Experiments on Language Normalization for Spanish to English Machine Translation

Paula Estrella[1] and Pablo Duboue[2]

[1] Facultad de Matemática, Astronomía y Física
Universidad Nacional de Córdoba
Córdoba, Argentina
pestrella@hal.famaf.unc.edu.ar,
[2] Computer Science Department
Columbia University
New York, USA
pablo@cs.columbia.edu,
http://www.cs.columbia.edu/~pablo

Abstract. In this article we describe a trained system we have built, using freely available statistical packages and present experiments specifically designed to improve the performance of the system in the Spanish into English task. We performed both qualitative and quantitative evaluations that show better perceived translation quality as well as better BLEU and NIST scores.

Key words: Natural Language Processing, Statistical Machine Translation, Language Normalization, Stemming.

1 Introduction

Human beings have a natural need to comunicate and interact with each other. This need has always been achieved in different ways according to the available resources and the evolution of mankind. Today, resources are almost ilimited and that need became a fundamental requisite in some aspects of daily life. For example, it is very common, thanks to the Internet, to find the information we need but written in a language we do not understand. That is a problem we still face nowadays, the language barrier.

We are interested in the problem of translating Spanish into English. In this article we describe a trained system we have built, using freely available statistical packages and present experiments specifically designed to improve the performance of these systems in the Spanish→English task. The statistical packages we have used include the GIZA++ [1] package, the CMU-Cambridge Statistical Language Modeling Toolkit [2] and the ISI decoder [3, 4].

We improved the results by conflating the translation table generated by GIZA++ with a stemming algorithm adapted for Spanish. Afterwards, we normalize the training corpus according to the new translation table and re-train the system with this new corpus.

Our experiments show better translation quality as well as better scores when evaluating with automatic methods. Using the BLEU [5] evaluation metric (explained in Sec. 4.1) we obtained 0.2401 when translating without the improvement and 0.2523 for the system trained with a normalized corpus. The NIST [6] metric (explained in Sec. 4.2) scored 5.1873 for the system without the improvement and 5.4067 for the improved system. We also performed a qualitative evaluation with human judges which preferred our system better or comparable to the existing system more than 72% of the time.

In Section 1.1, we briefly explain some grammatical differences between Spanish and English and how they affect a translation system. We also mention related work on Spanish morphology and stemming. In section 2, we expose the basics of Statistical Machine Translation and we relate it to the tools used in this work. Section 3 contains the details of our proposed technique and Section 4 is dedicated to the experiments we performed to measure the impact of our proposed modification. In Section 5, we conclude by explaining the advantages and drawbacks of the actual improvement and we propose some further work.

1.1 The Problem

We work in Spanish to English translation for a number of reasons. First, this research has taken place at a Spanish speaking country, and therefore it poses regional impact. And second, Spanish is the third most spoken language in the world and has a level of similary to English close enough to be appealing for Statistical Machine Translation (SMT).

The Spanish language is very rich and has a complex grammar. It has a number of articles (demostrative, determinative, etc.) which, toghether with nouns and adjectives, suffer modifications depending on gender (female and male) and number (singular or plural). Conjugations in Spanish are more difficult than in English: three kinds of conjugations are distinguished, depending on the ending of the infinite verb (*-ar, -er, -ir*), there are eight verb times and six modes. Verbs also suffer modifications such as person and number, among others. Finally, as with any language, it has a large number of exceptions.

These situations are illustrated in the following examples:

Example 1 (Article, verb and noun agreement).

Singular: *La* cocina *es* pequeña.
The kitchen is small.
Plural: *Las* cocina*s son* pequeña*s*.
The kitchens are small.

Example 2 (Pronoun, verb agreement).

Singular: *Ella* com*ía* una manzana.
She ate an apple.
Plural: *Ellas* com*ían* una manzana.
They ate an apple.

These morphological changes make it harder to obtain an accurate translation, because the system should have a large amount of evidence in the form of Spanish words (such as articles or some conjugations of the same verb) that translate into the same English word.

Instead of giving the machine translation system a huge amount of training data to solve this problem, we can improve the statistical evidence by means of normalizing the lexical differences between the two languages. Smarter algorithms has been shown to outperform large quantities of training material [7].

