Elaborating Formulae for Testing Word Similarity in Inflective Languages

Xavier Blanco,[1] Mikhail Alexandrov,[1,2] and Alexander Gelbukh[2]
[1] Department of French and Romance Philology, Autonomous University of Barcelona, Spain
[2] Center for Computing Research (CIC), National Polytechnic Institute (IPN), DF 07738 Mexico
Xavier.Blanco@uab.es; dyner1950@mail.ru, dyner@cic.ipn.mx; www.Gelbukh.com

Abstract

We consider similar words as words having the same base meaning (*sad, sadness, sadly*, etc.). Identification of such words is an important procedure for many IR applications, especially for constructing word frequency lists. Testing word similarity for multi-language and multi-thematic document collection can be implemented using empirical formulae trained on a small number of examples. Unlike rule-based methods, this approach does not need any linguistic resources such as morphological dictionaries or morphological rules. Unlike the statistical-based methods, it does not use probabilistic properties of letters and their combinations learnt from large corpora. One only needs to adjust slightly the formula's parameters to the given language, topic, and the desired balance of false positives vs. false negatives. In our previous works we have demonstrated how to construct such formulae of a given class using Ivahnenko's evolutionary algorithm. In this paper, we discuss the choice of the class of functions. The approach can be applied for the languages where the word base (the morphologically invariant part) is located at the beginning (or at the end) of the word, as in almost all European languages (and not in the middle or otherwise, as, say, in Arabic).

1 Introduction

1.1 The Problem

We consider the problem of determining the similarity for a given pair of words. By similarity, we mean the equality of the base meaning. Examples are English words *sad, sadness, sadly*. This task is crucial in information retrieval and in constructing word frequency lists. We consider the languages where the word base (the morphologically invariant part) is located at the beginning of the word, and not, say, in the middle or otherwise (as in Arabic). It is generally true for the majority of European languages.

We elaborate several empirical formulae for making a decision about word similarity using any two of the following three characteristics of the pair of words:

n: total number of *final* letters differing in the two words,
y: length of the common *initial* substring,
s: total number of letters in the two words,

so that $2y + n = s$. E.g., for the words *sadly* and *sadness*, the maximal common initial substring is *sad-*, thus the differing final parts are *-ly* and *-ness*, so that $n = 6$, $y = 3$, and $s = 12$.

A general rule for decision making based on empirical formula can be represented as:

$$F(n,y,s,a_0,a_1,a_2,\ldots,a_n) \le 0 \qquad (1)$$

where F is a model function, n, y, and s are characteristics of the pair of words, and a_0 to a_n are unknown parameters of the formula. When the inequality is correct, the hypothesis about the similarity is accepted. The form of the model formula is supposed to be given. The parameters are determined based on positive examples prepared by an expert. The examples are the word pairs having the same base meaning, see Table 1.

Table 1. Examples for determining the parameters

Examples			Equations
sad	sad*ness*	$n = 4, y = 3, s = 10$	$4/10 = a_0 + a_1 3$
hop*ing*	hope*full*	$n = 8, y = 3, s = 14$	$8/14 = a_0 + a_1 3$

To find the best parameters for the formula, we consider the extreme cases when inequalities (1) are transformed to the equalities. Therefore, we have the system of m equations, each of them being related with one pair of similar words from the given set of m pairs. For example, if the formula (1) has the form $n/s \le a_0 + a_1 y$, then we have the system of equations shown in Table 1.

Such a system can be easily solved by the least squares method. Some requirements for the training examples are:

- They must reflect statistical regularities of the language. So, it is desirable that the examples be selected by a skillful expert;
- The number of equations should be at least 3 times greater than the number of parameters: $m \ge 3(n + 1)$. This is necessary to filter out the noise in the examples.

Since the empirical formula is constructed on the basis of statistical regularities of the language, it leads to false posi-

tives and false negatives. In the former case (false positive), a pair of non-similar words is considered similar; in the latter case (false negative) a pair of similar words is considered non-similar. By tuning the model parameters, we can control the balance between these two kinds of errors, though we cannot completely avoid them.

