts. However this is use to the concept of logical in order to better identify with the differences in sense of a variety of linguistic terms.

1.1 Quantifications in Natural Language

Quantifiers are quite common in ordinary language and even more in specialty languages like sciences, in particular mathematics. Unfortunately for the linguists, the more difficult quantifiers are the most common in natural language. Indeed, as we shall see

besides the classical quantifiers of mathematics for all and there exists, natural language makes use of many other quantifiers, some of which are only implicitly stated. The existential quantifier is omnipresent in natural language.

i. Exact Quantification in Assertive sentences

In exact quantification, semantics of the quantification of assertive sentence is clear and simple. However inexact quantification is very complex for processes as compared to exact quantification. There are two example of exact quantification.

There are two people in the room. (1)

In this example 1, it is very clear that two people are in the room. There is no ambiguity in this sentence. It is easily understandable sentence.

ii.Exact Quantification in Interrogative Sentence

In simple sentence exact quantification is clear and easy to understand. However Interrogative Sentences and negative sentences exact quantification is complex.

Are there two people in the room? (2)

In example 2, this is interrogative sentence, after reading this sentence we cannot identify the exact semantics of this sentence. Semantic of this sentence depend on the answer. But sometime even we have answer of that sentence but we could not able to identify the exact meaning of the sentence.

Question: *Are there two people in the room?*

Answer: *Yes, there are two people in the room. No, there are not.*

If answer is yes then this is simple sentence and we know about the semantic of that sentence. However, answer is „No then there is ambiguity.

iii. Exact Quantification in Negative Sentence

In negative sentences

I did not solve three assignments (3)

In this example, there are 5 possible answers. Negative sentence is complex and we cannot identify these sentences easily. This is explaining the negative sentences by 5 states, which is good to analyze the negative sentences.

$$\{\text{All}, > n, = n, < n, N\}$$

There are five states for expressing a number, however there is a negative sentence example; I did not solve three assignments. There is one possibility is he solved exactly three problems.

States	*A*	$> n$	$= n$	$< n$	*N*
All	*A*	-	-	-	-
> three	-	$> n$	-	-	-
= three	-	-	$= n$	-	-
< three	-	-	-	$< n$	-
N	-	-	-	-	*N*

Table 1. Five state for expressing a number

iv. Imperative sentences of exact quantification

Imperative sentences in the English language are the sentences that make a command or request. An imperative sentence is

one that gives a command, direction, or request. The understood subject is the pronoun "*you*" (you should, I want you to).

Leave this place at once. (Order) (4)

In this example 4, talk about order or some straightforward statement. In imperative sentences there is less complex as compared to others.

2. Related Work

All known human languages make use of quantification (Wiese 2004). Languages differ in the variety of quantificational structures they employ [2].

Jon Barwise and Robin Cooper (1981) discussed the standard quantifiers of first order logic not sufficient to meet the quantitative expressions natural languages at least two points. Firstly, there are proposals simply cannot be interpreted merely First-order logic quantifiers. Secondly, the syntactic structure of quantified sentences in calculus is very different from the Quantified the syntactic structure of sentences in natural language. [3]

Themistoklis Chronopoulos, IsidorosPerikos and Ioannis Hatzilygeroudis (2010) conferred an Example-tracing Tutor for the conversion of a Natural language (NL) sentence into first order logic (FOL) sentence. This is a basic knowledge representation language. The tutor is based on the scripting of the process of the NL to FOL conversion and it has been authored using the Cognitive Tutoring Authoring Tool (CTAT) in which he has implemented a completed student interface and he also created a Behavior Recorder graph for process. [4]

Pedro Domingos, Stanley Kok, Daniel Lowd, Hoifung Poon, Matthew Richardson, and Parag Singla (2006) discuses about learners require representations that join probability and relational logic. Markov logic accomplishes this by attaching weights to first-order formulas and viewing them as templates for features of Markov networks. [5]

The subject matter discussed in the article-that seems to have been ignored by most non-computational linguists-is particularly relevant to Computational Linguistics. Michael Hess views the purpose of quantification, two types of explicit quantifier expressions are used in Natural Language (NL). First type of quantifier expressions comprises of the expressions, like "*all*" and "*most*". He argues that these types of expressions are comparatively unusual in real world NL sentences. Second type of quantifier expressions like, "*the*" and "*a*" (articles) cannot be considered straightforward quantifiers. The sources of implicit quantification in NL presented were: syntactic, semantic pragmatic, and stylistic. [6]