As shown in the previous examples, most of the changes in word morphology occur in the form of suffixes, so the stem of a word remains unchanged in most cases (with the exception of irregular verbs whose stem may also suffer changes). Therefore, our approach to neutralize these changes is to use a stemmer for the Spanish language and map groups of words with the same stem into one token.

Example 3 (Word stemming).
Consider the following words:
acordada
acordado
acordadas
acordados
These words have a common stem that is *acord-*, so the system should have to learn that they might map into the word *agreed,* thus the four cases are reduced to one case.

But not every word with the same stem should belong to one group (because we could lose important evidence).

Example 4 (Stemming drawback).
The following words have the same stem (*acept*):
aceptó
aceptablemente
aceptable
aceptamos
aceptando
Suppose we group them all together and the system learns these words translate into *acceptable.* Then, we might translate a verb (for example *aceptamos*, English *accept*) into an adjective (*aceptable*, English *acceptable*), thus losing accuracy.

The above examples show that stemming helps neutralizing differences among languages but it has the drawback of being too restrictive to our purpose.

1.2 Related Work

An overview of the importance of morphology in MT is given in [8] along with the related problems introduced by morphology; it also explains different techniques used in MT and compares some techniques used to solve these problems.

Although Hui [8] exposes why stemming should work in SMT, he does not show empirical evidence or experiments that support his theory.

Spanish morphology is analysed and explained in Tzoukermann and Liberman [9, 10], while presenting a system that generates and analyses inflectional and morphological forms with simple data structures, such as automata.

There are rather few works including stemming in SMT; stemming is widely used in Information Retrieval (IR). Fuller and Zobel [11] compare several stemming algorithms applied to IR and experiments show that the Porter Stemmer is very accurate for finding possibe conflations of a word.

2 Statistical Machine Translation

The statistical tool we use is GIZA++[3] [12], that employs the noisy channel model. In noisy channel parlance, our source language is English and the target language is Spanish. This may seem a little confusing but in the noisy channel model we try to find out what was sent through the channel in order to understand what we received (in this case we want to find the sentence that produced a given translation). We can think of Statististical Machine Translation (SMT) in terms of programms that take as input sentence pairs (s, e) and outputs the conditional probability $P(e|s)$, interpreting it as "*e is the best translation for s*"; it thus looks for the English sentence e that maximizes $P(e|s)$, often written as

$$\underset{e}{argmax}\, P(e|s) \tag{1}$$

The probabilities are obtained with the Bayes' rule:

$$P(e|s) = \frac{P(e)P(s|e)}{P(s)} \tag{2}$$

A program that assigns a probability $P(l)$ to each sentence l in a given language, is called a *language model.* In our case, $P(e)$ is the source language model, $P(s)$ is the target language model and $P(s|e)$ is the translation model. Given that $P(s)$ does not depend on e, the problem is reduced to find the e that maximizes $P(e)P(e|s)$.

The source language model is the basic probability asigned to a sentence and is computed using the text corpus we input to the system.

Example 5 (Language model generation).
For the input corpus:

1. The film was boring but the actors were fantastic.
2. I like mexican food.
3. It all started about 300 years ago.
4. I like mexican food.

the first and third sentences have $P(e) = 1/4$, sentences 2 and 4 have $P(e) = 1/2$ and any other sentence has $P(e) = 0$.

[3] Available at `http://www.isi.edu/~och/GIZA++.html`.

As shown in Example 5 sentences never seen in the corpus will get zero probability even if they were correct. To overcome this problem the sentences are split into shorter strings called n-grams, that is, a n words string. Now, the probability can be computed:

$$P(e_1...e_j) = P(e_1) * P(e_2|e_1) * ... * P(e_j|e_1...e_{j-1}). \quad (3)$$

The right side of the equation is the conditional n-gram probability, also called a parameter. Brown et al. [13] describe how these parameters can be automatically estimated. In the case that any P is zero (because the corpus has not shown exactly that sequence of words), *smoothing* is done. Smoothing tries to find out whether or not subsequences are likely; by doing this, smaller n-grams are computed and a non-zero factor is added, so that no string will have zero probability.

To obtain a language model we use the Carnegie Mellon Statistical Language Modeling Toolkit[4] [2]. Starting from a monolingual corpus, it generates a file contaning the corresponding language model.