The main advantage of the method is that it does not require knowing all possible endings and suffixes; i.e., the approach is language independent.

The limitations of the approach are obvious:

- It cannot detect similarity between irregular words, such as *buy* and *bought*;
- It cannot be used directly for non-inflective languages such as Arabic.

In our previous experiments with Romance languages, we have shown that by tuning the model parameters it is possible to obtain the error rate P_{err} of 12% and F-measure of 90% (Alexandrov *et al*, 2004). Since false negatives can be easily corrected manually, such a result is satisfactory for many applications.

In all publications related to the empirical formulae, only polynomial functions of a certain form have been constructed and elaborated. However, selection of these formulae has not been discussed. In this paper, we consider selection of the form of the formula.

1.2 Related Work

There are two main groups of methods for detecting the similarity between words: rule-based and statistical-based methods. The difference between them consists in using or not the knowledge of morphology of the language.

Lemmatization is the strongest method from the former group. It provides 100% accuracy on known words and good accuracy on the words absent in the dictionary (Gelbukh, 2003; Gelbukh and Sidorov, 2003). All words are reduced to the standard dictionary form (singular for nouns, infinitive for verbs, etc.) using a large morphological dictionary and morphological rules.

Stemming is a much less resource-consuming method: it uses lists of suffixes and suffix removal rules, but no dictionaries (Porter, 1980). It provides good accuracy: more than 95% for many languages. All words are truncated to their stems, which usually reflect their invariant meaning.

The methods of the second group use probabilistic distribution of letters and letter combinations on different positions in words of a given language and domain. Words are truncated at the point of the significant difference between real and expected probability of letter occurrence (Avetisian, 1981). For Spanish, this method provides the accuracy of 90% or more. The truncated part usually reflects the invariant meaning of the words.

Both approaches are inconvenient for working with multi-lingual and multi-domain document collections. The methods of the first groups require either large language resources or manual compilation of suffix removal rules, which can be very expensive. The methods of the second group need detailed statistical data, which often are absent.

Makagonov and Alexandrov (2002) suggested an empirical method for testing word similarity and described a methodology for constructing empirical formulae based on the number of the coincident letters in the initial parts of the two words and the number of non-coincident letters in their final parts. This approach was considered in detail in (Alexandrov *et al*, 2004) and was modified for taking into account some peculiarities of Romance languages in (Blanco et al., 2006), where n-gram techniques were used.

It should be emphasized that our goal is not to reveal the language morphology in the form of a set of endings, suffixes, and stems, but only to decide on similarity of a given pair of words. The algorithm for automatic extraction of stems from a large corpus of texts was described in (Goldsmith, 2001). We solved the same problem by using an optimization approach and genetic algorithm (Gelbukh *et al*., 2004). However, these works are not focused on our goal in this paper: to reveal the similarity between a pair of given words.

In Section 2, we briefly review our previous results concerning polynomial model function and Ivahnenko's evolutionary algorithm. In Section 3, we present the results concerning the applications of several other formulae. Section 4 gives the conclusions.

2 Polynomial Formula

2.1 Ivahnenko's Algorithm and Lineal Models

Without any information on the statistical properties of the initial base part and the final (differing) parts of the two words, we can consider the empirical formula in the form

$$\begin{aligned} &n/s \le F(y,a_0,a_1,a_2,\ldots,a_n), \\ &F(y,\ldots) = a_0 + a_1 y + a_2 y + \ldots + a_n y, \end{aligned} \qquad (2)$$

where n, y, and s are the characteristics of the word pair as described in Section 1.1 and F is a model function. Obviously, (2) is a particular case of (1). The following ideas were taken into account for the selection of such a model:

- The formula must be convenient for representation of a wide class of functions;
- The formula must have al least two degrees of liberty. This allows reflecting in the formula separate statistics of the base and the final parts. One can also consider a model in the form $n/s \le F(y/s)$ with only one degree of liberty, since $y/s = ½ (1 – n/s)$; however, the form (2) is more flexible. Our experiments clearly show this.
- The formula must smooth the sharp change of the characteristics of a word pair. One can also consider a model in the form $n \le F(y)$; however, the form (2) gives significantly more correct results.