Chris Barker (2002) has inspired the use of continuations as relating natural language meaning. The main event is based on the following analysis shows how you can consolidate certain aspects of quantification in new meaning. Which is flat and identify several types of reactions have been critical, therefore, outside, Chris Barker, NP duality of meaning, remove the ambiguity in the order and proportions. Inject some words means that inherently considered serialization. The duration of the method is well established in the theory of the semantics of the programming language. Perfect start as he intended. How certain features of natural language grammar based on continuous quantification new integrated technique: Start state truth except quantificational expressions often target reports displacement volume, duration and ambiguity. To prove it, shows the final composition without semantic context grammar hard for him to conclude that the majority of voters, but the IP quantificational semantic goals. [7]

Farid Meziane (1994) proposed an interactive methodology for generating formal specifications from English specifications. The approach in the area of natural language understanding has to analyze English specifications in order to identify ambiguities. The method used for analyzing natural language text is based on McCord s approach. This method consists of translating natural language sentences into a logical form language representation. [8]

Singla, Parag, and Pedro Domingos (2006) proposed a well-founded, integrated solution to the entity resolution problem based on Markov logic. Markov logic combines first-order logic and probabilistic graphical models by attaching weights to first-order formulas and viewing them as templates for features of Markov networks. [9]

Löbner, S. (1987) supposed the four groups of operators semantically have two operands. They take a predicate quantified

which define a positive phase or range of values on a scale. [10]

The connection between natural language syntax and semantics has received severe notice in both linguistics and computational linguistics for the several decades; it does not show that it has yet been completely adequately recognized. The present article was focuses on quantifier scope ambiguity in a try to identify such a connection. They show that there were some readings that were incorrectly allowed by the theories and that other readings that were available were allowed for the wrong reason. [11]

3. Proposed Framework

A natural language parser is a program that works out the grammatical structure of sentences, for instance, which groups of words go together (as "*phrases*") and which words are the subject or object of a verb. Probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the *most likely* analysis of new sentences.

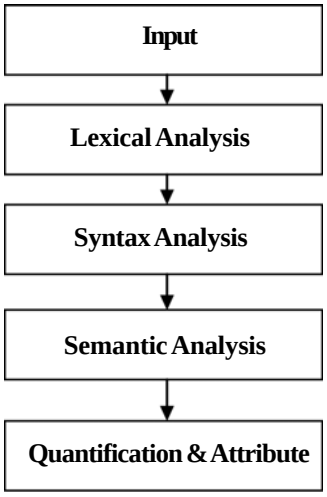

Figure 1. Used Framework

3.1 Input Acquisition

In this phase we give input in the form of English sentences. There is four type of English sentences. Reads input characters and produces a sequence of tokens as output. In natural languages, English has four main sentence types, Assertive sentences, affirmative sentences, Interrogative sentences and Imperative sentences. Exact quantification of these sentences is simple and easy except affirmative sentences and interrogative sentences. An affirmative sentence gives a positive or negative sense. An interrogative sentence is based on answers either true or false. That s why Quantification of affirmative sentences and interrogative sentences is complex. However inexact quantification of these sentences is ambiguous and complex. Quantifiers bound the range of indication of things. They provide information about "*how much entity*", or "*how many*" entities, not about "*which*" entities. They indicate a quantity somewhere between '*all*' and '*none*'. This is not concerned in reducing natural language expressions to a restricted number of logical concepts. However this is use to the concept of logical in order to better identify with the differences in sense of a variety of linguistic terms.

3.2 Lexical Analysis

Lexical analysis is the process of converting a sequence of characters into a sequence of tokens. Probabilistic parsers use knowledge of language gained from hand-parsed sentences to try to produce the *most likely* analysis of new sentences. Lexical analyzer reads input characters and produces a sequence of tokens as output. Trying to understand each element in a program. *Token* is a group of characters having a collective meaning.

Input Example: *I'm bringing some friends with me.*

POS Tagging: I/PRP'm/VBP bringing/VBG some/DT Friends/NNS with/IN me/PRP./.

3.3 Morphological analysis

The length of lexical item and their codification can greatly affect subsequent processing times in automated language

understanding systems. One obvious storage simplification is to perform morphological analysis on words. Morphological analysis is an integral component of the experimental program which is beneficial for storage economy, interpretive assistance, Learning new words and derivational information.

3.4 Syntax Analysis

In syntax analysis, analyze the dependency and generate parsing. Dependency parsing is declared as syntactic model for natural language text. The result of dependency parsing is a graph of words and relations (dependencies) between them within a sentence. Examples of such parsers/model are: Link Grammar, Malt Parser, Stanford parser, etc. syntactic analysis is the process of analyzing a string of symbols.

