UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Departament de Sistemes Informàtics i Computació

IARFID master's thesis:

Selecting translations to be post-edited by Sentence-Level Automatic Quality Evaluation

---

Presented by:
Mercedes García Martínez

Supervised by:
Prof. Francisco Casacuberta Nolla
Antonio L. Lagarda Arroyo

June 2012

Acknowledgements

I am grateful to my family and friends, in special my classmates Ximo, Sergi and Marcos and all my labmates, particularly Jesús for the support and help received. Moreover, I would like to thank my research group PRHLT and my supervisors for their advices and guidelines.

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 287576 (CasMaCat)). Finally, I would like to thank the ITI to lend me a place to work with them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Translation industry is and will remain a major sector with a high rate of growth because it plays an essential role in the economical, cultural and social development of the country and even more with globalization. The work of professional translators has become a basic need in all countries over the world.

In the last years, translation activities have grown rapidly. This increase has been motivated by various causes, amongst which are technological and scientific advances, the increase of international relations, growth of the tourism sector, increase of the new information diffusion media, more communication between different linguistic communities, etc.

For example, in the European Parliament is crucial to translate lots of documents immediately and the human translators cannot meet the demand. The European Union is a community of 27 countries, in which 23 official languages are spoken and three alphabets are used: Latin, Greek and Cyrillic. The European Parliament is committed to debate and discussion in all European Union languages. Some statistics about the multilingualism and its cost in the European Parliament are (Duch Guillot, 2007):

- For 2006 the cost of translation is estimated at €800 million in 2005 and the cost of interpretation was almost €190 million.
- Multilingualism expenditure represents over one third of the total expenditure of Parliament.
- The European Parliament translated 673,000 pages during the first half of 2007.
- Since 2005 the European Parliament has translated over a million pages a year.
- The European Union system, on average, requires over 2000 translators and 80 interpreters per day.

For these reasons, a growing interest on an automatic method to translate has been shown by politicians and research groups. Machine translation (MT) investigates this problem. Therefore, the translation industry increasingly uses more MT because it helps translators to increase their job production.

For instance, the newspaper *El periódico de Catalunya* is automatically translated from Spanish to Catalan every day.

Despite progress in research on MT, the translations are not automatically getting a high quality and must be reviewed by a human translator to achieve the expected quality. This review effort can be tedious and even cost more time than translate them from scratch. However, the translation quality is not the same in all the sentences. Therefore, it is important to have a system to know the quality of the translation. If the translator does not have a lot of time to post-edit the translation of a text, the quality of the translations guide him through the worse or better translated sentences. Moreover, a confidence measure in each sentence is similar to other known techniques like *fuzzy match threshold* used it by translators. *Fuzzy match threshold* gives the degree of match between a source document segment and a translation memory segment.

The next two sections, explain in detail and describe the state of the art of machine translation and confidence measures, respectively. Finally, we explain the problem statement in the last section of this chapter.

1.1 Machine Translation

Machine translation (MT) is a sub-field of natural language processing (NLP). NLP extracts meaningful information from natural language and produces natural language through the interaction between humans and machines. MT investigates how to translate automatically a given source text or speech into a target text or speech, preserving the same meaning.

An example of use translating from Spanish to English is:

- Source: "Hola ¿qué tal?"
- Target: "Hello, what's up?"

On a basic level, MT performs simple substitution of words in one language for words in another language, but that alone usually cannot produce a good translation of a text, because recognition of whole phrases and their closest equivalent is needed. Solving this problem with corpus and statistical techniques is a rapidly growing field that is leading to better translations, dealing differences in linguistics, idioms and anomalies. MT has different approaches: rule-based, statistical, example based and hybrid MT:

- **Rule-based** denotes machine translation systems based on linguistic information about source and target languages retrieved from bilingual dictionaries and grammars. It is a very costly method because making the rules requires linguistic experts.

- **Statistical machine translation** (SMT) is a MT paradigm where translations are generated on the basis of statistical models whose parameters are derived from the analysis of bilingual text corpora. SMT systems have proven in the last years to be an important alternative to rule-based MT systems, even outperforming commercial MT systems in the tasks that have been trained on. Moreover, the development effort behind a rule-based MT system and a SMT system is dramatically different, the latter is able to adapt to new language pairs with little or no human effort, whenever suitable corpora are available (Hutchins, 2003).