When a word s produces a word e, they are said to be aligned; sometimes a word produces zero or more words, the number of English words produced by a Spanish word is called the Spanish word *fertility*. Word alignments are likely to have nearby positions in the source and target sentences (for example, words at the beginning align to words at the beginning, and so on), however some words may appear far from the word that produced it. That is, for instance, the case of adjectives and nouns: in Spanish adjectives are usually preceded by nouns but in English their order is inverted. This effect is called *distortion*.

The estimation of the translation model is done using different algorithms or models (IBM-1 to IBM-5), fully described in [13]. These models differ, for example, in the way they treat alignments: Model 1 assigns equal probabilities to all possible alignments, that means a Spanish word can produce an English word at any position. Model 2 is modified to let the order of both the Spanish and English word affect the probability $P(s|e)$; here two words are connected depending on the position they have and the length of the sentences they belong to. In Model 3 words only connect to at most another word and this connection has the same contrains as in Model 2. Model 4 improves the distortion probabilties, thus allowing better Subject-Verb relations. Model 5 is a refinement of Model 4, being this model the most efficient, in the sense that it does not lose probability on strings that are of no interest; in contrast Models 3 and 4 are not efficient.

GIZA++ is based on GIZA[12], the SMT toolkit developed during the summer workshop in 1999 at Johns-Hopkins University. GIZA++ extends the GIZA toolkit by implementing IBM Models 4-5 and improving some features like smoothing for fertility, among others. Och [14, 15] describes and compare the alignment models implemented in GIZA++.

The main tables GIZA++ generates are:

[4] Available at `http://mi.eng.cam.ac.uk/~prc14/toolkit.html`.

Translation: contains the distribution for $P(s|e)$.

Fertility: contains a list of numbers per source token, indicating the probability of having fertiliy zero, one, etc. up to the maximum defined in the program.

Alignment: contains the probability of moving a source word to given position; it also has the length of the source and target sentences.

Distortion: contains the distortion probabilities for each token.

P0: the probability of not inserting `NULL` tokens.

So far we have presented, a minimal overview of the process of obtaining the probabilities $P(e|s)$ from a set of sentence pairs (e, s); this process is called *training*. The final stage to determine the translation of a string, is the search of the sentence that maximices Equation 1, called *decoding*. There are many implementations of decoding algorithms but some of these tools are not freely available, as is the case of the original decoder implemented by IBM. We decided to use the ISI ReWrite Decoder[5] [3] not only because of its availability, but also because it can be run as server making the decoding process lighter.

3 Our Technique

In addition to an aligned bilingual corpus, we need a vocabulary file to perform the training. GIZA++ requires a vocabulary file with the following format:

Example 6 (Vocabulary file sample).

Unique Id	String	No. occurrences
597	primero	248
598	propio	248
599	estructurales	248
600	valor	247
601	régimen	247
602	relaciones	246
603	temor	245

The translation table generted by GIZA++ has the following format:

`s_id` `t_id` $P($`t_id`$|$`s_id`$)$

where `s_id` is the unique id assigned to source words, `t_id` is the unique id of the target word and P is the probability of translating `s_id` as `t_id`. These ids correspond to the vocabulary file used to train the model.

Example 7 (Translation table sample).

[5] Available at `http://www.isi.edu/licensed-sw/rewrite-decoder/`.

Unique source Id		Unique target Id		Probability
1284	(amplia)	1166	(broad)	0.167711
4759	(ampliada)	4859	(enlarged)	0.361962
9596	(ampliamente)	2984	(reclassified)	0.124721
14962	(amplificador)	22749	(aggravating)	0.142527
9349	(amplitud)	3556	(breadth)	0.49884
1388	(amsterdam)	1955	(amsterdam)	0.580654
12942	(anatoly)	22920	(photographer)	0.498836
7544	(ancha)	12156	(broadband)	0.16628
2758	(anciana)	17047	(senior)	0.0355824
6456	(andaluz)	20303	(andalusian)	0.124147

This table do not store the strings learned but a unique identifier; the words in parenthesis are extracted from the vocabulary file.

The first step (Step 1), is to use a stemmer over the translation table. The candidate algorithm is an adaptation for Spanish of the Porter stemmer[6] [16]. One stem produces a family of words that may include different grammatical entities. If we map a whole family into the same token, we might lose important evidence. Therefore, we decided to cluster family words during the stemming process (Step 2).