Our first task was to find the best form of the model function, i.e., for the models (2), including the degree of the polynomial. For this, we used a so-called inductive method of model self-organization (IMMSO) developed by Ivahnenko (1980). It requires a limited number of examples, which are divided into training and control sets. The method evaluates the models of increasing complexity, one by one,

and stops when the optimum of an external criterion of model quality is reached, i.e., when the models stop improving. The given examples are used to calculate the model parameters and the difference between the models using the external criterion. This method cannot find an exactly optimal model in a continuous class because it is based on competition of models; this is why the method is called inductive. For detailed explanation of this method, see (Alexandrov *et al*, 2004).

Table 2 presents the results of application of this method for four Romance languages. All models proved to be lineal. This is very important, since it indicates high complexity of the real phenomenon we are trying to model. Our model can reflect only a tendency (a_1 is the first derivative of the model function) but not the shape of the function. In this case, we cannot expect very good results; success can be reached only for a narrow-domain text corpus.

Table 2. Parameters for different languages

Language	Parameters
French	$n/s \le 0.481 - 0.024\ y$
Italian	$n/s \le 0.571 - 0.035\ y$
Portuguese	$n/s \le 0.528 - 0.029\ y$
Spanish	$n/s \le 0.549 - 0.029\ y$
English	$n/s < 0.551 - 0.032\ y$

We have also constructed empirical formula for English**.** However, we did not try to find the best form of the model function (2) for this case: the obtained results suggested considering only lineal functions. Naturally, this is only a first approximation that must be tuned on a given domain.

2.2 Experimental Results

We tested the formula for Spanish on a real domain-oriented document corpus. Our purpose was to evaluate the sensibility of the formula to the changes of its parameters. For the experiment, we used the documents on mortgage and crediting. This theme is narrow enough to provide a representative set of similar words. We took six articles containing in total about 22,000 words excluding numbers and words with less than 4 letters. To reduce the number of comparisons, we randomly selected some paragraphs from the mentioned document collection. Since we assumed that similar words had different final parts, it was natural to order all words alphabetically and to compare the neighbors. This is not the best way for grouping similar words; we used it only to simplify manual testing of word pairs. As a result, we obtained an alphabetically ordered list of words. With 536 words, we made 535 manual comparisons of the adjacent words and detected 165 similar word pairs and 370 non-similar ones.

Initially, we applied the algorithm for testing word similarity using the basic parameters from Table 1: $a_0 = 0.549$ and $a_1 = 0.029$. Then, we varied the parameters a_0 and a_1 of the model function to 10% to both sides around the central point. The results are presented in Table 3 below. Here, P_p and P_n are the probability of false positive and false negative and R and P are recall and precision. The former values are usually used in mathematical statistics (Cramer, 1946) and the latter in information retrieval (Baeza-Yates and Ribero-Neto, 1999); the relation between them is expressed by the formulae: $R = 1 - P_n$ and $P = R/(R + P_p)$. In our case, we are interested in the total error rate and the ratio between the errors of the two kinds.

The results show that when we change the parameters by 10%, the errors of both kinds change approximately by 1.5 to 2 times (in the opposite directions). As we have mentioned above, the procedure of testing word similarity is used for constructing word frequency lists. In this case, false positives cannot be corrected: the similar words are substituted by their common part. Therefore, it is desirable that the rate of this kind of errors be very low. On the other hand, false negatives can be manually corrected. Therefore, their rate may be higher.

Compare columns 2 and 4 of Table 3. In column 2, we have the minimum of false positives: only 2% of the non-similar words are joined as similar (1 of 50). However, the user here has to join manually as many as 27% of all similar words (1 of 4). In column 4, we have the minimum of the total error rate. In this case, 6% of non-similar words are joined (1 of 17) and only 13% of similar words need manual correction (1 of 8). Therefore, this case is better.