Parse Tree:

```
(ROOT
  (S
    (NP
      (PRP I))
        (VP (VBP 'm)
          (VP (VBG bringing)
            (NP (DT some) (NNS friends))
              (PP (IN with) (NP (PRP me)))))
(. .)))
```

Typed dependencies, collapsed:

```
nsubj (bringing – 3, I – 1)
aux (bringing – 3, 'm – 2)
root (ROOT – 0, bringing – 3)
det (friends – 5, some – 4)
dobj (bringing – 3, friends – 5)
prep_with (bringing – 3, me – 7)
```

3.5 Semantic Analysis

Semantic analysis could benefit the results of syntactic parser. "*Pure syntactic analysis*" provides a group of potential results and the best result can be only found by semantic analysis.

Quantification is the binding of variable using quantifiers which is all, some, a lot of, many, few, a few, more than half, b/w 5 to 10 etc. All known human languages make use of quantification.

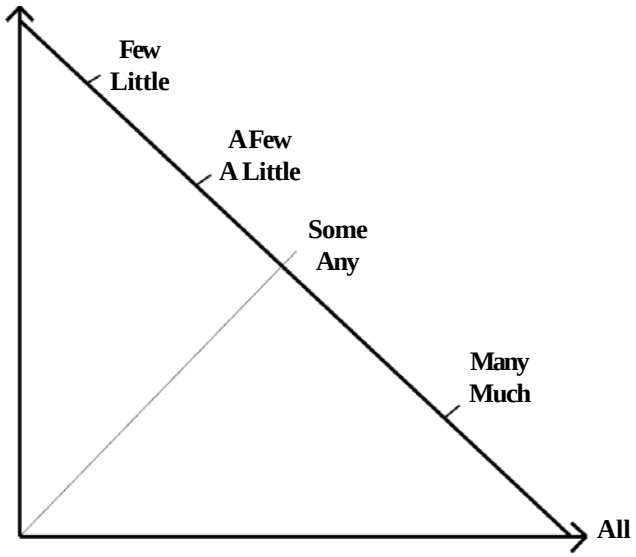

Figure 2. Quantification

Natural language sentences also show a complex range of scope, unlike first-order logic, which clear and simple. All languages are known to user that can be used to quantify. To classify the Quantification of English sentences we can use Markov logic network. There is a Markov logic for assertive, interrogative, Negative and imperative sentence. To assign the weights for these sentences and use the Markov logic formula. An interrogative sentence is based on answers either true or false. That s why Quantification of affirmative sentences and interrogative sentences is complex. However inexact quantification of these sentences is ambiguous and complex. Quantifiers bound the range of indication of things. They provide information about "*how much entity*", or "*how many*" entities, not about "*which*" entities. They indicate a quantity somewhere between '*all*' and '*none*'. This is not concerned in reducing natural language expressions to a restricted number of logical concepts. However this is use to the concept of logical in order to better identify with the differences in sense of a variety of linguistic terms.

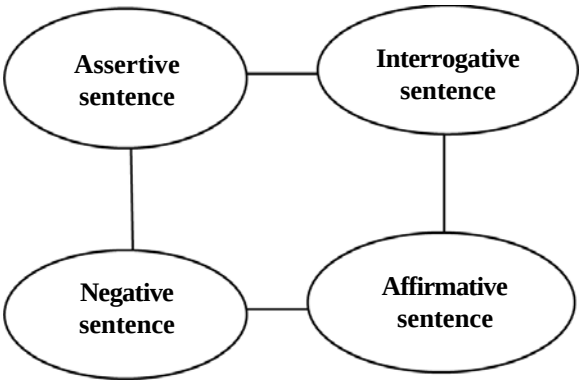

Figure 3. Proposed Markov Logic Network

In Figure 3, we can combine the assertive sentence with the interrogative. The simple assertive sentence is not complex but when it is with interrogative then become more complex.

