UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

# Ambiguity Identification and Measurement in Natural Language Texts

Mariano Ceccato, Nadzeya Kiyavitskaya, Nicola Zeni, Luisa Mich,
Daniel M. Berry

December 2004

Technical Report DIT-04-111

# Ambiguity Identification and Measurement in Natural Language Texts

Mariano Ceccato[1], Nadzeya Kiyavitskaya[2], Nicola Zeni[2], Luisa Mich[2], Daniel M. Berry[3]

[1] Centro per la Ricerca Scientifica e Tecnologica (in English), ITC-irst, Trento, Italy
ceccato@itc.it
[2] Department of Information and Telecommunication Technologies, University of Trento, Italy
{nadzeya.kiyavitskaya, nicola.zeni, luisa.mich}@unitn.it
[3] School of Computer Science, University of Waterloo, Canada
dberry@uwaterloo.ca

**Abstract.** Text ambiguity is one of the most interesting phenomenon in human communication and a difficult problem in Natural Language Processing (NLP). Identification of text ambiguities is an important task for evaluating the quality of text and uncovering its vulnerable points. There exist several types of ambiguity. In the present work we review and compare different approaches to ambiguity identification task. We also propose our own approach to this problem. Moreover, we present the prototype of a tool for ambiguity identification and measurement in natural language text. The tool is intended to support the process of writing high quality documents.

**Keywords:** ambiguity identification, ambiguity measures, text quality, natural language.

## 1. Introduction

Ambiguity is phenomenon of natural language. It means the capability of being understood in two or more possible senses or way. Identification of ambiguous words and phrases is a crucial aspect in text processing applications and many other areas concerned with human communication. The main focus of the present work is the problem of ambiguity identification in natural language documents. In this paper we review existing methods for ambiguity identification and present a prototype of the tool for ambiguity identification and measurement.

The paper is organized as follows: section 2 recalls some theoretical issues of the ambiguity problem domain; section 3 reviews the main research projects related to ambiguity identification; section 4 introduces our approach to this problem; section 5 presents the results some experiments; section 6 describes the requirements for an ambiguity identification tool, and presents a prototype of this tool and its evaluation on the example of two case studies. Conclusions are drawn in section 7.

## 2. Theoretical Background

### 2.1 Motivation

Ambiguity is a pervasive phenomenon in human languages, and is fundamentally a property of linguistic expressions. There are two basic interpretations of ambiguity:

i) the capability of being understood in two or more possible senses or ways;

ii) uncertainty [1].

Uncertainty means lack of sureness about something and has to do with the writer's and reader's knowledge of the background. The issue of uncertainty will not be considered in this paper; here we use the first interpretation of ambiguity.

A word, phrase, sentence, or other message is called *ambiguous* if it can be reasonably interpreted in more than one way. It is difficult to find words that do not have at least two possible meanings, and sentences which are (out of context) several ways ambiguous are the rule, not the exception. Ambiguity gives natural language its flexibility and usability, and consequently it cannot be eliminated.

Ambiguity is of great importance in many areas. For instance, art and politics are the domains in which, for different reasons, ambiguity is essential. Songs and poetry often rely on ambiguous words for artistic effect, as in the song title 'Don't It Make My Brown Eyes Blue' where 'blue' can refer to the color, or to sadness. In literature and rhetoric ambiguity is often used as a source of humor and jokes, one well-known example is: 'Last night I shot an elephant in my pajamas. What he was doing in my pajamas I'll never know'.

In politics or law, on the other hand, ambiguity creates space for defining relationships or bargaining over shared goals. Any legal document that acts as a recipe or standard for performance must be precise, accurate, and self-consistent, anticipating all possible contingencies. This also applies to contracts, patents, wills, statues, political agreements, medical prescriptions, etc.

The similar properties of being precise and consistent are also shared by system requirement specifications (SRSs). Ambiguity in requirement specifications can cause numerous problems. SRSs often form the basis of a contract between the customer and the system supplier. Consequently, SRSs need to be very precise, because they can serve as the basis of a specification of what the system is supposed to do. SRSs also need to be clear enough such that the customer can confirm that a system built from requirement specification satisfies his/her needs.

Another application that requires ambiguity identification is Machine Translation (MT). The existence of ambiguous words makes it more difficult for an MT system to capture the appropriate meaning of a source sentence so as to produce the required translation. Therefore translation systems must be able to identify and correctly resolve such cases.

Ambiguity also plays important role in Natural Language Generation (NLG). When generating text, ambiguity must be managed carefully: some ambiguities can be preserved or eliminated; and the problem arising in NLG systems is to decide which ones should be removed, and which can be allowed to remain.

Ambiguity of words must be resolved when performing Information Retrieval (IR) or Information Extraction (IE) to ensure that the results of a query are relevant to the intended sense of the query term(s). Ambiguity identification is also crucial for the part-of-speech tagging, speech processing, hypertext management, semantic annotation, or any other text processing application dealing with content.

The main motivation for the present paper is that the tools for ambiguity identification can come in handy to assist the writer to create high quality documents, warning him/her about the doubtful points. If the user is additionally provided with the information on the kind of ambiguity presented in a particular place, this knowledge can also help him in rewriting this part of the text.

Of course, not all the ambiguities can be easily uncovered. Dealing with some of them requires deep linguistic analysis. The following section recalls briefly the main types of ambiguity in natural language (the review is based mainly on the [1]).

## 2.2 Types of Ambiguity

***Lexical ambiguity*** occurs when a word has several meanings. For instance, the word 'light' as an adjective can mean 'of comparatively little physical weight or density' or 'having relatively small amount of coloring agent', etc. (taken from WordNet 2.0). Words like 'light', 'note', 'bear' and 'over' are lexically ambiguous. Lexical ambiguity can be further subdivided into homonymy and polysemy.

*Homonymy* occurs when two different words have the same written and phonetic representation, but unrelated meanings and different etymologies, i.e., different histories of development. Each of the homonyms has its own semantics. An example is a word 'bank' which can mean 'financial institution' or 'slope'. *Polysemy* occurs when a word has several related meanings but one etymology. The different meanings of a polysemic expression have a base meaning in common. An example is word 'point': 'punctuation mark', 'sharp end', 'detail, argument' etc.

***Syntactic* (*structural*) *ambiguity*** occurs when a given sequence of words can be given more than one grammatical structure, and each has a different meaning (when the sentence has more than one parse). For example, the phrase 'Tibetan history teacher' or the sentence 'The police shot the rioters with guns' are structurally ambiguous. A syntactic ambiguity can be classified as an analytical, attachment, coordination, or elliptical ambiguity.

*Analytical* ambiguity occurs when the role of the constituents within a phrase or sentence is ambiguous (ex., 'porcelain egg container').

*Attachment* ambiguity occurs when a particular syntactic constituent of a sentence, such as a prepositional phrase or a relative clause, can be legally attached to two parts of a sentence. The most popular pattern of attachment ambiguity is a prepositional phrase that may modify either a verb or a noun. For example, the sentence 'The girl hit the boy with a book' can be interpreted either as 'the girl used a book to hit the boy' or as 'the girl hit the boy who had a book'.

*Coordination* ambiguity occurs when:

- more than one conjunction, "and" or "or", is used in a sentence (ex., 'I saw Peter and Paul and Mary saw me');
- one conjunction is used with a modifier (ex., 'young man and woman').

*Elliptical* ambiguity occurs when it is not certain whether or not a sentence contains an ellipsis. Ellipsis is the deliberate omission of some aspect of language form whose meaning can be understood from the context of that form. Ellipsis is sometimes called gapping by linguists. Example of elliptical ambiguity is 'Perot knows a richer man than Trump'. The phrase has two meanings. First, that Perot knows a man who is richer than Trump is, and second, that Perot knows a man who is richer than any man Trump knows. The first meaning corresponds to having no ellipsis, and the second corresponds to having an ellipsis which is the implied 'knows' coming after 'Trump'.

***Semantic ambiguity*** occurs when a sentence has more than one way of reading it within its context although it contains no lexical or structural ambiguity. Semantic ambiguity can be viewed as ambiguity with respect to the logical form, usually expressed in predicate logic of a sentence. Semantic ambiguity can be caused by:

- coordination ambiguity,
- referential ambiguity, and
- scope ambiguity.

*Coordination* ambiguity can cause both syntactic and semantic ambiguity and its notion was already discussed above.

*Referential* ambiguity will be discussed below because it is on the border line between semantic and pragmatic ambiguity (as far as referential ambiguity can happen within a sentence or between a sentence and its discourse context).

*Scope ambiguity* occurs when quantifier or negation operators can enter into different scoping relations with other sentence constituents. Quantifier operators include such words as 'every', 'each', 'all', 'some', 'several', 'a', etc., and the negation operators include 'not'. An example is the sentence 'Every man loves a woman', which has two distinct readings: for each man there is "his" woman, and he loves her, or, alternatively: there is a special woman which is loved by all the men.

***Pragmatic ambiguity*** occurs when a sentence has several meanings in the context in which it is uttered. The context comprises the language context, i.e., the sentences uttered before and after, and the context beyond language, i.e., the situation, the background knowledge, and expectations of the speaker and hearer or the writer and reader. It is traditionally distinguished between referential ambiguity and deictic ambiguity.

The relation between a word or phrase and an object of the real world that the word or phrase describes is called a *reference*. An *anaphor* is an element of a sentence that depends for its reference on the reference of another element, possibly of a different sentence. This other element is called the *antecedent* and it appears earlier in the same sentence or in a previous sentence.

*Referential ambiguity* occurs when an anaphor can take its reference from more than one element, each playing the role of the antecedent. Anaphora includes pronouns, e.g., 'it', 'they', definite noun phrases, and some forms of ellipses. An example of referential ambiguity is 'The trucks shall treat the roads before they freeze'. Ellipses can have the same effect as pronouns and definite nouns: '…If the ATM accepts the card, the user enters the PIN. If not, the card is rejected'.