SMT is as the problem of translating a given source sentence:

$x = x_1 \dots x_j \dots x_J$ into a target sentence y, which $\hat{y} = y_1 \dots y_i \dots y_I$ and the sentence chosen is the one that maximizes the posterior probability (Brown et al., 1993). This statement is formalized in the following equation:

$$\hat{y} = \arg\max_{y} Pr(y|x) \tag{1.1.1}$$

- **Example based** approach (Nagao, 1984) is characterised to be used with bilingual corpora and its main knowledge base at run-time. It is a translation by analogy and can be viewed as an implementation of case-based reasoning approach of machine learning.
- **Hybrid machine translation** (HMT) merges statistical and rule-based translation approaches. There are too main approaches:
  - Rules post-processed by statistics: Translations are performed using rules based engine. Statistics are then used in an attempt to adjust/correct the output from the rules engine.
  - Statistics guided by rules: Rules are used to pre-process data in an attempt to better guide the statistical engine. Rules are also used to post-process the statistical output in order to perform functions such as normalization.

1.2 Confidence measures

A confidence measure (CM) is a number indicating the degree of belief that a unit output by a recogniser (word, phrase, sentence) is indeed (Cox, 2004). Confidence measures are widely used in speech recognition (Wessel et al., 2001), but until recently they have not been applied in the area of MT. Gandrabur and Foster (2003) introduced confidence measures for a translation prediction task in an interactive environment.

The number of MT applications has grown in recent years, the demand for the ability to detect erroneous sentences or words also has increased (Ueffing et al., 2003; Ueffing and Ney, 2007). Confidence estimation metrics use features extracted from machine translations, and usually also from the source text and monolingual and bilingual corpora. It also uses information about parameters of the MT system used to translate. The extracted features are given to a machine learning algorithm that learns a model that predicts the quality of the translation. Some authors have investigated about translation quality from data annotated with scores derived either from automatic MT evaluation metrics (Blatz et al., 2004), NIST (Doddington, 2002), WER (Tillmann et al., 1997) or using human annotation at the sentence-level (Quirk, 2004; Specia et al., 2009a). This master thesis is focused on CM at the sentence-level.

The related work begins with Blatz et al. (2004). Regressors and classifiers are trained on features extracted for translations. They use NIST for classification and map the estimated scores into two classes for regression. Quirk (2004) uses classifiers with labels for bad and good translations.

Gamon et al. (2005) train an SVM classifier using a number of linguistic features extracted from machine and human translations to distinguish between human and machine translations. Specia et al. (2009a) use a number of features to train a Partial Least Squares regression model to estimate both NIST and human scores. In Specia et al. (2009b) is predicted a continuous score based on human annotation. The selection of sentences in multiple MT systems estimating confidence scores is presented in Specia et al. (2010b). He et al. (2010) train a binary classifier to predict whether the SMT output is more suitable than the translation memory output using a standard translation edit rate (TER) to measure the distance between a reference translation.

1.3 Problem statement

The aim of this master thesis is to describe a system that predicts a confidence measure by sentence-level of a given text. Using this method, the user or translator that has to post-edit the MT translation has an extra information about its confidence measure that can help him in its post-edition. Moreover, taking into account this confidence measure, the user can indicate a threshold to select, on the one hand, the worst translated sentences to reject or post-edit them or, on the other hand, the best translated sentences to post-edit just them or leave them without post-edition. In this way, the user can avoid post-edition effort saving time post-editing good translated sentences (useful, for instance, for webpages that do not need a high translation quality) or removing bad translated sentences because is better to translate it from scratch.

The method proposed improves other simpler systems used, like the ones that use just sentence length. For this purpose, a model is trained with features extracted from source and target sentences ("black-box", MT system-independent) and those features that provide more information are selected. We present a study of classifiers, selection methods and their corresponding parameters. Initially, it was thought as application of the translation systems Celer[1] and Pangeanic[2] , but for financial reasons and added problems, it is a research study and can be applied in the future to a real translation platform.