3 Other Formulae

3.1 Pre-processing for Selection of the Formula

The choice of polynomial model function is rather arbitrary. To evaluate such a choice, we will plot the dependence to be modeled in the graphical form using given examples, look at the curve, and select an appropriate model function.

If the plot is very close to horizontal or vertical line or looks very fuzzy, then such dependence is not fit for model description. In the former case, it does not reflect changes in the model parameters; in the latter case, the model would be very inaccurate.

Figure 1 presents 20 examples (pairs of similar words) in the coordinate plane (y, n/s). Some points are coincident. One can see that the dependence is clear and can be described by a lineal function $F = a_0 + a_1 y$ or exponential function $F = a_0 \exp(a_1 y)$. Both approximations constructed by the least squares method are presented in Figure 2.

We considered two other empirical formulae: $2y/s \ge a_1 + a_2 n$ and $n \le a_0 + a_1 y$. For this, we represented the examples on the coordinate planes (n, $2y/s$) and (y, n); see Figures 3 and 4, respectively. One can see that the dependences are fuzzy and close to a horizontal line. So, the results cannot be expected to be very good.

3.2 Experimental Results

Our experiments consisted of two steps. First, we constructed the empirical formulae using the least squares method and given examples; the results are presented in Table 7 below. Then, we applied the obtained formulae to the same data that we used for constructing the polynomial model. The results are presented in Tables 4–6, respectively.

Table 3. Experimental results for the original polynomial formula

$n/s \le a_0 + a_1 y$

Parameters		0.55, –0.029	0.49, –0.029	0.61, –0.029	0.55, –0.026	0.55, –0.032
Similar cases		164	128	183	167	142
Non-similar cases		371	407	352	368	393
False alarms		22	8	34	23	14
Omissions		23	45	16	21	37
False positive	P_p	5.9%	2.2%	9.2%	**6.2%**	3.8%
False negative	P_n	13.9%	27.3%	9.7%	**12.7%**	22.4%
Total error	P_{err}	19.8%	29.5%	18.9%	**18.9%**	26.2%
Recall	R	86.1%	72.7%	90.3%	87.3%	77.6%
Precision	P	93.6%	97.1%	90.8%	93.4%	95.3%
F-measure	F	89.7%	83.1%	**90.5%**	90.2%	85.6%

Summary: Min P_{err} = 18.9%, Max F = 90.5%

Table 4. Experimental results for the formula for the plane (*y, n/s*)

$n/s \le a_0 \exp(a_1 y)$

Parameters		0.61, –0.09	0.55, –0.09	0.67, –0.09	0.61, –0.08	0.61, –0.1
Similar cases		150	136	188	165	146
Not similar		385	399	347	370	389
False alarm		18	13	19	23	17
Omission		31	40	29	21	33
False positive	P_p	4.9%	3.5%	5.1%	**6.2%**	4.6%
False negative	P_n	18.8%	24.2%	17.6%	**12.7%**	19.4%
Total error	P_{err}	23.7%	27.7%	22.7%	**18.9%**	24.0%
Recall	R	81.2%	75.8%	82.4%	87.3%	80.6%
Precision	P	94.3%	95.6%	94.1%	93.4%	94.6%
F-measure	F	87.3%	84.5%	87.9%	**90.2%**	87.0%

Summary: Min P_{err} = 18.9%; Max F = 90.2%

Table 5. Experimental results for the formula for the plane (*n, y/s*)

$y/s \ge a_1 + a_2 n$

Parameters		0.73, –0.013	0.67, –0.013	0.79, –0.013	0.61, –0.08	0.61, –0.1
Similar cases		120	153	96	119	126
Not similar		415	382	439	416	409
False alarm		4	16	0	5	6
Omission		49	26	68	49	46
False positive	P_p	1.1%	**4.3%**	0.0%	1.4%	1.6%
False negative	P_n	29.7%	**15.8%**	41.2%	29.7%	27.9%
Total error	P_{err}	30.8%	**20.1%**	41.2%	31.1%	29.5%
Recall	R	70.3%	84.2%	58.8%	70.3%	72.1%
Precision	P	98.5%	95.1%	100.0%	98.1%	97.8%
F-measure	F	82.0%	**89.4%**	74.0%	81.9%	83.0%