*Deictic ambiguity* occurs when pronouns, time and place adverbs, such as 'now' and 'here', and other grammatical features, such as tense, have more than one reference point in the context. The context includes a person in a conversation, a particular location, particular instance of time, or an expression in the previous or following sentence. In contrast to an anaphor, a deictic or another definite expression is often used to introduce a referent. Anaphoric references include pronouns and definite noun phrases that refer to something that was mentioned in the preceding linguistic context, by contrast, deictic references refer to something that is present in the non-linguistic context. Note that a pronoun, in particular, can be anaphoric or deictic. When a pronoun itself refers to a linguistic expression, or chunk of discourse, the pronoun is deictic; when a pronoun refers to the same entity to which a prior linguistic expression refers, the pronoun is anaphoric (ex., 'A man walked in the park. He whistled.'). An ambiguity also occurs when a pronoun can be read as an anaphoric or deictic expression, as shown in the following example, which has elements of scope, referential and deictic ambiguities: 'Every student thinks she is a genius'.

## 3. Related Works

A large variety of work has been done in the field of ambiguity and a number of linguistic theories has been developed previously. Resolution of different types of ambiguity is a stage required for many natural language understanding applications.

If we consider in particular Word Sense Disambiguation (WSD), this research field has a long history. Broad review of the most important historical steps of the development in WSD is given in [5]. This work analyzes the main innovations in the WSD research area until 1997. Yarowsky et al. [11] made have made a significant improvement by applying statistical techniques to WSD problems. At the end of 1997 an important step was done with the constitution of an international organization SENSEVAL (www.senseval.org) which evaluated the quality of WSD Systems. The core mission of SENSEVAL is to organize and run evaluation and related activities to test the strengths and weaknesses of WSD systems with respect to different words, different aspects of language and different languages. SENSEVAL has provided an excellent test bed for the development of practical strategies for analyzing text. However, the main concern of our work is ambiguity identification task.

One of the earliest works devoted to estimation of the degree of polysemy in texts was made by Harper in early days of machine translation (as it is reported in [5]). Working on Russian texts, he determined the number of polysemous words in an article on physics to be approximately 30% and 43% in another sample of scientific writing. He also estimated the degree of polysemy in dictionaries.

Research on the identification of text ambiguity was also performed in the Requirements Engineering (RE) domain. In order to evaluate the quality of requirement documents the tools available so far in RE use lexical and syntactical analysis to measure the level of ambiguity. Ambiguity value is rated by evaluating its sub characteristics which are specific for the RE domain, such as: vagueness, subjectivity, optionality, and weakness of the requirements text.

For instance, automatic analysis tool called QuaARS (Quality Analyzer of Requirement Specification) [3] has been devised to evaluate the quality of real SRSs. NLP techniques are applied to requirements documents in order to control the vocabulary and style of writing. The tool points out the defects and allows the user to decide whether to modify the document or not. One of the qualitative properties taken into account by the tool is non-ambiguity, i.e. the capability of each requirement to have a unique interpretation. The approach uses set of keywords to detect potential defects in the NL requirements.

The similar tool, named ARM (Automated Requirement Measurement) [10], was developed by NASA (National Aeronautics and Space Administration) for automated analysis of requirement specifications. In ARM, a quality model similar to that defined in QuARS is employed. The tool also identifies text quality indicators, such as weak and ambiguous phrases in requirement texts. In both tools the user can supply new domain-dependent quality indicators.

Another work on ambiguity measures in RE [8] also investigated the indices of ambiguity in natural language. The definition of these indices for single words is based on the number of semantic meanings, which is denoted as semantic ambiguity, and syntactic roles, denoted as syntactic ambiguity. Ambiguity indices can be also represented as a weighted function depending on the frequency of occurrences of different meanings or syntactic roles. In similar way ambiguity measures for the sentence are defined. Ambiguity measures are calculated using the functionalities of general purpose NLP system LOLITA.

The work [6] also refers to the problem of detecting ambiguities in requirements documents and suggests using metamodels for identification of ambiguities. The approach proposed in this work aims at identification of those ambiguity types that can occur in a particular RE context. The authors offer an inspection technique for detecting ambiguities in informal requirements documents before formal software specifications are produced. In order to detect ambiguities two methods are selected: checklists and scenario-based reading. They can be developed if a metamodel that characterizes the particular RE context is available. A metamodel is supposed to define a language for model specification (which should be taken from previous metamodelling efforts). Several heuristics that analyze relations, concepts and specifications, are developed to identify ambiguities.

In some applications the researchers try not to resolve but rather to minimize the number of ambiguities. One of the projects that exploit this approach is KANT machine translation system [9]. This work introduces the following restrictions on the language of input text: constrained lexicon, a constrained grammar, semantic restriction using a domain model, and the limitations of noun-noun compounding. KANT supports also interactive disambiguation of text. Its on-line authoring system is able to indicate lexical and structural ambiguities. If the sentence is considered to be ambiguous the author is asked to rewrite it.

The problem of ambiguity detection applied to NLG domain is considered in [2]. This work proposes ambiguity notification tool which follows similar approach as in KANT project. The system notifies potential ambiguities in the text to be generated and describes them to the user which can select whether to leave or to change the text. This approach introduces interesting notions of the seriousness and tolerance levels. *Seriousness level* – is the seriousness of the different ambiguities. *Tolerance level* – is a level of seriousness below which an ambiguity is tolerated. These levels are adjusted while learning during interactions with user and they may be reset at any point.

The approach used in the present work to build ambiguity identification and measurement tool is based on the ideas introduced by Mich and Garigliano in [8].

## 4. Our Approach

In our approach we begin with the simplest task, i.e. identification of lexical ambiguity. The dictionaries have been traditionally used to identify lexical ambiguity, because people usually refer to the meaning reported in dictionary while talking about sense of word. Dictionaries describe the meanings of all word senses and they can be also used to assign the right sense to a word. In our work, in order to identify lexical ambiguity based on the number of words senses, we use machine readable lexical resources. Two of them are dictionaries and the third is a thesaurus. The reasons for choosing these particular resources are threefold. Firstly, their free availability on-line. Secondly, the convenience of using them which allows quickly browsing the knowledge base. Thirdly, they have user friendly interfaces integrated with commonly used text editors and common way to represent output.

In our approach we tried to extend the notion of lexical ambiguity from the word level to the sentence level (as a first approximation to semantic ambiguity measure of the sentence). As we know ambiguities of words within a sentence in some sense 'multiply', therefore our starting assumption is that the ambiguity of the whole sentence can be expressed as the combination of lexical ambiguities of its each word (this combination can be simply product or sum, or other more complex mix of ambiguity values).

We suppose that, in general, semantic ambiguity of the sentence can be represented as a function that depends on the dictionary parameters (dimension of the dictionary or other more expressive characteristics, e.g. domains coverage), ambiguity of each word in the sentence and other parameters.

$$SA = F(D;\ a_1 \ldots a_N;\ \ldots),$$

where $SA$ – sentence ambiguity;
$F$ – some function;
$D$ – dictionary parameter;
$a_i$ – lexical ambiguity of word $i$ in the sentence;
$N$ – number of words in the sentence.

In our experiments, as possible measures to estimate semantic ambiguity of the sentence, we developed function $F$ by two mathematical functions: summarization and multiplication of lexical ambiguities of each word (to the product value was then applied a logarithm on base 2, related to the definition of an amount of information from Information Theory [4]). Sentence ambiguity calculated as a sum is the simplest, "rough" measure that one could apply to estimate ambiguity. Here we assume that for any word the probabilities of all its senses are the same. More sophisticated measures of sentence ambiguity could take into account also that some senses of words are more frequent or less frequent than others [7]. In our experiments we investigated how representative are the measures introduced above.

$$SA_{sum} = \sum_{j=1}^{N} n_j \qquad\qquad SA_{prod} = \log_2 \prod_{j=1}^{N} n_j \quad \text{where}$$

$SA_{sum}$ – sentence ambiguity calculated as a sum of lexical ambiguities of each word;
$SA_{prod}$ – sentence ambiguity calculated as a logarithm from product of lexical ambiguities of each word;
$n_j$ – number of senses for word $j$ in a given dictionary;
$N$ – number of words in the sentence.

Another type of ambiguity that can be identified is syntactic (structural). Syntactic ambiguity can be captured by returning parse trees of a phrase or sentence. In order to produce the set of all possible structures a parser can be used.

Pragmatic ambiguity is the most difficult type of ambiguity to discover using automatic tools. It should be handled using a representation language for sentence meaning, therefore recognizing this type of ambiguity demands advanced level of linguistic analysis, involving large knowledge bases and sophisticated methods for representing different semantic interpretations.

Basing on the previous considerations we propose the prototype of an ambiguity identification tool. The developed tool deals with only lexical ambiguities. Though in future development the tool can be extended to be able to deal with more complicated types of ambiguity.

## 5. Case Studies

Before implementing the ambiguity identification tool, we considered two preliminary case studies in software engineering domain. In the first one our goal was to investigate lexical ambiguity of single words, in the second one we tried to extend the notion of lexical ambiguity from the word level to the sentence level (as a first approximation to semantic ambiguity measure).

In our experiments we used the following lexical resources:

- WordReference (www.wordreference.com) is based on Collins English dictionary, which covers a wide range of fields. From the chosen dictionaries it is the biggest lexical resource in terms of total number of senses provided.
- WordNet (http://www.cogsci.princeton.edu/~wn/) is considered to be the one of the most important resources available to researchers in computational linguistics, text analysis, and many related areas. The main feature of WordNet is that the senses are semantically related each other.
- Babylon's English dictionary (http://www.babylon.com/) is an expansive language resource, comprised of general, encyclopedic, slang and informal terms. It covers a wide range of different professional fields.

We chose these resources because they are freely available on-line, and also because they were suitable for the purposes of our research (all these lexical resources provide for each word a list of its senses and syntactic roles, i.e. part of speech). They are heterogeneous resources, in sense that WordReference and Babylon are lexical dictionaries and WordNet is commonly defined a thesaurus.

The following table reports the total number of senses in each dictionary.

**Table 1. Dimensions of the lexical resources.**

| **WordReference** (headwords) | **WordNet** (strings) | **Babylon** (definitions) |
|---|---|---|
| 180.000 | 144.309 | 138.814 |

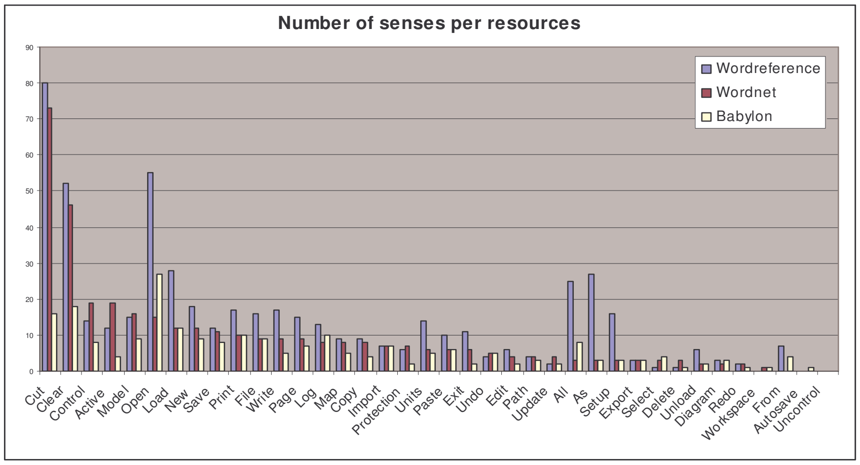
The resources are difficult to compare because senses are defined differently for each resource.