The stemming algorithm has three main procedures to remove suffixes:

Delete attached pronoun: pronouns can be removed when attached to verbs because they are implicitly mentioned. Some pronouns are *me, te, se, le, la, lo.*

Delete standard suffix: there are many suffixes that identify different grammatical entities. For example, *mente* stands, in most cases, for adverbs and *logía* for nouns. In this step we decided to form three categories: nouns (*logía, logías, encia, encias*), adverbs (*mente, amente*) and other suffixes.

Verb suffix removal: because of the complexity of verbs conjugation in Spanish there are many suffixes that can be removed; we distinguish three categories depending on the suffix: past (for example *aba, ía, ido*), continuous (*iendo, ando*) and other verbs (examples are *iera, iese, ar, er, ir*).

After that, we have a normalizer specific to a corpus and a translation model. The normalization table should have the following format:

acortar	TOKEN11
acorten	TOKEN11
acortándolo	TOKEN12
acortada	TOKEN13
acortado	TOKEN13
acertadamente	TOKEN15
acertar	TOKEN16
acertemos	TOKEN17
acertada	TOKEN17

[6] Available at `http://www.tartarus.org/~martin/PorterStemmer/`.

Finally, the source language corpus should be normalized according to the results previously obtained (Step 3); these steps must preserve corpus alignment in order to re-train GIZA++ with the new corpus.

The process described in this Section makes Spanish a little more like English, by means of treating many words as English does. Another reason to expect it to work, is that we are reducing the number of parameters, a situation similar to having less noise in our channel. Figure 1 shows an example normalized sentence (the original sentence was *"creo que precisamente suecia , un ejemplo de transparencia en el marco de su actual presidencia , sería la llamada para dar aquí un paso hacia delante ; debe quedar claro que este debate es muy importante para el ciudadano europeo"*).

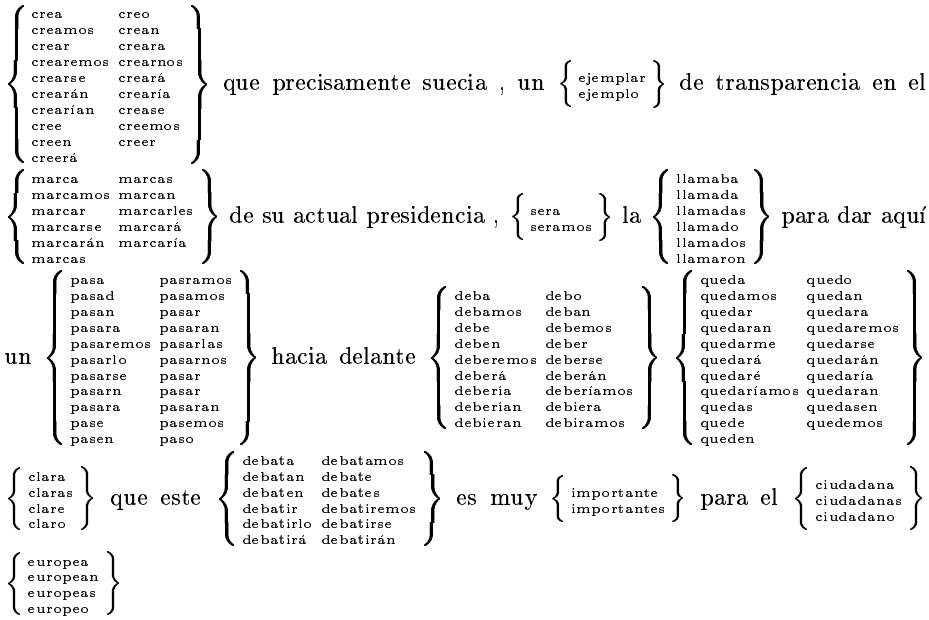

Fig. 1. Example normalized sentence.

4 Experiments

This Section details the experiments we performed to find out wether the proposed technique improves translation or not. The system was trained twice: the first time with the original bilingual corpus and the second time, the Spanish part was normalized according to the previous Section. These two systems translated 50 test sentences and we evaluated the output with BLEU and NIST methods.

We used a subset of the Europarl corpus as training set consisting of 64,281 sentences. Europarl is a multilingual corpus containing the Proceedings of the

European Parliament in the 11 official languages. Koehn built Europarl as described in [17] and it has been used afterwards in numerous works. For the evaluation process we built a testing set with 34 sentences from Europarl (not contained in the training set) and 16 sentences from the Linux Howto.