Summary: Min P_{err} = 20.1%; Max F = 89.4%

Table 6. Experimental results for the formula for the plane (*y, n*)

$n \le a_0 + a_1 y$

Parameters		5.54, 0.10	4.99, 0.10	6.09, 0.10	5.54, 0.09	5.54, 0.11
Similar cases		120	151	185	154	157
Not similar		415	384	350	381	378
False alarm		4	24	43	24	25
Omission		49	37	24	35	32
False positive	P_p	1.1%	6.5%	**11.6%**	6.5%	6.8%
False negative	P_n	29.7%	22.4%	**14.5%**	21.2%	19.4%
Total error	P_{err}	30.8%	28.9%	**26.2%**	27.7%	26.2%
Recall	R	70.3%	77.6%	85.5%	78.8%	80.6%
Precision	P	98.5%	92.3%	88.0%	92.4%	92.3%
F-measure	F	82.0%	84.3%	**86.7%**	85.0%	86.0%

Summary: Min P_{err} = 26.2%; Max F = 86.7%

Table 7. Formulae for different planes

Plane	Formula
(y, n/s)	$n/s \leq 0.614 \exp(-0.090\, y)$
(n, y/s)	$y/s \geq 0.729 - 0.013\, n$
(y, n)	$n \leq 5.54 + 0.104\, y$

It can be seen that the model $n/s \leq a_0\, exp(a_1\, y)$ gives the same results as the lineal model $n/s \leq a_0 + a_1\, y$ considered in Section 2.2. This result followed from Figure 2 and indicates high complexity of the phenomenon we intent to model: our model reflects only a tendency but not the specific shape. The model $y/s \geq a_1 + a_2\, n$ gives good result only for one combination of parameters and significantly worse results at the all other combinations. This means that this model is very unstable and depends very much on the data.

The model $n \leq a_0 + a_1\, y$ is the worst one. Indeed, it is too sensitive to the parameters of the pair of words, which change in a discrete scale. The relations n/s or y/s used in other formulas compensate for this effect; see Section 2.1.

3 Conclusions

We have proposed a simple approach for choice of formulae for testing word similarity, based on visual analysis of the presented examples. We tested our proposal for several formulae on a real domain oriented document collection. It has been shown that successful formula can provide 5%–7% of false positives and 12%–17% of false negatives. This is rather acceptable in semi-automatic setting since a human expert can easily join the similar words after the automatic grouping procedure.

Our experimental results show that a good formula should reflect common-sense considerations: say, a forrmula such as y^n cannot be expected to give good results. The formula should reflect the linguistic regularities of the language:

– fewer differing letters at the end of words increase the probability of similarity;
– the endings have similar length independently from the length of the base.

In the future, we plan to construct formulae for other inflective languages.

Acknowledgments

The work was supported partially supported to the third author by Mexican Government (CONACyT, SIP-IPN). We thank Prof. M. Porter for useful suggestions related to application of the formula.

References

[Avetisian, 1981] D. Avetisian. *Problems of Information Retrieval (effectiveness, automatic coding, strategies of search)*. "Finance and Statistics" Publ., Moscow, 1981 (in Russian).

[Alexandrov *et al*, 2004] M. Alexandrov, X. Blanco, P. Makagonov. Testing Word Similarity: Language Independent Approach with Examples from Romance. *Lecture Notes in Computer Science* 3136, Springer, pp. 223–234, 2004.