**Case Study – Command Menu**

The first experiment had the goal to consider the problem of measuring ambiguity for single words. As an example for this task we considered a program menu (the idea was suggested in [8]). We chose a common CASE tool: Rational Rose.

We analyzed in particular the first two sections of menu: "File" and "Modify", that are similar in many other software applications. For each command menu item, we retrieved the number of senses, and the number of syntactic roles referring to three chosen dictionaries.

The number of senses defined for each word by the corresponding lexical resource strongly influences ambiguity measure (see table 1 in appendix). As we can see from the graph 1, the word "cut" is the most polysemous in WordNet and WordReference, but in Babylon the most ambiguous of the given items is "open".



**Graph 1. Number of senses for menu items in three dictionaries.**

To evaluate average ambiguity of each word we calculated also weighted average ambiguity (weighted on the dimension and the number of senses in 3 dictionaries):

$$WA_i = \frac{\sum_{k=1}^{3} n_i^{\,k} \times D^k}{\sum_{k=1}^{3} D^k}$$ where *i=1...number of* items;

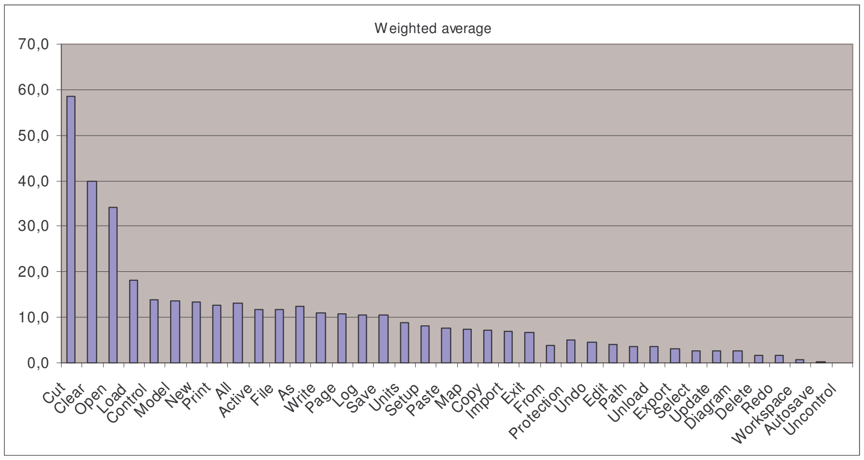
$WA_i$ – weighted average ambiguity of item *i*;

$n_i^{\,k}$ – number of senses for item *i* in dictionary *k;*

$D^k$ – dimension of dictionary *k;*

$K$ – number of the dictionary (in our case we have 3 different dictionaries).

Since we cannot compare the dictionaries on other characteristic except dimension, the weighted average is a possible way to homogenize the differences between them. The weighted average ambiguity values of menu items are represented in graph 2.



**Graph 2. Weighted average ambiguity.**

To determine if there is any relation between the resources we calculated the correlation coefficient[1] as reported in the following table:

**Table 2. Correlations between lexical resources.**

| WordNet-WordRefence | WordNet-Babylon | WordRefence-Babylon |
|---|---|---|
| 0,85 | 0,63 | 0,83 |

All the three values are quite high, which means that there is a high correlation between resources. The higher correlation coefficient between WordReference and the other two resources can be explained by the bigger dimension of WordReference dictionary comparing with two other lexical resources.

These results are the first approximation because they are computed basing on a small set of items from the command menu. In order to have more accurate estimation for correlation coefficient it is necessary to use a larger corpus.

We can observe from graphs 1 and 2 that both diagrams depict the same trend. This can be explained, in fact, also by the high correlation between the lexical resources.

**Case Study – Requirements Documents**

In the second experiment we considered the problem of calculating the semantic ambiguity measure for complete sentence. In literature, there is no standard function to calculate this measure. Therefore, as a first approximation to semantic ambiguity, we extended the notion of lexical ambiguity from the word level to the sentence level.

For the sentence ambiguity we also introduced weighted measure which is calculated taking into account the dimension of each considered dictionary.

$$WSA = \frac{\sum_{k=1}^{3} SA^{k} \times D^{k}}{\sum_{k=1}^{3} D^{k}}$$

*WSA* – weighted sentence ambiguity;
$D^k$ – dimension of dictionary *k;*
$SA^k$ – sentence ambiguity in dictionary *k;*
*k* – number of the dictionary (in our case we have 3 different dictionaries).

[1] The correlation coefficient provides a measure of linear association between variables. It is comprised by -1 and 1 where 1 represent maximum linear association and 0 mean no correlation.

In this experiment we considered two texts which are SRSs (see fig.1). For each word we retrieved the number of its senses and then we applied the ambiguity measures described previously. We calculated also the average ambiguity of the sentences within the given text. As we can evince from charts, the ambiguity measure calculated basing on WordReference resource was the highest.

We summarized our calculations in the tables 3 and 4 for the multiplication type of function (the results for summarization function are similar). The details of calculations are given in Appendix (see tables 7-15 and graphs 3-14).

**Table 3. Library ($SA_{prod}$).**

| Phrase | Word per sentence | Stop Word | Word Reference | Word Net | Babylon | Avg | Wavg |
|---|---|---|---|---|---|---|---|
| Phrase 10 | 19 | 6 | 63.8 | 33.0 | 28.0 | 41.6 | 43.5 |
| Phrase 8 | 17 | 6 | 62.0 | 33.2 | 24.2 | 39.8 | 41.7 |
| Phrase 6 | 15 | 4 | 50.7 | 35.6 | 32.3 | 39.5 | 40.5 |
| Phrase 7 | 21 | 9 | 58.9 | 23.7 | 28.2 | 36.9 | 38.7 |
| Phrase 9 | 15 | 4 | 51.6 | 24.3 | 28.1 | 34.7 | 36.1 |
| Phrase 14 | 13 | 2 | 45.6 | 26.1 | 27.6 | 33.1 | 34.1 |
| Phrase 1 | 11 | 4 | 39.8 | 27.6 | 17.9 | 28.4 | 29.4 |
| Phrase 4 | 11 | 5 | 41.6 | 19.8 | 18.5 | 26.6 | 27.8 |
| Phrase 3 | 10 | 3 | 34.5 | 20.5 | 16.3 | 23.8 | 24.7 |
| Phrase 16 | 10 | 3 | 31.2 | 18.8 | 18.7 | 22.9 | 23.6 |
| phrase 11 | 10 | 3 | 34.6 | 18.6 | 14.5 | 22.5 | 23.6 |
| phrase 13 | 12 | 6 | 40.6 | 13.1 | 12.1 | 21.9 | 23.5 |
| phrase 5 | 7 | 1 | 23.5 | 16.7 | 16.0 | 18.7 | 19.1 |
| phrase 2 | 8 | 3 | 28.6 | 13.1 | 11.8 | 17.8 | 18.7 |
| phrase 15 | 7 | 1 | 22.4 | 14.5 | 12.0 | 16.3 | 16.8 |
| phrase 12 | 8 | 1 | 16.3 | 12.5 | 9.2 | 12.7 | 13.0 |
| average | 12.1 | 3.8 | 40.4 | 21.9 | 19.7 | 27.3 | 28.4 |

**Table 4. SoftCom ($SA_{prod}$).**

| Phrase | Word per sentence | Stop Word | Word Reference | Word Net | Babylon | Avg | Wavg |
|---|---|---|---|---|---|---|---|
| phrase 1 | 20 | 5 | 57.4 | 32.2 | 41.5 | 43.7 | 44.8 |
| phrase 9 | 18 | 4 | 57.5 | 35.8 | 35.5 | 42.9 | 44.2 |
| phrase 10 | 15 | 5 | 49.9 | 30.4 | 27.2 | 35.8 | 37.0 |
| phrase 17 | 12 | 3 | 43.0 | 30.9 | 24.3 | 32.7 | 33.6 |
| phrase 7 | 12 | 3 | 36.9 | 26.8 | 22.6 | 28.8 | 29.5 |
| phrase 6 | 12 | 3 | 37.5 | 22.2 | 26.3 | 28.7 | 29.4 |
| phrase 15 | 10 | 4 | 37.2 | 18.7 | 21.8 | 25.9 | 26.8 |
| phrase 18 | 8 | 2 | 32.9 | 24.1 | 18.4 | 25.1 | 25.8 |
| phrase 14 | 10 | 4 | 34.7 | 19.3 | 17.5 | 23.8 | 24.7 |
| phrase 4 | 8 | 3 | 32.4 | 19.1 | 20.7 | 24.1 | 24.7 |
| phrase 2 | 9 | 2 | 30.9 | 18.6 | 21.7 | 23.8 | 24.3 |
| phrase 12 | 9 | 2 | 27.8 | 16.9 | 20.8 | 21.8 | 22.3 |
| phrase 16 | 10 | 4 | 27.5 | 13.1 | 15.5 | 18.7 | 19.4 |
| phrase 11 | 7 | 2 | 25.0 | 16.8 | 13.7 | 18.5 | 19.1 |
| phrase 3 | 6 | 2 | 25.2 | 15.7 | 11.9 | 17.6 | 18.2 |
| phrase 5 | 9 | 4 | 26.8 | 9.8 | 14.2 | 16.9 | 17.7 |
| phrase 13 | 6 | 2 | 19.1 | 12.2 | 9.7 | 13.7 | 14.1 |
| phrase 8 | 5 | 0 | 12.1 | 8.2 | 6.2 | 8.8 | 9.1 |
| average | 10.3 | 3 | 34.1 | 20.6 | 20.5 | 25.1 | 25.8 |

We can observe from the tables that the average ambiguity of both texts is quite the same. It is not possible to say if this value is low or high, it only means that the texts have the similar style of writing and average length of the sentences.