The best way to do semantics is to specify the '*translations*' of sentences into logic with a model theory and proof theory. (Ted Briscoe and Stephen Clark, 2011) We have considered the semantics of a small fragment of English (which we have only dealt with partially, ignoring e.g. tense/aspect). They have constructed these translations compositionally so that the meaning of a sentence is a function of the meaning of its constituents. They have implemented such a compositional semantics in CFG. [12]

Assertive sentences	Negative sentences	ω
T	T	2.5
T	F	0.5
F	T	2
F	F	0

Table 1. Complexity of Sentences

If a sentence is Assertive and Negative at the same time then this sentence is more complex and Probability of this sentence is high.

$$\forall x \ \forall y: \textit{Together}\,(x, y) \Rightarrow \textit{Assertive}\,(x) \Leftrightarrow \textit{Negtive}\,(y))$$

Clausal Form:

$$\neg \textit{Together}\,(x, y) \vee \textit{Assertive}\,(x) \vee \neg\, \textit{Negtive}\,(y))$$

$$\neg \textit{Together}\,(x, y) \vee \neg\, \textit{Assertive}\,(x) \vee \textit{Negtive}\,(y))$$

If sentence is not assertive or negative at the same time. Then this sentence is simple and probability of sentence is low.

$$\forall x\,(\textit{Alone}\,(x) \Rightarrow \textit{Simple}\,(x))$$

Assertive sentences	Interrogative sentences	ω
T	T	2
T	F	0.5
F	T	1.5
F	F	0

Table 2. Complexity of Sentences

If a sentence is Assertive and Interrogative at the same time then this sentence is more complex and Probability of this sentence is high.

$$\forall x \ \forall y: \textit{Together}\ (x, y) \Rightarrow \textit{Assertive}\ (x) \Leftrightarrow \textit{Interrogative}\ (y))$$

Clausal Form:

$$\neg \textit{Together}\ (x, y) \vee \textit{Assertive}\ (x) \vee \neg\ \textit{Interrogative}\ (y))$$

$$\neg \textit{Together}\ (x, y) \vee \neg\ \textit{Assertive}\ (x) \vee \textit{Interrogative}\ (y))$$

If sentence is not assertive or Interrogative at the same time. Then this sentence is simple and probability of sentence is low.

$$\forall x\ (\textit{Alone}\ (x) \Rightarrow \textit{Simple}\ (x)$$

Affirmative sentences	Interrogative sentences	ω
T	T	2.5
T	F	1.0
F	T	1.5
F	F	0

Table 3. Complexity of Sentences

$$\forall x \ \forall y: \textit{Together}\ (x, y) \Rightarrow \textit{Affirmative}\ (x) \Leftrightarrow \textit{Interrogative}\ (y))$$

Clausal Form:

$$\neg \textit{Together}\ (x, y) \vee \textit{Affirmative}\ (x) \vee \neg\ \textit{Interrogative}\ (y))$$

$$\neg \textit{Together}\ (x, y) \vee \neg\ \textit{Affirmative}\ (x) \vee \textit{Interrogative}\ (y))$$

Affirmative sentences	Negative sentences	ω
T	T	3.0
T	F	1.0
F	T	2.0
F	F	0

Table 4. Complexity of Sentences

$$\forall x \ \forall y: \textit{Together}\ (x, y) \Rightarrow \textit{Affirmative}\ (x) \Leftrightarrow \textit{Negative}\ (y))$$

Clausal Form:

$$\neg \textit{Together}\ (x, y) \vee \textit{Affirmative}\ (x) \vee \neg\ \textit{Negative}\ (y))$$

$$\neg \textit{Together}\ (x, y) \vee \neg\ \textit{Affirmative}\ (x) \vee \textit{Negative}\ (y))$$

A. Look up Table of Natural Language Quantifiers

There is a lookup table of quantifiers with weights, when we check quantification using markov formula then match the results with this quantifier s lookup table to find which quantification lies in the sentence. Quantifiers are quite common in ordinary language and even more in specialty languages like sciences, in particular mathematics. Unfortunately for the linguists, the more difficult quantifiers are the most common in natural language. Indeed, as we shall see besides the classical quantifiers of mathematics for all and there exists, natural language makes use of many other quantifiers, some of which are only implicitly stated. The existential quantifier is omnipresent in natural language.

Quantifiers	ω
Exactly one	0.5
Less than	1.0
Greater than	1.0
More than half	2.0
All	0.2
Each	0.5
Every	0.5
Some	10
Few	1.5
A few	1.2
Little	1.8

Table 5. Value of Quantifiers

B. Using Markov Logic for Classification

Markov logic is (ML) probabilistic logic that the Markov insight in first-order logic, such as the uncertainty statements. Goals of Markov logic generalize first-order logic, in the sense that each statement is unsatisfiable probability equal to zero in a certain limit, and allow all tautologies.