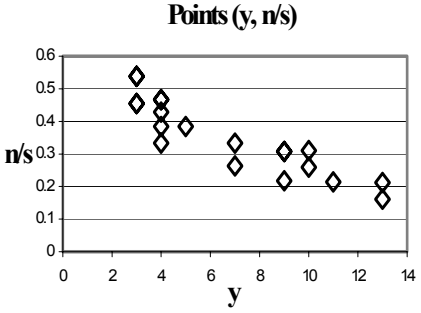

Figure 1. The dependence $n/s = F(y)$

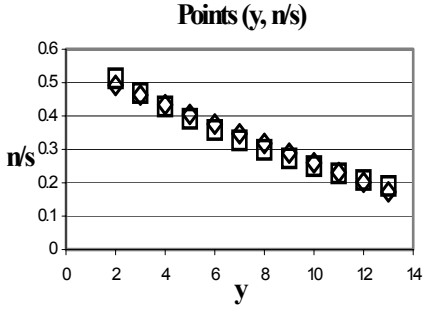

Figure 2. Approximations for $n/s = F(y)$

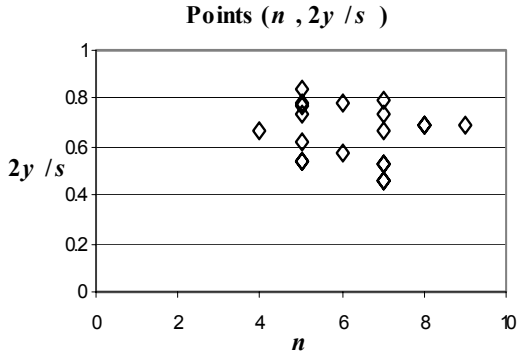

Figure 3. The dependence $2y/s = F(n)$

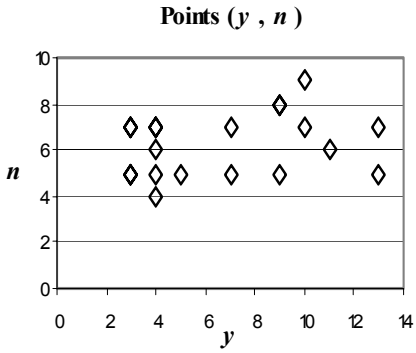

Figure 4. The dependence $n = F(y)$

[Baeza-Yates and Ribero-Neto, 1999] R. Baeza-Yates, B. Ribero-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[Blanco et al., 2006] X. Blanco, M. Alexandrov, A. Gelbukh. Modified Makagonov's Method for Testing Word Similarity and its Application to Constructing Word Frequency Lists. In: Advances in Natural Language Processing. *Research in Computing Science* 18, pp. 27–36, 2006.

[Cramer, 1946] H. Cramer. *Mathematical methods of statistics*. Cambridge, 1946.

[Gelbukh *et al*., 2004] A. Gelbukh, M. Alexandrov, Sang Yong Han. Detecting Inflection Patterns in Natural Language by Minimization of Morphological Model. *Lecture Notes in Computer Science* 3287, Springer, pp. 432–438, 2004.

[Gelbukh, 2003] A. Gelbukh. Exact and Approximate Prefis Search under Access Locality Requirements for Morphological Analysis and Spelling Correction. *Computación y Sistemas*, vol. 6, N 3, pp. 167–182, 2003.

[Gelbukh and Sidorov, 2003] A. Gelbukh, G. Sidorov. Approach to construction of automatic morphological analysis systems for inflective languages with little effort. *Lecture Notes in Computer Science* 2588, Springer, pp. 215–220, 2003.

[Goldsmith, 2001] J. Goldsmith. Unsupervised Learning of the Morphology of a Natural Language. *Computational Linguistics*, **27**, N 2. pp.153–197, 2001.

[Ivahnenko, 1980] A. Ivahnenko. *Manual on typical algorithms of modeling*. "Tehnika" Publ., Kiev, 1980 (in Russian).

[Makagonov and Alexandrov, 2002] P. Makagonov, M. Alexandrov. Empirical Formula for Testing Word Similarity and its Application for Constructing a Word Frequency List. *Lecture Notes in Computer Science* 2276, Springer, pp. 425–432, 2002.

[Porter, 1980] M. Porter. An algorithm for suffix stripping. *Program*, 14, pp. 130–137, 1980.