Our approximate functions provide a kind of upper level, rough ambiguity measure, a more precise ambiguity could be calculated taking into account the part of speech of the words. The suggested functions represent very similar results, therefore they can be used equivalently. The one advantage of choosing specially summarization is that it is less computationally expensive.

From the calculated results we can see the existence of dependencies between such parameters as dimension of the dictionary, number of words, their lexical ambiguity and the value of ambiguity function. Hence, the initial hypothesis of our experiments on the general view of semantic ambiguity function was confirmed.

Moreover, it was shown that in both texts there is a strong correlation between the number of words in the sentence and overall ambiguity value, as well as between the number of all words and the number of stop-words (see tables 5 and 6).

**Table 5. Correlation in Library text.**

| Num words – weighted average | Num words –number of stop words |
|---|---|
| 0.92 | 0.85 |

**Table 6. Correlation in SoftCom text.**

| Num words – weighted average | Num words –number of stop words |
|---|---|
| 0.95 | 0.76 |

In fact, the dependency between input parameters can be very complicated, and the function can take into account not only the number of senses of each word in sentence, but also syntactic roles of the words and grammatical agreement between them. This way the number of possible senses of particular word can be reduced. Also the so called stop-words, as prepositions, articles, even pronouns, can be discarded from the consideration (as it is done in WordNet).

## 6. Ambiguity Identification Prototype

In this section we present the prototype developed pursuing the aim to provide a tool for supporting the writing of the high quality documents.

### 6.1 Tool Requirements

Firstly, we describe the requirements to the ambiguity identification and measurement tool, its desired features and how the users are supposed to interact with its functionality. In this version of the prototype not all of the requirements listed here have been implemented. This set of requirements is in some sense ‘the higher bound’ for the ambiguity identification and measurement system that we would like to have at disposal. In the present work we tried to provide at least the initial framework of the interactive tool supporting ambiguity identification.

The central part of the interface is to be text area where the author types documents in natural language. Such text area has to support all basic text processing aids which help in loading, saving and retrieving a document in the file system. Some advanced text facing features must be supported as well (such as, changing the font types and sizes, the ability to use italics and bold face, the ability to give a structure to a document using titles, sections and paragraphs).

The tool starts text processing with parsing of the input text, then the ambiguities must be identified. Since in our approach we take into account ambiguity of single words and of the whole sentence, ambiguity identification and notification of the user should be done in two different phases. The first notification can performed as soon as a word has been typed, i.e. “on-line”, in the case if the word is known as highly ambiguous. Of course, the writer cannot always avoid using some common words, but their meaning should be restricted for each particular application (for example, adding these words to a local dictionary; then if meeting these words again the tool should automatically refer to the meaning in this dictionary). One of the most important features of the tool is that this notification is done in real time, during the typing. “On-line” notification has to be evident and clear but not too heavy, because it should not interrupt the author during her/his writing. For example a small pop-up window or a sound are considered disturbing notification, because they attract the author attention too much and they can make her/him forget the main idea she/he is going to write down. A suitable notification mechanism can be done by using a different color or by underlining the very ambiguous word, so it becomes evident in the text and the typing, and thinking work is not stopped.

The second phase notification can be done off-line; it means that it does not happen as soon as the text is typed, but after a sentence is completed or when the user explicitly requires it by selecting the appropriate command from the application menu. In this phase the tool highlights the sentence ambiguity, by using different color for the background of the text. A white background means that a sentence has a low ambiguity degree, so it is easily understandable and it requires no changes. A light grey background indicates that the ambiguity degree for the sentence is medium, because its meaning is not so obscure, but it can be much improved with a minimum amount of work, for example by changing one or two words. In the last case, a dark grey background denotes that a sentence is very ambiguous, thus some changes are required in order to bring it in an appropriate comprehensible form.

In our tool we do not fix the value of threshold for ambiguity identification and the choice of threshold is given to user. This is mainly motivated by the reason that it is complex to choose the level of ambiguity which must be signaled to the user. The threshold may differ depending on the goals of particular task, document writing policy or domain of the document.

We suggest for future development that approximate level of the threshold can be estimated from the sample of a document. This document can be considered as the most unambiguous. The tool must analyze the sample document, calculate its average ambiguity measures, and suggest them to the user as default threshold values.

In order to increase the comprehensibility of a sentence and upgrade its color level, it is important to focus the effort on the most ambiguous worlds in a sentence, since the ambiguity of a sentence is influenced by the ambiguity of each word composing it. A good starting point for this change task can be all the words from the "black list" detected in the first phase. Once this first word set has been used up, the tool must help the user by suggesting the next ones to be worked on.

When the author works out the comprehensibility of his/her text, in the worst case he/she will need to change the whole sentence and reformulate it in a different way. In the other cases the work is concentrated on changing the single words, inserting more appropriate ones. In such situations the tool should help the author providing her/him a list of candidates for the change, thus speeding up this phase.

This tool is intended to facilitate writing textual documents. We expect that even if on the first usages the tool many notifications for required changes comes up in the text, as the users become more and more expert, the amount of changes required next time would become smaller and smaller, because the authors would learn how to develop high quality documents.

## 6.2 Prototype Description

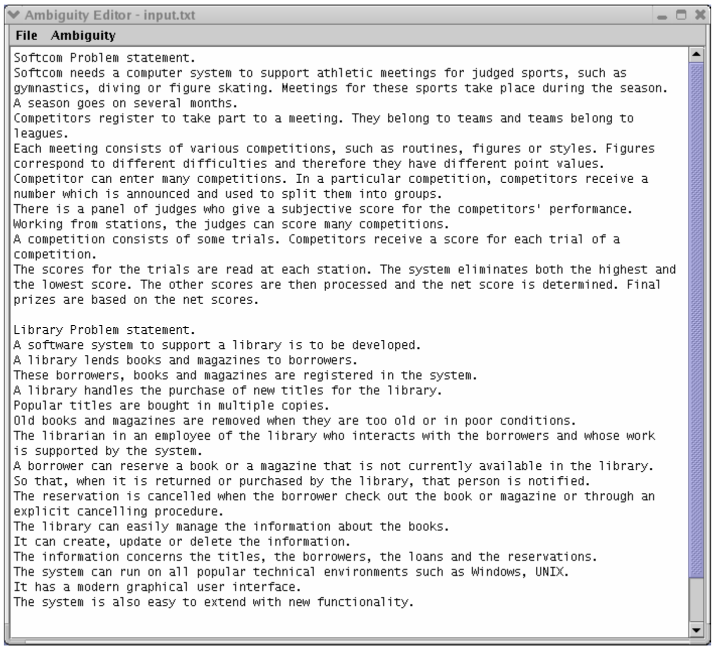
The prototype was implemented in Java programming language. As a lexical resource we used WordNet because it is freely available machine readable dictionary which also provides the source of its knowledge base. The tool is intended to support writing qualitative documents.

For analysis of input text our tool uses simply a tokenization function (it can be integrated in future with other tools that perform more complex text parsing). So far, the prototype is able to calculate ambiguity of the sentences basing on the number of senses of the words, as they are given in WordNet. The prototype can also rank the sentences accordingly to the level of ambiguity.

The prototype developed in our project can be further extended by adding more sophisticated techniques for calculating ambiguity measures. Also additional functionality for interactive work with the documents can be built into the tool. However, the features to be added will depend on the domain and task and must be adapted while porting the tool to the particular application.

## 6.3 Case Study

We demonstrate the work of prototype on example of the following use case. First of all, the user has to open an input file (so far, the tool supports only plain text as an input format).

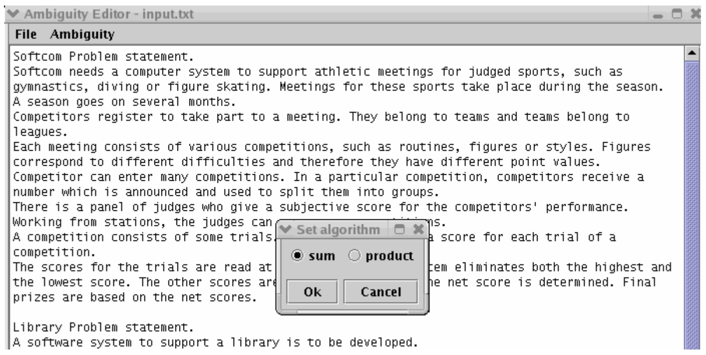


```
Softcom Problem statement.
Softcom needs a computer system to support athletic meetings for judged sports, such as
gymnastics, diving or figure skating. Meetings for these sports take place during the season.
A season goes on several months.
Competitors register to take part to a meeting. They belong to teams and teams belong to
leagues.
Each meeting consists of various competitions, such as routines, figures or styles. Figures
correspond to different difficulties and therefore they have different point values.
Competitor can enter many competitions. In a particular competition, competitors receive a
number which is announced and used to split them into groups.
There is a panel of judges who give a subjective score for the competitors' performance.
Working from stations, the judges can score many competitions.
A competition consists of some trials. Competitors receive a score for each trial of a
competition.
The scores for the trials are read at each station. The system eliminates both the highest and
the lowest score. The other scores are then processed and the net score is determined. Final
prizes are based on the net scores.

Library Problem statement.
A software system to support a library is to be developed.
A library lends books and magazines to borrowers.
These borrowers, books and magazines are registered in the system.
A library handles the purchase of new titles for the library.
Popular titles are bought in multiple copies.
Old books and magazines are removed when they are too old or in poor conditions.
The librarian in an employee of the library who interacts with the borrowers and whose work
is supported by the system.
A borrower can reserve a book or a magazine that is not currently available in the library.
So that, when it is returned or purchased by the library, that person is notified.
The reservation is cancelled when the borrower check out the book or magazine or through an
explicit cancelling procedure.
The library can easily manage the information about the books.
It can create, update or delete the information.
The information concerns the titles, the borrowers, the loans and the reservations.
The system can run on all popular technical environments such as Windows, UNIX.
It has a modern graphical user interface.
The system is also easy to extend with new functionality.
```

**Fig. 1.** Loaded input text.

The next step is to choose the function which defines how the sentence ambiguity must be calculated (so far, the choice is between sum and logarithm of the product; see section 4).

The number of the senses for each word is taken from the WordNet database index. The tool addresses to the database at the run-time. The user can recalculate measures anytime if the input text is changed.