• Each formula matches one clique

• Each formula owns a weight that reflects the importance of this formula

• If a world violates one formula then it is less probable but not impossible

• Concrete: The weight of this formula will be ignored (that means the weight is 0). Following is the used formula of ML.

$$P(X=x)=\frac{1}{Z}\prod_i \phi_i(x_{\{i\}})^{n_i(x)}=\frac{1}{Z}exp\left(\sum_i \omega_i n_i(x)\right)$$

$$P(X=x)=\frac{1}{Z}\prod_i \phi_k(x_{\{k\}})=\frac{1}{Z}exp\left(\sum_i \omega_j f_j(x)\right)$$

Following are the used three assumptions:

1. Unique Names

2. Domain closure

3. Known functions

In grammar, a type of determiner (such as *all, some,* or *much*) that expresses a relative or indefinite indication of quantity. Quantifiers usually appear in front of nouns (as in *all children*), but they may also function as pronouns (as in All *have returned*). A *complex quantifier* is a phrase (such as *a lot of*) that functions as a quantifier.

Stanford Parser is used to identify and separate the quantifiers. After identification, see in the look up table which quantification is involves in the sentence. Stanford Parser is used in comprehensive lexical and syntax analysis of English sentences. The

presented approach is also implemented in Java as a prototype tool. [13]

4. Conculsion

Natural language quantifiers are classified according to their semantic type in addition to their syntactic expression. Inexact quantifiers consist of "*many, much, a lot of several, some, any, a few, little, fewer, fewest, Less, at least, more, at most*". "*Many*", "*several*", "*a few*", "*fewer*", and "*fewest*" are only used with „Count Nouns . "*Much,*", "*Little*", "*a little*" are used with mass nouns . "*Some*", "*any*"," *a lot of*", "*least*", "*less*" may be used with both '*mass*' and '*count*' nouns. "*Some*" and "*any*" have a range of uses they could justify their inclusion in the class of articles. To identify the problems of Natural language Quantification, convert these Natural Language sentences into First order logic by attaching weights and classify these complex sentences by using Markov Logic. The results of Markov logic formula are matched with quantifiers look up table and identify which type of quantification involve in the sentence. In quantifiers where "*more than half*" quantification involve in the sentence that is more complex than other quantifiers.

5. Future Work

Stanford Parser is used to identify and separate the quantifiers. After identification, see in the look up table which quantification is involves in the sentence. Stanford Parser is used in comprehensive lexical and syntax analysis of English sentences. The future work is the implementation of this purposed framework in java.

References

[1] Nilsson. http://www-scf.usc.edu. [Online]. http://www-scf.usc.edu/~csci460/docs/lecture/logic.pdf

[2] Mark Baker. (2005). On the Absence of Certain Quantifiers in Mohawk, p. 1-613.

[3] Jon Barwise, Robin Cooper. (1981). Generalized Quantifiers And Natural Language, in *Linguistics and Philosophy*, 4, p. 159-219.

[4] Themistoklis Chronopoulos, Isidoros Perikos, Ioannis Hatzilygeroudis. (2010). An Example-Tracing Tutor for Teaching NL to FOL Conversion, in *IFIP Advances in Information and Communication Technology*, p. 170-178.

[5] Matthew Richardson, Pedro Domingo. (2006). Markov Logic Networks, in *Machine Learning*, 62, p. 107-136.

[6] Hess, M. (2010). How does Natural Language Quantify?, University of Zurich, Seminar of General Linguistics, Plattenstrasse 54, CH-8032 Zurich, Switzerland.

[7] Barker, Chris. (2002). Continuations and the nature of quantification. *Natural language semantics* 10 (3) 211-242.

[8] Vadera, Sunil, FaridMeziane. (1994). From English to formal specifications. *The Computer Journal*, 37 (9) 753-763

[9] Singla, Parag, Pedro Domingos. (2006). Entity resolution with markov logic. *Data Mining.ICDM'06. Sixth International Conference on*. IEEE.

[10] Löbner, S. (1987). Quantification as a major module of natural language semantics.*Studies in discourse representation theory and the theory of generalized quantifiers*, p. 53-85.

[11] Keenan, Edward, L. (1996). 2 the Semantics of Determiners.

[12] Ted Briscoe, Stephen Clark. (2011). University of cambridge. [Online]. http://www.cl.cam.ac.uk/teaching/1112/L107/materials.html

[13] Naeem, M. Asif, Imran SarwarBajwa. (2012). Generating OLAP Queries from Natural Language Specification. *In*: Proceedings of the International Conference on Advances in Computing, Communications and Informatics. ACM.