We applied the improvement as follows:

Align a parallel corpus: The data from Europarl was aligned at the sentence level using *align* [18], a program for aligning sentences at the paragraph and sentence level. *Align* is based on the fact that parallel translations of text do not differ much in length.

Normalize the corpus: the stemming algorithm was implemented maintaining the modifications detailed in Section 3. We trained GIZA++ to obtain the translation table corresponding to the aligned corpus. After that, we applied the normalization algorithm to the table. To prepare the corpus to re-train we used a script that replaces words (from the translation table) with the corresponding tokens; the corpus structure remains unaltered.

Build a translation model (re-train GIZA++): as explained in Section 2, a translation model is generated by training GIZA++ with the modified corpus.

Build a language model: we used the tables generated in the previous step to build a language model using CMU-Cambridge toolkit.

Translate test sentences: in order to translate text, the test corpus must be consistent with the data used to train the system. Therefore, we first normalized the test set and then used the ISI ReWrite decoder.

4.1 BLEU metric

Evaluating a MT system is very important to measure the impact of the system and also to find out wether the ideas involved are effective or not. There are many ways to achieve evaluations but it is preferably to have human judges evaluate the translations. Unfortunately, this is a time consuming task and the work done is not reusable. Papineni [5] propose a method (*Bilingual Evaluation Understudy*) to automatically evaluate translations, making this process quicker and with no human labor costs.

BLEU is based on *Word Error Rate* (WER) metric used in speech recognition but apropriately modified to deal with multiple references. BLEU uses WER to measure *closeness* to human translations and human references as ideal translations. The main idea behind BLEU is to assign higher scores to translations more likely to the human translations (scores range from 0 to 1); this algorithm uses a weighted average of $n-$grams matches against the given references (the more $n-$gram match the reference, the higher the score would be) and it penalises translations whose lengths differ significantly from the references given.

This method correlates with human evaluations and is language-independient, which makes it reusable and accurate. Papineni et al. gives enough details of the method, so it can be easily implemented.

4.2 NIST metric

Based on BLEU, NIST desinged its own evaluation method and implemented it as a freely available toolkit[7]. The major difference between NIST and BLEU is the formula used to compute the $n-$gram matches: BLEU uses a geometric average of $n-$gram counts and NIST uses an arithmetic average. This formula also modifies the brevity penalty to minimize the influence of small variations in lenghts over the final score.

A comparison between NIST and BLEU scores, showed that NIST score improves significantly in score stability and reliability. It also showed that NIST score correlates better than BLEU for human judges.

4.3 Results

At this point, we had two translations of the same test corpus (one with normalized corpus (1) and the other without the normalization (2)). We evaluated these two translations using two references. Table 1 summarizes the overall scores and Table 2 shows $n-$gram precision.

Table 1. BLEU and NIST scores for translation 1 and 2.

	System 1	System 2
BLEU	0.2523	0.2401
NIST	5.4067	5.1873

Table 2. $N-$gram precision score.

	1−gram	2−gram	3−gram
NIST System 1	4.7466	0.5942	0.0589
NIST System 2	4.5663	0.5580	0.0526
BLEU System 1	0.6247	0.2310	0.1123
BLEU System 2	0.6002	0.2196	0.1074

We also made a qualitative evaluation of twenty sentences out the fifty used for the quantitative evaluation; to avoid benefiting any of the MT systems, the sentences were randomly chosen. We count with three native speakers, who contributed to our work. The judges received the following instructions:

> *You have here a Spanish sentence (E), a reference translation (M) and two automatically translated English sentences, that come (in random*

[7] Available at `http://www.nist.gov/speech/tests/mt/mt2001/resource/`

order) from our two systems (TA,TB). We ask you to choose which one is better or say if both are equally good.

Table 3 shows the results obtained for System 1 and System 2. To summarize results we counted the votes each sentence received, assesing a better performance of the system to which we applied the normalization technique. The differences are significant according to a three-column χ^2-square test. The human judges mentioned most of the times the differences were at the verb, with the wrong (baseline) translation having a completely outrageous verb form. We believe the reason behind this behavior of the baseline system lies in the fact that very few training cases are available for each inflected verb form, strongly misleading the statistical system into picking surrounding words as the real translation.