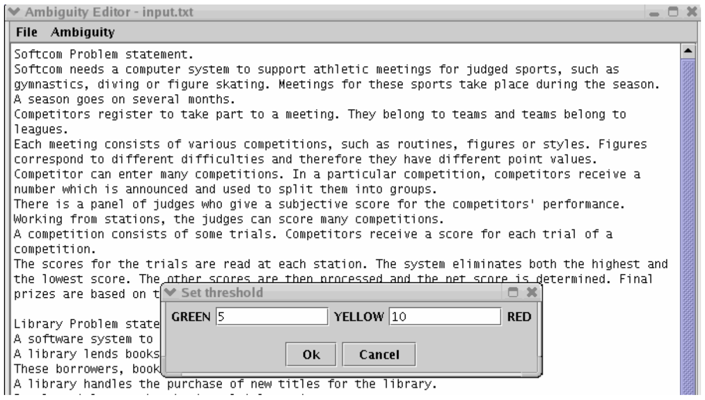
```
Softcom Problem statement.
Softcom needs a computer system to support athletic meetings for judged sports, such as
gymnastics, diving or figure skating. Meetings for these sports take place during the season.
A season goes on several months.
Competitors register to take part to a meeting. They belong to teams and teams belong to
leagues.
Each meeting consists of various competitions, such as routines, figures or styles. Figures
correspond to different difficulties and therefore they have different point values.
Competitor can enter many competitions. In a particular competition, competitors receive a
number which is announced and used to split them into groups.
There is a panel of judges who give a subjective score for the competitors' performance.
Working from stations, the judges can                    ns.
A competition consists of some trials.                   a score for each trial of a
competition.
The scores for the trials are read at                    tem eliminates both the highest and
the lowest score. The other scores are                   he net score is determined. Final
prizes are based on the net scores.

Library Problem statement.
A software system to support a library is to be developed.
```

[✓ sum] product

**Fig. 2.** Choosing the function for calculating sentence ambiguity.

Then the user is provided the facility to tune the thresholds for ambiguity identification algorithm. That means how the highlighting schema distinguishes between the levels of low, medium and highly ambiguous sentences.

Setup of the thresholds is not well-elaborated feature of the tool so far, because it is difficult to propose the exact values. We suggest that by default the thresholds should be set up calculating the ambiguity values for all the sentences and dividing them into 3 different groups with low, medium and high ambiguity level.



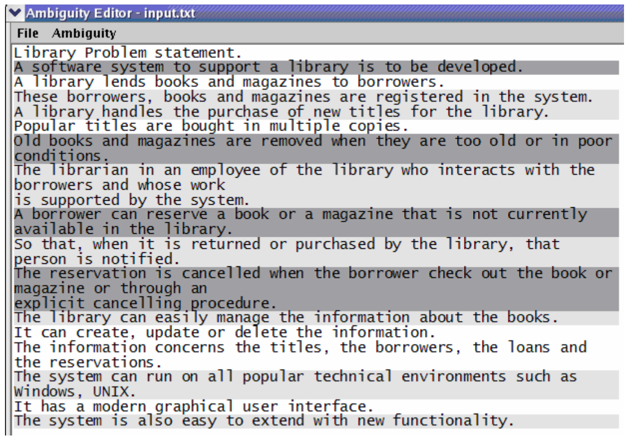
```
Softcom Problem statement.
Softcom needs a computer system to support athletic meetings for judged sports, such as
gymnastics, diving or figure skating. Meetings for these sports take place during the season.
A season goes on several months.
Competitors register to take part to a meeting. They belong to teams and teams belong to
leagues.
Each meeting consists of various competitions, such as routines, figures or styles. Figures
correspond to different difficulties and therefore they have different point values.
Competitor can enter many competitions. In a particular competition, competitors receive a
number which is announced and used to split them into groups.
There is a panel of judges who give a subjective score for the competitors' performance.
Working from stations, the judges can score many competitions.
A competition consists of some trials. Competitors receive a score for each trial of a
competition.
The scores for the trials are read at each station. The system eliminates both the highest and
the lowest score. The other scores are then processed and the net score is determined. Final
prizes are based on t

Library Problem state
A software system to
A library lends books
These borrowers, book
A library handles the purchase of new titles for the library.
```

GREEN 5 YELLOW 10 RED

**Fig. 3.** Setting threshold values.

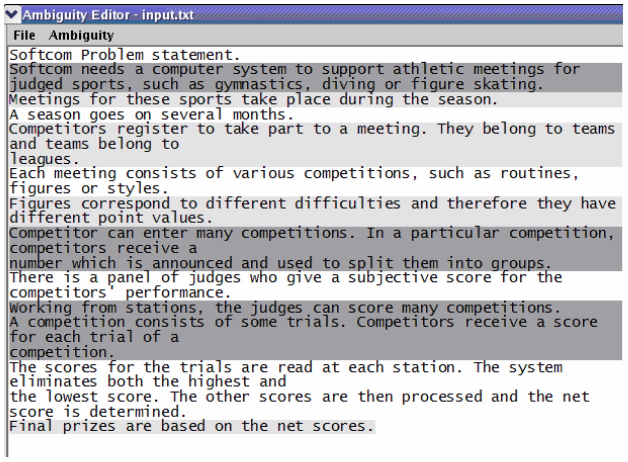
The final output is the text where the sentences are highlighted accordingly to the detected values of ambiguity.

A white (uncolored) background means that a sentence has a low ambiguity degree; a light grey background indicates that the ambiguity degree for the sentence is medium; a dark grey background denotes that a sentence is highly ambiguous and some changes are required in order to bring it in an appropriate form.



```
Library Problem statement.
A software system to support a library is to be developed.
A library lends books and magazines to borrowers.
These borrowers, books and magazines are registered in the system.
A library handles the purchase of new titles for the library.
Popular titles are bought in multiple copies.
Old books and magazines are removed when they are too old or in poor
conditions.
The librarian in an employee of the library who interacts with the
borrowers and whose work
is supported by the system.
A borrower can reserve a book or a magazine that is not currently
available in the library.
So that, when it is returned or purchased by the library, that
person is notified.
The reservation is cancelled when the borrower check out the book or
magazine or through an
explicit cancelling procedure.
The library can easily manage the information about the books.
It can create, update or delete the information.
The information concerns the titles, the borrowers, the loans and
the reservations.
The system can run on all popular technical environments such as
Windows, UNIX.
It has a modern graphical user interface.
The system is also easy to extend with new functionality.
```

**Fig. 4.** Computing ambiguity of sentences and highlighting them (Library text).



```
Softcom Problem statement.
Softcom needs a computer system to support athletic meetings for
judged sports, such as gymnastics, diving or figure skating.
Meetings for these sports take place during the season.
A season goes on several months.
Competitors register to take part to a meeting. They belong to teams
and teams belong to
leagues.
Each meeting consists of various competitions, such as routines,
figures or styles.
Figures correspond to different difficulties and therefore they have
different point values.
Competitor can enter many competitions. In a particular competition,
competitors receive a
number which is announced and used to split them into groups.
There is a panel of judges who give a subjective score for the
competitors' performance.
Working from stations, the judges can score many competitions.
A competition consists of some trials. Competitors receive a score
for each trial of a
competition.
The scores for the trials are read at each station. The system
eliminates both the highest and
the lowest score. The other scores are then processed and the net
score is determined.
Final prizes are based on the net scores.
```

**Fig. 5.** Computing ambiguity of sentences and highlighting them (Softcom text).

## 7. Conclusions

Identification of ambiguities is an important task for evaluating the text quality and uncovering its vulnerable points. The present work reviews existing approaches to ambiguity identification and investigates the problems of ambiguity measures calculation. We introduced functions which are the approximate measures of sentence ambiguity.

Moreover, we developed and presented the prototype of a tool for ambiguity identification and measurement in natural language text. The tool is intended to support the process of writing qualitative documents.

Summarizing the results of our research, we would like to highlight the following problems that appeared to be crucial in ambiguity identification task:

- It is not possible to equally identify ambiguity using different lexical resources, because the number of senses depends on the dimension and domain coverage of the dictionary.
- In the literature there exist no standard measures to identify ambiguities of words, sentences or texts. In order to devise these measures, further investigations are required.
- The choice of lexical resource plays important role in measuring the ambiguity. The dictionary must be machine readable and easily accessible to be used by automatic tool.

As we can see, there are still open issues left. They can be the subject for the future development In conclusion, we would like to emphasize the importance of ambiguity identification problem, because undetected ambiguities cause different people to act different ways in response to the text, each according to his interpretation of each ambiguity.

## References

[1] Berry D. M., Kamsties E., Kay D. M., Krieger M. M. *From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity*, A Handbook, University of Waterloo, Waterloo, Ontario, Canada, 2003, http://se.uwaterloo.ca/~dberry/#Handbook.

[2] Chantree F. *Ambiguity Management in Natural Language Generation*, 7th Annual CLUK Research Colloquium, January 2004.

[3] Fabbrini F., Fusani M., Gnesi S., Lami G. The Linguistic Approach to the Natural Language Requirements, Quality: Benefits of the use of an Automatic Tool, In *Proc. of the 26th Annual IEEE Computer Society - NASA GSFC Software Engineering Workshop*, Greenbelt, MA, November 27-29 2001, pp: 97-105.

[4] Guiasu S. Information theory with applications, McGRAW-HILL, 1977.

[5] Ide N., Véronis J. Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24:1, pp. 1-40, 1998.

[6] Kamsties E., Berry D., Paech B. Detecting Ambiguities in Requirements Documents Using Inspections, In *Proc. of the 1st Workshop on Inspection in Software Engineering (WISE'01)*, M. Lawford, D. L. Parnas (eds.), Paris, France, 23 July 2001, pp. 68-80.

[7] Mich L. On the use of ambiguity measures in requirements analysis, In *Proc. of NLDB'01*, A. Moreno, R. Van de Riet (eds.), Madrid, 28-29 June 2001, pp. 143-152.

[8] Mich L., Garigliano R. Ambiguity Measures in Requirement Engineering. In *Proc. of International Conference on Software - Theory and Practice - ICS2000*, 16th IFIP World Computer Congress, Beijing, China, 21-25 August 2000, Y. Feng, D. Notkin, M. Gaudel (eds.), Publishing House of Electronics Industry: Beijing, pp. 39-48, 2000.

[9] Mitamura T. Controlled Language for Multilingual Machine Translation, In *Proc. of Machine Translation Summit VII,* Singapore, 1999.