In chapter 2 is explained the method proposed with all the phases (feature extraction, classification and feature selection), in chapter 3 is showed the experiments carried out and results. Finally, in chapter 4 states the conclusions and future work.

[1] http://www.celersol.com
[2] http://www.pangeanic.com

Chapter 2

Computing Confidence Measures in MT

2.1 Introduction

Nowadays, it is important to know the degree of correction of a translation extracted from a MT system, since these translations usually do not reach the desired quality, this degree can help the post-edition of the MT translation. For this purpose, the method proposed provides a quality estimation and detects the best and the worst translated sentence for a given translation.

Our approaches are based on machine learning, a branch of artificial intelligence that given samples (data), can capture features of interest to recognize complex patterns and make intelligent decisions based on data.

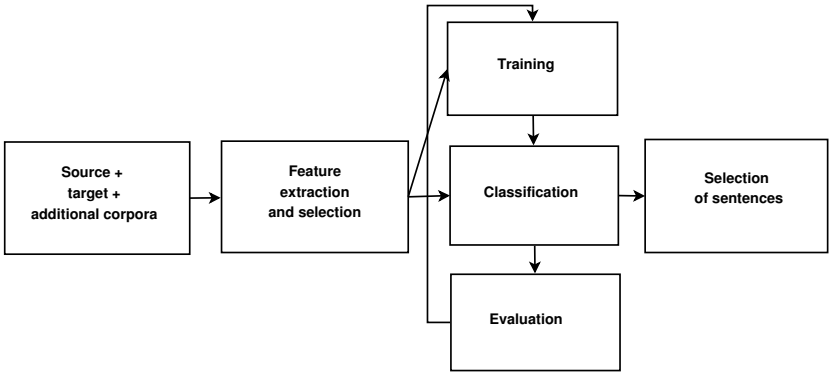
There are several types of machine learning algorithms but we focus on the type called supervised learning because of the data and that we want to predict the correction value of the translations. Supervised learning generates a function that maps inputs to desired outputs. For example, in a classification problem, the learner approximates a function mapping a vector into classes and in a regression problem the learner gives a continuous value.

Given a set of training samples of the form: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ where $\mathbf{x} = \{x_1, x_2, \ldots, x_D\}$, N is the number of samples and D is the number of dimensions or features. A learning algorithm seeks a function $g : X \to Y$, where X is the input space and Y is the output space.

The method proposed (illustrated in figure 2.2) estimates the Translation Error Rate (TER) value of given translated sentences. TER is an automatic measurement of the effort of a user to correct MT output to make it good translation. This prediction is useful to know an approximation of the quality of a translation giving a confidence measure of them. In addition, this measurement has the advantage to be automatic, whereas other measurement as human scores requires a human expert to tag.

In order to predict the TER, we extract features from the source and target text and additional corpora, the section 2.2 explains this step in more detail. Apart from the features, we calculate the TER of the training samples with the given reference and translated sentences.

Figure 2.1: Diagram of the method



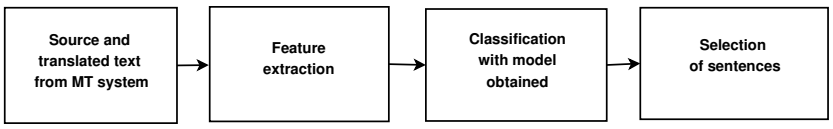
Then, we carry out the feature selection choosing the features that correlate better with the TER. Moreover, we train a statistical model using the selected features and calculate TER with optimized parameters.

Furthermore, the model is evaluated. We have proven several classifiers (explained in 2.3) and features selection methods (explained in 2.4). In order to optimize the parameters, we select the best classifier and the feature selection method that fits better, we evaluate the samples with cross-validation (we split the training samples in 5 partitions and each partition is tested while the rest are trained).

To conclude, we classify with regression the test samples predicting their TER. Optionally, the user can select the best or worst translated sentences.

Once the model is trained, the process is simplified, just extracting the selected features and classifying with the model, optionally the user can select the sentences as good or bad translations:

Figure 2.2: Diagram of the process



In the next sections, each phase is explained in more detail.

2.2 Feature extraction

The feature extraction comprises 156 features (for more information see apendix I).

