# Collocations Computed from the Web

**Tomasz Adam Maksymilian WYSOCKI**
Engineering Physics, Lund Institute of Technology
Ehrensvärdsg. 22, 212 13 Malmö, Sweden
pogodno@gmx.de

## Abstract

This paper describes a prototype system implemented for verifying the correctness of all verb-preposition-collocations found in a given text. The verification is done using statistics from the world's largest corpus - the Internet. The tool used for obtaining these statistics is the Google Web APIs service[1]. The probability of correctness is computed according to the concepts of proportional score, t-score and mutual information.

## 1 Introduction

### 1.1 Preface

Almost each and every one of us has tried to learn a foreign language at some point in our lifes, being more or less successful in doing so. The obstacles that one has to conquer in the process of learning a natural language are more or less complex, depending on ones mother tongue, learning ability, other languages of which one already has command etc. What should a person do when exposed to a new, unfamiliar expression? Trying to express something in German, linguistic rules valid for English could always be applied. A word-for-word translation could also be tried. The attained result might be acceptable, but in most cases it would not. Regarding fixed expressions, or collocations, one can almost be sure of the non-feasibility of a literal translation. Collocations have often a very long history and can be very specific for the given language. They are linguistic entities that one has to learn by heart, or continue to be ignorant. In this paper I concentrate on the collocations of a verb and a preposition, e.g. *"depend on"*. The following sections describe the prototype of an application helping the user in checking if such a collocation was written correctly or not. The lexicon used for verification is the Internet and the tool employed is Googles Web APIs service. As measures for the probability of correctness, the concepts of proportional score, t-score and mutual information are applied.

In the following, whenever the word *collocation* is stated, it will have the meaning of *verb-preposition-collocation* only.

### 1.2 Background

In order to verify the correctness of a collocation, one could ask a native speaker of that language, but after a while he would get tired of being asked these things over and over again. Instead, one might try consulting a dictionary, or checking the frequency of the collocation in a large text corpus. This last alternative is what the application implemented in this project makes use of. The corpus used for verification is the largest cohesive and searchable existing for the time being - the Internet. Using any search engine on the Web, one can look up the collocation in doubt, and check the amount of hits generated. Then the process could be repeated, substituting another preposition for the original one. Applying the concept of proportional score, the conclusion would then be that the proper preposition to be used in that collocation is the one generating most hits. In the next section this very method, together with two others, will be described.

Google, known for its popular search engine, provides a service for software developers, allowing the applications of each of them to state 1000 automated queries per day. This service offers many of the options found in the original Web-based search engine of Google's. In this project, the Google

[1] Google Web APIs service.

Web APIs service is used constructing a java-application for checking the collocations in a text.

## 2 Approach

### 2.1 System

The sentences one would like to verify should be saved in a text file and tagged using a part-of-speech tagger (e.g. the MXPOS Tagger[2], as done in this project). The input fed to the java-program implemented in this project is a tagged text file. The application recognizes as verbs all the words labeled with either of the tags _VBZ, _VBN, _VBG, _VBP and as prepositions all the words labeled with the tag _IN (these are the tags used by the MXPOST). The probability for each collocation found is then calculated using one of the three scoring methods described below and the figures obtained using Google's Web APIs service.

The result is then sent to the standard output. The printout presents the probability for the collocation with the original preposition. Additionally, if there is a preposition giving higher probability, it is suggested as the more proper one. Otherwise, the second best preposition is presented along with the original one.

### 2.2 Obtaining Probabilities

When calculating the probability of a collocation (e.g. *"depends on"*) being correct, the following figures have to be known:

*c(v)*, the amount of hits for the verb *"v"* ( *"depends"*);

*c($p_i$)*, the amount of hits for the preposition *"$p_i$"* ( *"on"*);

*c(v, $p_i$)*, the amount of hits for the expression *"v $p_i$"* (*"depends on"*).

The collocation (consisting of the verb $v$ and the preposition $p_i$) considered to be the most correct one is the one having the largest probability $\pi_i$, which is defined as

$$\pi_i = \frac{score(v,p_i)}{\sum_j score(v,p_j)}$$

where

$$\sum_i \pi_i = 1$$

The list of prepositions considered by the application is

$$\begin{aligned} L \quad = \quad & \{as, at, by, for, from, in, of \\ & off, on, than, to, upon, with\} \end{aligned}$$

Figures for each of those prepositions are obtained just once, in one of the initial steps of the algorithm.

The number $N$, representing the total number of words in English texts on the Internet is roughly estimated to equal around 11 billion. This figure is obtained by checking the frequency of such words as "*in*", "*on*" and "*of*" in a large, fixed-size text corpus, then checking the amount of those words on the Internet, multiplying the both figures for each word, and taking the average among all of them.

### 2.3 Scoring Methods

The application uses one of the three most common scoring methods for calculating the probabilities.

#### 2.3.1 Proportional Score

The proportional score is the same as the amount of hits obtained for the collocation, that

$$score_p(v, p_i) = c(v, p_i)$$

#### 2.3.2 T-Score

The t-score is defined by

$$score_t(v, p_i) = \frac{c(v,p_i) - \frac{1}{N}c(v)c(p_i)}{\sqrt{c(v,p_i)}}$$

The t-score shows in what extent the association between two words v and p is non-random. In case of a high t-score, the result can be assumed to be quite confident.

[2]Maximum Entropy Part-Of-Speech Tagger.

### 2.3.3 Mutual Information

The mutual information is defined by

$$score_t(v, p_i) = \log_2 N \frac{c(v,p_i)}{c(v)c(p_i)}$$

The mutual information puts the probability of observing the collocation *"v p"* in comparison with the probabilities of observing $v$ and $p$ independently. This implies that if the collocation occurs often compared with the occurrence of the words $v$ and $p$, it should be considered as probable.