[10]Wilson W.M., Rosenberg L.H., Hyatt L.E. Automated analysis of requirement specifications, In *Proc. of the 19th Int. Conf. on Software Engineering (ICSE-97)*, Boston, Massachusetts, US, May 17-23, ACM Press, New York, NY, USA, 1997, pp. 161 – 171

[11]Yarowsky D. Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora, In *Proc. of COLING-92*, Nantes, pp. 454-460, 1992.

## Appendix

**Table 7. Menu items (number of senses and syntactic roles)**

| Word | WordNet | | WordReference | | Babylon | | Weighted average |
|---|---|---|---|---|---|---|---|
| | Senses | Syn. roles | Senses | Syn. roles | Senses | Syn. roles | |
| *File* | 9 | 2 | 16 | 2 | 9 | 2 | 11.7 |
| *New* | 12 | 2 | 18 | 2 | 9 | 2 | 13.4 |
| *Open* | 15 | 2 | 55 | 3 | 27 | 3 | 34.1 |
| *Save* | 11 | 2 | 12 | 4 | 8 | 3 | 10.5 |
| *Autosave* | NP* | NP | NP | NP | 1 | 1 | 0.3 |
| *As* | 3 | 2 | 27 | 5 | 3 | 3 | 12.3 |
| *Log* | 8 | 2 | 13 | 1 | 10 | 2 | 10.5 |
| *Clear* | 46 | 4 | 52 | 4 | 18 | 4 | 39.9 |
| *Load* | 12 | 2 | 28 | 2 | 12 | 2 | 18.2 |
| *Model* | 16 | 3 | 15 | 2 | 9 | 3 | 13.5 |
| *Workspace* | 1 | 1 | NP | NP | 1 | 1 | 0.6 |
| *Units* | 6 | 1 | 14 | 1 | 5 | 1 | 8.8 |
| *Unload* | 2 | 1 | 6 | 1 | 2 | 1 | 3.6 |
| *Control* | 19 | 2 | 14 | 2 | 8 | 2 | 13.8 |
| *Uncontrol* | NP | NP | NP | NP | NP | NP | 0 |
| *Write* | 9 | 1 | 17 | 1 | 5 | 1 | 10.9 |
| *Protection* | 7 | 1 | 6 | 1 | 2 | 1 | 5.1 |
| *Import* | 7 | 2 | 7 | 2 | 7 | 2 | 7.0 |
| *Export* | 3 | 2 | 3 | 2 | 3 | 2 | 3.0 |
| *Update* | 4 | 2 | 2 | 2 | 2 | 2 | 2.6 |
| *Print* | 10 | 2 | 17 | 2 | 10 | 2 | 12.7 |
| *Page* | 9 | 2 | 15 | 2 | 7 | 2 | 10.7 |
| *Setup* | 3 | 1 | 16 | 3 | 3 | 1 | 8.1 |
| *Edit* | 4 | 1 | 6 | 2 | 2 | 1 | 4.2 |
| *Path* | 4 | 1 | 4 | 1 | 3 | 1 | 3.7 |
| *Map* | 8 | 2 | 9 | 2 | 5 | 2 | 7.5 |
| *Exit* | 6 | 2 | 11 | 2 | 2 | 2 | 6.7 |
| *Undo* | 5 | 1 | 4 | 1 | 5 | 2 | 4.6 |
| *Redo* | 2 | 1 | 2 | 1 | 1 | 1 | 1.7 |
| *Cut* | 73 | 3 | 80 | 3 | 16 | 3 | 58.6 |
| *Copy* | 8 | 2 | 9 | 2 | 4 | 2 | 7.2 |
| *Active* | 19 | 2 | 12 | 2 | 4 | 2 | 11.8 |
| *Diagram* | 2 | 2 | 3 | 2 | 3 | 2 | 2.7 |
| *Paste* | 6 | 2 | 10 | 2 | 6 | 2 | 7.6 |
| *Delete* | 3 | 1 | 1 | 1 | 1 | 1 | 1.6 |
| *Select* | 3 | 2 | 1 | 1 | 4 | 2 | 2.5 |
| *All* | 3 | 2 | 25 | 3 | 8 | 2 | 13.0 |
| *From* | NP | NP | 7 | 1 | 4 | 1 | 3.9 |

* NP: word is not given in dictionary

**Table 8. Library text ($SA_{prod}$).**

| Phrases | WordNet | WordReference | Babylon |
|---|---|---|---|
| *phrase 1* | 27.6 | 39.8 | 17.9 |
| *phrase 2* | 13.1 | 28.6 | 11.8 |
| *phrase 3* | 20.5 | 34.5 | 16.3 |
| *phrase 4* | 19.8 | 41.6 | 18.5 |
| *phrase 5* | 16.7 | 23.5 | 16.0 |
| *phrase 6* | 35.6 | 50.7 | 32.3 |
| *phrase 7* | 23.7 | 58.9 | 28.2 |
| *phrase 8* | 33.2 | 62.0 | 24.2 |
| *phrase 9* | 24.3 | 51.6 | 28.1 |
| *phrase 10* | 33.0 | 63.8 | 28.0 |
| *phrase 11* | 18.6 | 34.6 | 14.5 |
| *phrase 12* | 12.5 | 16.3 | 9.2 |
| *phrase 13* | 13.1 | 40.6 | 12.1 |
| *phrase 14* | 26.1 | 45.6 | 27.6 |
| *phrase 15* | 14.5 | 22.4 | 12.0 |
| *phrase 16* | 18.8 | 31.2 | 18.7 |
| *Average Text Ambiguity* | 21.9 | 40.4 | 19.7 |

**Table 9. SoftCom text ($SA_{prod}$).**

| Phrases | WordNet | WordReference | Babylon |
|---|---|---|---|
| *phrase 1* | 32.2 | 57.4 | 41.5 |
| *phrase 2* | 18.6 | 30.9 | 21.7 |
| *phrase 3* | 15.7 | 25.2 | 11.9 |
| *phrase 4* | 19.1 | 32.4 | 20.7 |
| *phrase 5* | 9.8 | 26.8 | 14.2 |
| *phrase 6* | 22.2 | 37.5 | 26.3 |
| *phrase 7* | 26.8 | 36.9 | 22.6 |
| *phrase 8* | 8.2 | 12.1 | 6.2 |
| *phrase 9* | 35.8 | 57.5 | 35.5 |
| *phrase 10* | 30.4 | 49.9 | 27.2 |
| *phrase 11* | 16.8 | 25.0 | 13.7 |
| *phrase 12* | 16.9 | 27.8 | 20.8 |
| *phrase 13* | 12.2 | 19.1 | 9.7 |
| *phrase 14* | 19.3 | 34.7 | 17.5 |
| *phrase 15* | 18.7 | 37.2 | 21.8 |
| *phrase 16* | 13.1 | 27.5 | 15.5 |
| *phrase 17* | 30.9 | 43.0 | 24.3 |
| *phrase 18* | 24.1 | 32.9 | 18.4 |
| *Average Text Ambiguity* | 20.6 | 34.1 | 20.5 |

**Table 10. Library text ($SA_{sum}$).**

| Phrases | WordReference | Babylon | WordNet |
|---|---|---|---|
| *phrase 1* | 177 | 45 | 102 |
| *phrase 2* | 119 | 29 | 38 |
| *phrase 3* | 144 | 46 | 72 |
| *phrase 4* | 174 | 49 | 57 |
| *phrase 5* | 85 | 44 | 50 |
| *phrase 6* | 200 | 87 | 113 |
| *phrase 7* | 222 | 84 | 114 |
| *phrase 8* | 289 | 68 | 99 |
| *phrase 9* | 201 | 72 | 92 |
| *phrase 10* | 260 | 87 | 128 |
| *phrase 11* | 128 | 35 | 52 |
| *phrase 12* | 48 | 21 | 30 |
| *phrase 13* | 134 | 32 | 41 |
| *phrase 14* | 244 | 75 | 108 |
| *phrase 15* | 102 | 28 | 44 |
| *phrase 16* | 115 | 49 | 75 |
| *Average Text Ambiguity* | 165.1 | 53.2 | 75.9 |

**Table 11. SoftCom text ($SA_{sum}$).**

| Phrases | WordReference | Babylon | WordNet |
|---|---|---|---|
| *phrase 1* | 246 | 115 | 100 |
| *phrase 2* | 214 | 78 | 100 |
| *phrase 3* | 171 | 37 | 59 |
| *phrase 4* | 218 | 75 | 95 |
| *phrase 5* | 75 | 33 | 24 |
| *phrase 6* | 142 | 71 | 62 |
| *phrase 7* | 178 | 67 | 109 |
| *phrase 8* | 38 | 14 | 23 |
| *phrase 9* | 254 | 116 | 125 |
| *phrase 10* | 242 | 73 | 123 |
| *phrase 11* | 107 | 35 | 43 |
| *phrase 12* | 107 | 66 | 51 |
| *phrase 13* | 76 | 27 | 29 |
| *phrase 14* | 181 | 51 | 61 |
| *phrase 15* | 161 | 61 | 63 |
| *phrase 16* | 111 | 47 | 45 |
| *phrase 17* | 184 | 70 | 110 |
| *phrase 18* | 158 | 50 | 92 |
| *Average Text Ambiguity* | 159.1 | 60.3 | 73.0 |

**Table 12. Library text ($SA_{prod}$, average and weighted average).**

| Phrase | Average | Weighted Average |
|---|---|---|
| *phrase 1* | 28.4 | 29.4 |
| *phrase 2* | 17.8 | 18.7 |
| *phrase 3* | 23.8 | 24.7 |
| *phrase 4* | 26.6 | 27.8 |
| *phrase 5* | 18.7 | 19.1 |
| *phrase 6* | 39.5 | 40.5 |
| *phrase 7* | 36.9 | 38.7 |
| *phrase 8* | 39.8 | 41.7 |
| *phrase 9* | 34.7 | 36.1 |
| *phrase 10* | 41.6 | 43.5 |
| *phrase 11* | 22.5 | 23.6 |
| *phrase 12* | 12.7 | 13.0 |
| *phrase 13* | 21.9 | 23.5 |
| *phrase 14* | 33.1 | 34.1 |
| *phrase 15* | 16.3 | 16.8 |
| *phrase 16* | 22.9 | 23.6 |
| *Overall Average Text Ambiguity* | 27.3 | 28.4 |