Table 3. Qualitative evaluation results.

Sentence	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	total
System 1 is better	3	0	2	2	0	1	0	3	0	1	2	1	2	0	0	2	3	3	2	2	29
System 2 is better	0	2	0	0	2	1	1	0	3	1	0	0	1	2	2	0	0	0	0	0	17
Both equally good	0	1	1	1	1	1	2	0	0	1	1	2	0	1	1	1	0	0	1	1	14

5 Conclusion

This work focuses on the unresolved problem of obtaining automatic translations with human quality. We made a small scale experiment that turned out to work fine but has some drawbacks. First, the normalization we applied should be refined or combined with other techniques, because it still can lose evidence by mapping too many words into a same token. Second, we did not count with the necessary resources to try the technique with a larger corpus. Moreover, aside from its computational complexity, SMT also demands a large amount of human effort to try an idea or implement a system. Nevertheless we find it a very interesting area with plenty of work to do in MT.

5.1 Further Work

We propose the extension of this technique to other language pairs where the target is more complex than the source. To name a few examples: English→Chinese, Korean→English, and German→English. New language pairs can be a challenging task because of the differences, for example, among English and German; in our opinion German has a more more complex grammar than Spanish. The normalization technique should be improved to overcome the drawbacks explained in Section 5, it could be combined with other techniques or it could be implemented including some linguistic information (for example Part-Of-Speech tagging).

References

1. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics **29** (2003) 19–51
2. Clarkson, P., Rosenfeld, R.: Statistical language modeling using the CMU-cambridge toolkit. In: Proceedings of Eurospeech '97, Rhodes, Greece (1997) 2707–2710
3. Germann, U., Jahr, M., Knight, K., Marcu, D., Yamada, K.: Fast decoding and optimal decoding for machine translation. In: Annual Meeting of the Association for Computational Linguistics (ACL-2001). (2001) 228–235
4. Germann, U.: Greedy decoding for statistical machine translation in almost linear time. In: Annual Meeting of the Association for Computational Linguistics (ACL-2003). (2003) 72–79
5. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. Technical report, IBM (2001)
6. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-ocurrences statistics. In: Human Language Technology: Notebook Proceedings, San Diego, CA (2002)
7. Lapata, M., Keller, F.: The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of nlp tasks. In: Proceedings of HLT-NAACL, Boston, MA (2004)
8. Hui, B.: The role of morphology in machine translation. Technical Report CS-98-22, University of Waterloo, Computer Science Department (1998)
9. Tzoukermann, E., Liberman, M.Y.: A finite-state morphological processor for spanish. Technical report, AT&T Labs (1990)
10. Tzoukermann, E., Liberman, M.Y.: A finite-state morphological processor for spanish. In: 13th International Conference for Computational Linguistics (COLING-90), Helsinki, Finland (1990)
11. Fuller, M., Zobel, J.: Conflation-based comparison of stemming algorithms. In: Proceedings of the Third Australian Document Computing Symposium, Sydney, Australia. (1998)
12. Al-Onaizan, Y., Curin, J., Jahr, M., Knight, K., Lafferty, J., Melamed, I.D., Och, F.J., Purdy, D., Smith, N.A., Yarowsky, D.: Statistical machine translation, final report, JHU workshop. Technical report, Johns-Hopkins University, Baltimore, Maryland, MD (1999)
13. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics **19** (1993) 263–311
14. Och, F.J., Ney, H.: Improved statistical alignment models. In: Annual Meeting of the Association for Computational Linguistics (ACL-2000), Hong-Kong, China (2000) 440–447
15. Och, F.J., Ney, H.: A comparison of alignment models for statistical machine translation. In: International Conference for Computational Linguistics (COLING-2000), Saarbrücken, Germany (2000) 1086–1090
16. Porter, M.F.: An algorithm for suffix stripping. Program **14** (1980) 130–137
17. Koehn, P.: Europarl: A multilingual corpus for evaluation of machine translation. Available on-line at `http://www.isi.edu/ koehn/europarl/` (2002)
18. Gale, W.A., Church, K.W.: A program for aligning sentences in bilingual corpora. In: Annual Meeting of the Association for Computational Linguistics (ACL-91). (1991) 177–184