## 3 Examples

The following sentences are examples of those used for verification of the system.

1. *The weather depends on the climate.*
2. *The rate depends of the initial values.*
3. *The health of children depends at least partially on their access to health services.*
4. *Lato was substituted for Maradona.*
5. *The luxurios champagne was substituted with less expensive, but even more sofisticated bavarian wheat beer.*

## 4 Results

The final results obtained for the sentences from the previous section were the same for each method, although the probabilities differed widely.

1. *The weather depends on the climate.*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 77 | 48 | 27 |

Second best: *depends upon*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 16 | 22 | 23 |

2. *The rate depends of the initial values.*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 1 | 4 | 8 |

Suggestion: *depends on*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 76 | 49 | 27 |

3. *The health of children depends at least partially on their access to health services.*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 0 | 1 | 2 |

Suggestion: *depends on*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 76 | 48 | 27 |

4. *Lato was substituted for Maradona.*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 50 | 29 | 18 |

Second best: *substituted by*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 18 | 18 | 15 |

5. *The luxurios champagne was substituted with less expensive, but even more sofisticated bavarian wheat beer.*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 12 | 14 | 14 |

Suggestion: *substituted by*

| Method | c | t | m |
|---|---|---|---|
| Probability (%) | 49 | 29 | 18 |

The most time-consuming part of the program is the communication with Google. If this procedure could be made faster, it would actually be feasible to include this feature into some word-processing software in order to make it easier to process a larger mass of text.

## 5 Discussion

As seen in the previous section, all the three methods delivered exactly the same results. The single issue that differs between them

is the confidence in the result being unquestionably correct.

Using the proportional score, the result space contains in each case a single significant peak, making it easy to distinguish the correct (according to this method) preposition.

The mutual information delivers several peaks having similar values, the highest of them still being the correct solution.

The level of confidence of the t-score method lies somewhere in-between the two others.

The third sentence is actually erroneous, although the system finds it to be correct. This is due to the fact that the sentence consists of a principal clause (*The health of children depends on their access to health services.*) and a subordinate clause (*at least partially*) and the commas are left out in the source file. Even if the commas would be there, Google does not make any difference between "*depends at*" and "*depends, at*", thus the results would be identical.

It is free for anyone to construct a webpage of his own, in the language and with the contents of his choice. Due to this nature, the Internet contains a lot of noise. This contributes to the fact that much of the virtual substance, the collocations in general, contains grammatical errors.

The errors observed in the contents of a webpage are correlated with the native language of its author. It his highly probable, that the members of a language group make the same mistakes, applying for instance the concept of literal translation of expressions and collocations. If the language group is large, the texts produced by its members could cause significant noise, as it is correlated. The largest noise peaks will probably origin from such large languages as Spanish, French and Chinese, whereas the noise produced by members of minor language groups would not be significant, although noticeable. Since the mass of English text constructed by its native speakers is much larger than the single native groups', this noise should not reach the amplitude of the real, correct signal.

The situation would look somewhat different if our language of interest would be other than English. A small language is always strongly exposed to such noise, because one false entry makes a large contribution to the total corpus in this tongue.

There are always errors made by the natives as well, but this should not be correlated enough to give peak noise, it would rather be an amplification of the basic noise level. If an error convicted by the native speakers would become significant, it should rather be considered as synonym with the original one rather than incorrect.

The system is partly optimized in order to delimit the processing time and the amount of requests to Google. The check of the amount of hits for the single prepositions is done only once, at the startup of the system. Still, one run is performed for the verb and the collocation for each sentence, even if they have been checked before. The system could be modified to remember previous searches in order to save time and enquiry-credits. Such a procedure would though increase both the memory space needed and the internal running time of the system. However, the large limitation of the costly server connection-time would cause the net save in running time to be positive.

The system is easily extended to cover other scoring methods. Constructing a subclass to the already written one and adding the desired methods is all that is needed.

Since Google's search service is not limited to searches in the English language only, the system could easily be modified in order to handle almost any other tongue. The issue of tagging the text remains, though part-of-speech taggers for several different languages are available on the Internet itself. The tags indicating the verbs and the prepositions would have to be the same as used by the MXPOS tagger, otherwise the method extracting the collocations would have to be overwritten.

## 6 Conclusions

As we could see, the system arrives at the correct conclusions for each of the examples used for verification. Although, the example space is small and extended testing is required in order to obtain the confidence-statistics for the system.

It is also necessary to develop a model, which would combine the results yielded by all the methods. For example, in case of the three methods having the same outcome (as in the examples presented in section 5), the delivered answer could be considered to be certain.

The essential conclusion is that, using the Internet as language source and Google's APIs service as searching tool, it is possible to construct an effective linguistic assistant, not only for English, but also for almost any language. Such an application must also not be limited to handle mere verb-preposition-collocations, but also other expressions, or even spell checking. To make such a system practical, one would have to incorporate the system into word-processing software. This would make it much easier and faster to access. To increase the speed even further, the server connection time would have to be delimited in order to yield a feasible running time. This could be solved by keeping a local record storing previous searches. It would also be necessary to higher the limit of searches allowed, as a standard-size text would require many more than 1000 queries.

## 7 References

Google Web APIs service, 2004, http://www.google.com/apis/

Maximum Entropy Part-Of-Speech Tagger (MXPOST), 2004, http://www.cis.upenn.edu/~adwait/statnlp.html

NuguesPierre, Language Processing and Computational Linguistics, Lecture Notes, LTH, 2004