**Table 13. Softcom text ($SA_{prod}$, average and weighted average).**

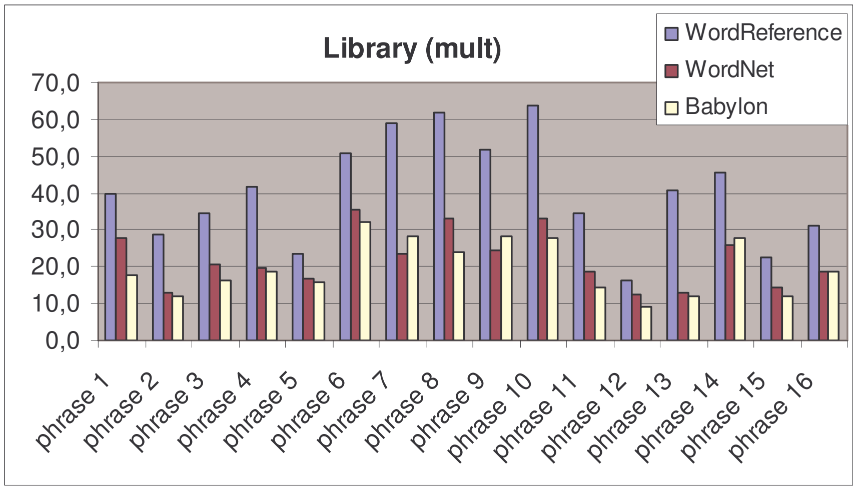
| Phrase | Average | Weighted Average |
|---|---|---|
| *phrase 1* | 43.7 | 44.8 |
| *phrase 2* | 23.8 | 24.3 |
| *phrase 3* | 17.6 | 18.2 |
| *phrase 4* | 24.1 | 24.7 |
| *phrase 5* | 16.9 | 17.7 |
| *phrase 6* | 28.7 | 29.4 |
| *phrase 7* | 28.8 | 29.5 |
| *phrase 8* | 8.8 | 9.1 |
| *phrase 9* | 42.9 | 44.2 |
| *phrase 10* | 35.8 | 37.0 |
| *phrase 11* | 18.5 | 19.1 |
| *phrase 12* | 21.8 | 22.3 |
| *phrase 13* | 13.7 | 14.1 |
| *phrase 14* | 23.8 | 24.7 |
| *phrase 15* | 25.9 | 26.8 |
| *phrase 16* | 18.7 | 19.4 |
| *phrase 17* | 32.7 | 33.6 |
| *phrase 18* | 25.1 | 25.8 |
| *Overall Average Text Ambiguity* | 25.1 | 25.8 |

**Table 14. Library text ($SA_{sum}$, average and weighted average).**

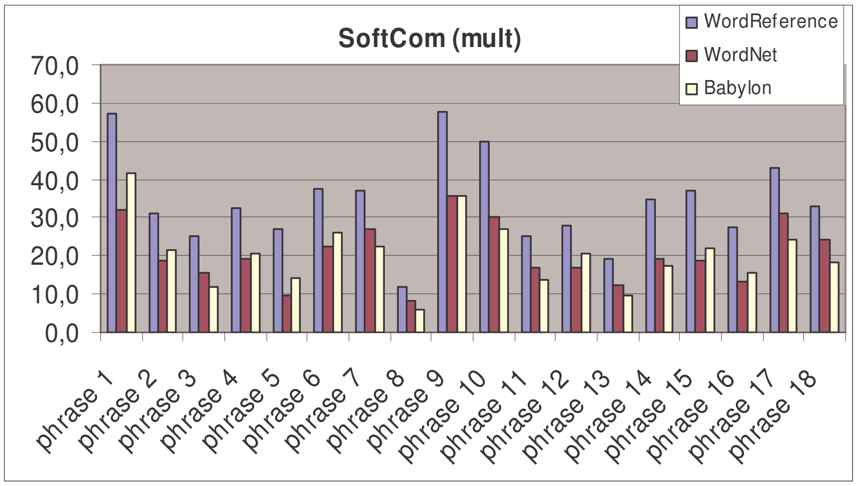
| Phrase | Average | Weighted Average |
|---|---|---|
| *phrase 1* | 108 | 114.1 |
| *phrase 2* | 62 | 66.8 |
| *phrase 3* | 87.3 | 92.2 |
| *phrase 4* | 93.3 | 100.1 |
| *phrase 5* | 59.7 | 61.8 |
| *phrase 6* | 133.3 | 139.0 |
| *phrase 7* | 140.0 | 147.0 |
| *phrase 8* | 152 | 163.6 |
| *phrase 9* | 121.7 | 128.4 |
| *phrase 10* | 158.3 | 167.0 |
| *phrase 11* | 71.7 | 76.4 |
| *phrase 12* | 33.0 | 34.3 |
| *phrase 13* | 69.0 | 74.4 |
| *phrase 14* | 142.3 | 151.0 |
| *phrase 15* | 58.0 | 61.7 |
| *phrase 16* | 79.7 | 82.8 |
| *Overall Average Text Ambiguity* | 98.1 | 103.8 |

**Table 15. Softcom text ($SA_{sum}$, average and weighted average).**

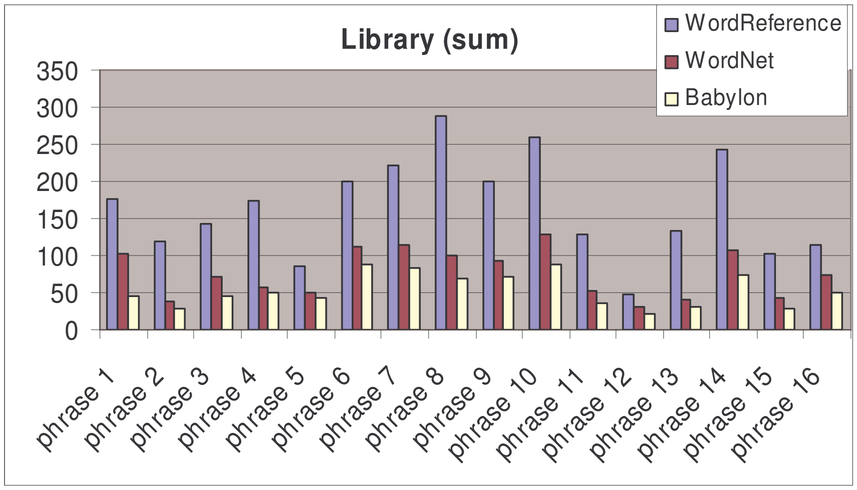
| Phrase | Average | Weighted Average |
|---|---|---|
| *phrase 1* | 153.7 | 161.2 |
| *phrase 2* | 130.7 | 137.7 |
| *phrase 3* | 89.0 | 95.9 |
| *phrase 4* | 129.3 | 136.8 |
| *phrase 5* | 44.0 | 46.5 |
| *phrase 6* | 91.7 | 95.8 |
| *phrase 7* | 118.0 | 123.2 |
| *phrase 8* | 25.0 | 26.1 |
| *phrase 9* | 165.0 | 172.4 |
| *phrase 10* | 146.0 | 154.3 |
| *phrase 11* | 61.7 | 65.5 |
| *phrase 12* | 74.7 | 77.3 |
| *phrase 13* | 44.0 | 46.7 |
| *phrase 14* | 97.7 | 104.6 |
| *phrase 15* | 95.0 | 100.5 |
| *phrase 16* | 67.7 | 71.3 |
| *phrase 17* | 121.3 | 126.8 |
| *phrase 18* | 100.0 | 105.1 |
| *Overall Average Text Ambiguity* | 97.5 | 102.7 |



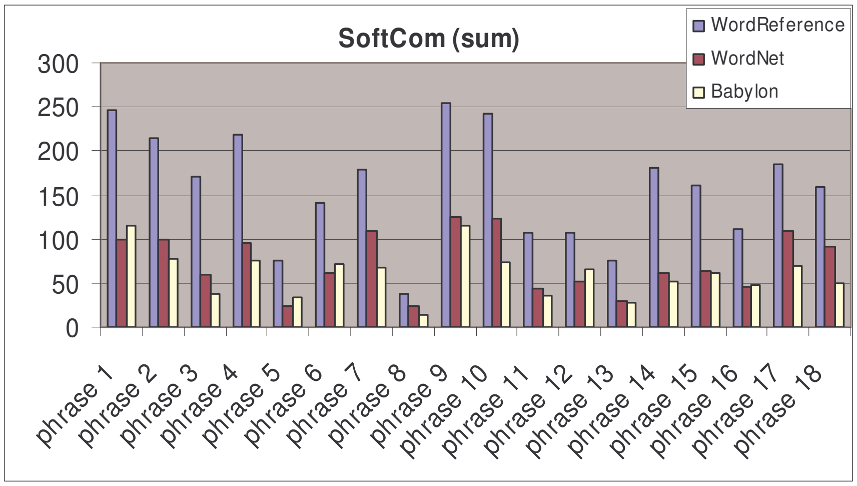
**Graph 3. Library text ($SA_{prod}$).**



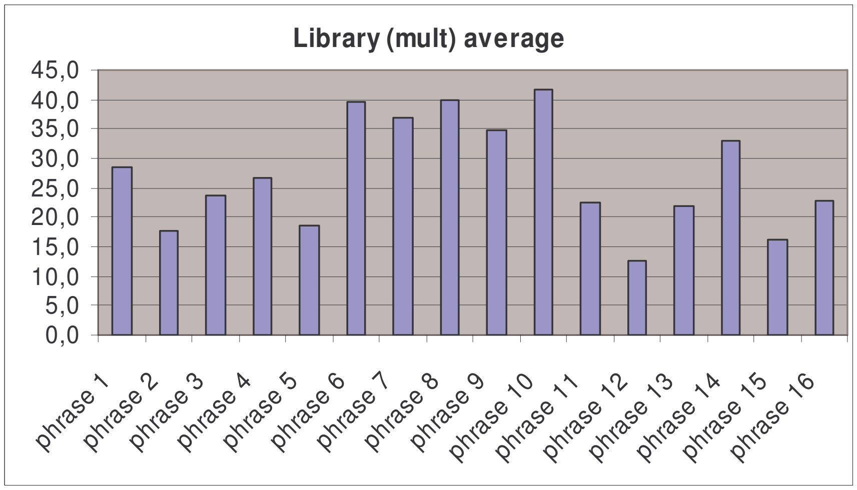
**Graph 4. Softcom text ($SA_{prod}$).**



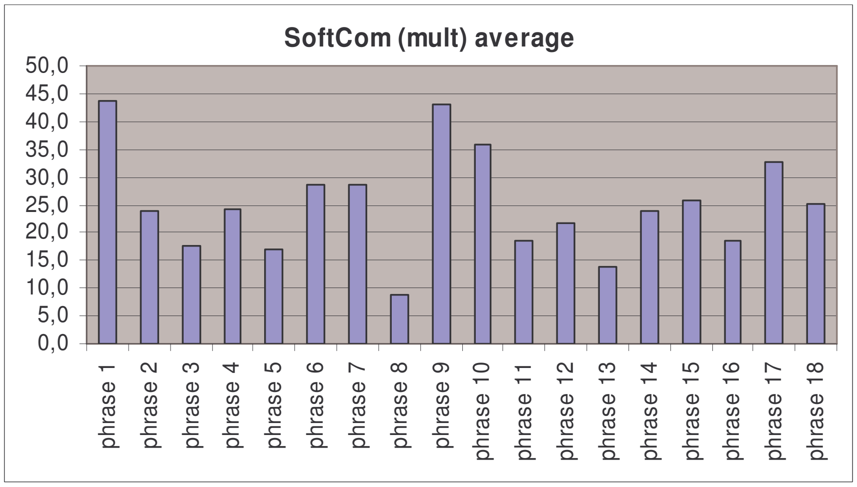
**Graph 5. Library text ($SA_{sum}$).**



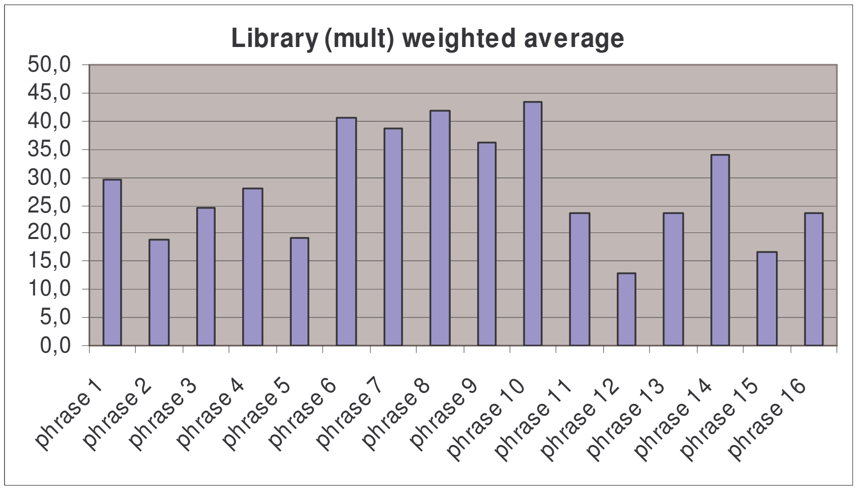
**Graph 6. SoftCom text ($SA_{sum}$).**



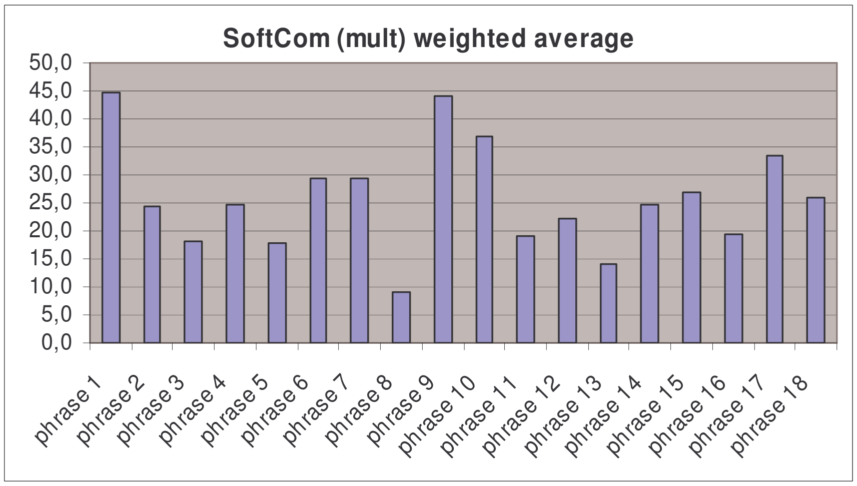
**Graph 7. Library text ($SA_{prod}$ average).**



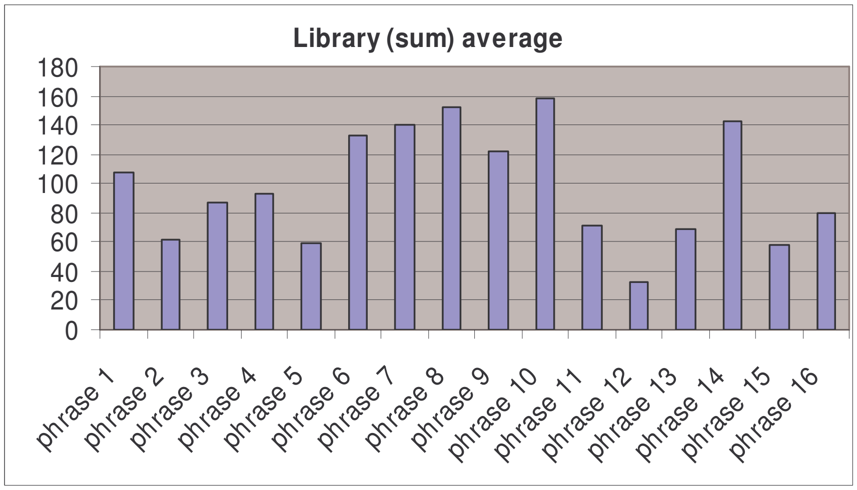
**Graph 8. Softcom text ($SA_{prod}$ average).**



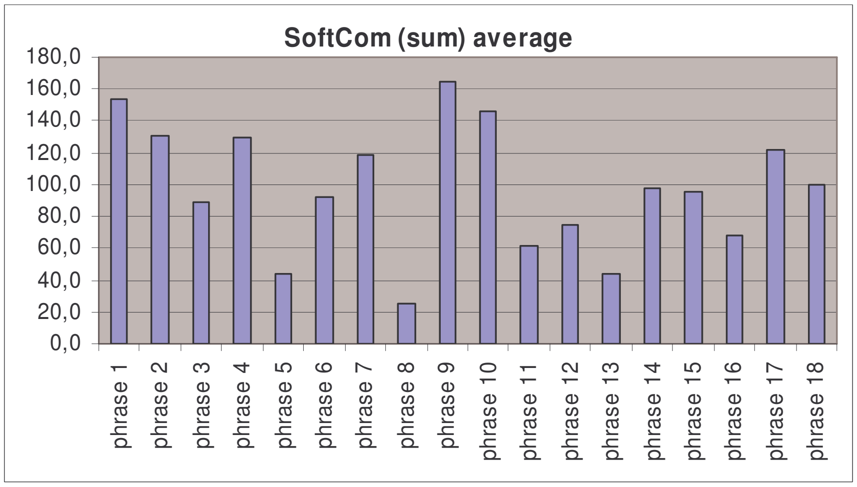
**Graph 9. Library text ($SA_{prod}$ weighted average).**



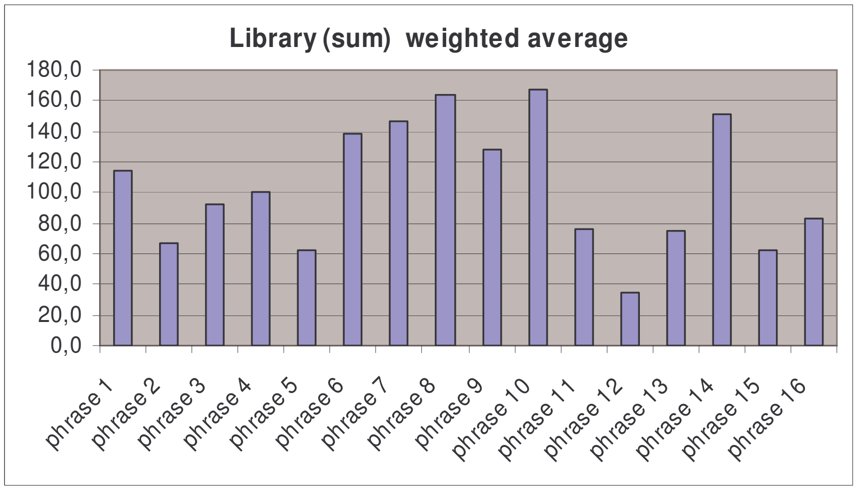
**Graph 10. Softcom text ($SA_{prod}$ weighted average).**



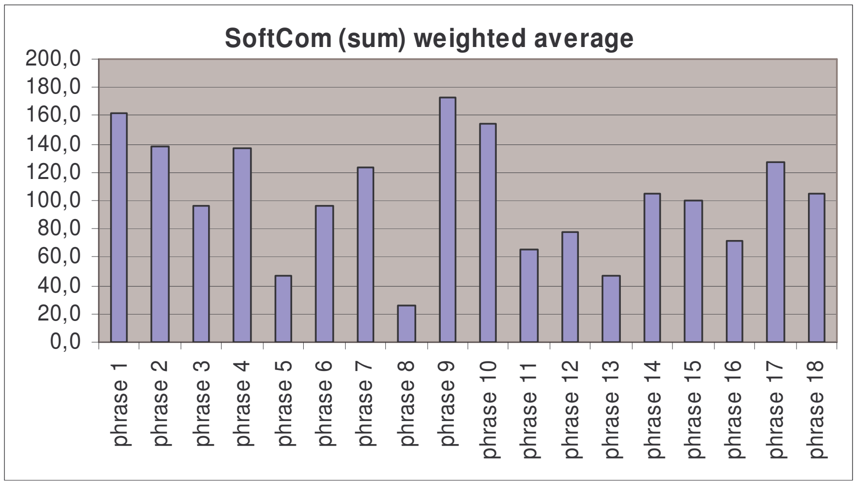
**Graph 11. Library text ($SA_{sum}$ average).**



**Graph 12. Softcom text ($SA_{sum}$ average).**



**Graph 13. Library text sum weighted average.**



**Graph 14. Softcom text ($SA_{sum}$ weighted average).**