We use the following data sources to extract the features:

- Source and target sentences.
- Monolingual and parallel corpora Europarl version 2 (Koehn, 2006), version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
- Inverse translation of the target sentences using a model trained with Europarl version 2.
- 1000-best extracted translating the source sentences with a model trained with the corpus Europarl version 2 and the software Moses (Koehn et al., 2007).

The system use "black-box" features because they are extracted given only the input sentence, its translation and monolingual or parallel corpora.

The features extracted can be divided in two groups: those extracted given only the input sentence and its translation and those extracted using in addition a monolingual or parallel corpora.

- Features given only the input sentence and its translation are:
  - Number of tokens and punctuation marks in the source and target sentence and if the target sentence has mismatched quotation marks.
  - Token length average in the source sentence, average number of occurrences of the target word within the target sentence, sentence lengths and percentages of punctuation symbols, stop-words (Google, 2001) and numbers of the source and target sentences and their ratio.
  - Number of mismatching brackets, each punctuation symbol (and all of them), numbers in absolute terms and normalized by sentence length between source and target sentences and number of mismatched brackets of the target sentence.
  - TER value between POS-tagged source and target sentences. POS-tagging is extracted with Freeling (Padró, 2011)) by TALP Research Center in Universitat Politècnica de Catalunya.
  - Number of verbs, nouns and adjectives in the source and target sentences and their ratio using Freeling to detect POS-tags.
- Features that also use monolingual or parallel corpora are:
  - Unigrams, bigrams and trigrams language model probability and perplexity of source and target sentence obtained using the source and target side of the corpora Europarl version 5 (Koehn, 2010), as well as News Commentary (Callison-Burch et al., 2010).

- Average number of translations per source word in the sentence, as given by probabilistic dictionaries produced by GIZA++ (Och and Ney, 2000) extracted from the parallel corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010), thresholded using percentage 0.01 unweighted and percentage 0.2 weighted by the inverse frequency of the words in the source sentence.
- Percentage of 1-grams and 1-2grams in quartiles 1 and 4 of frequency, percentage of distinct 1-3grams and average frequency of 1-grams, 2-grams and 3-grams belonging to each quartile of source words of the corpora Europarl version 5 and News Commentary of the source language.
- Trigrams language model probability of target sentence trained on POS-tagged corpora of the target language Europarl version 5and News commentary extracted using Freeling (Padró, 2011).
- TER computed with source sentence as reference and the machine translation of target sentence to the source language as hypothesis. The translation is done using a model trained with the corpus Europarl version 2 (Koehn, 2006) and the software Moses (Koehn et al., 2007).
- Word-level features combined into sentence-level:
  * Geometric average words of the IBM1 probability (Ueffing et al., 2003) trained with the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) in the target sentence.
  * Average of words in absolute value, ranking and considering probabilities of frequency according to Levenstein, target position, any position and average position criteria of the 1000-best (Sanchis, 2004) translated with a model trained with the corpus Europarl (Koehn, 2006) and the software Moses (Koehn et al., 2007).
  * Average of words probability that the word is correct given by Naïve Bayes (Sanchis, 2004) classifier of the source sentence belonging to each quartile using the corpus Europarl version 2 of the source language. The optimization of the parameters was done with a separate development set Europarl version 2.

The first 17 features are given with the software from NAACL workshop 2012 (Callison-Burch et al., 2012). The features extracted using 1000-best and Naïve Bayes model are calculated with the software of Sanchis (2004).

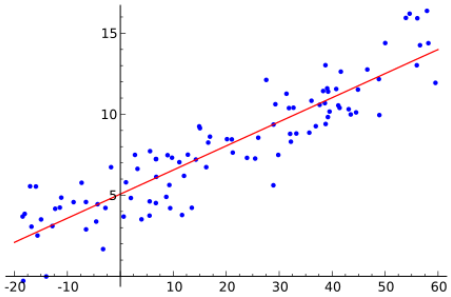
2.3 Classification methods

In this section, we explain the different classification methods tested (linear regression, simple linear regression, radial basis function, three decision stump, multilayer perceptron, support vector machines and partial least squares regression) to train the model and predict the TER. A set of extracted features X and a predicted value y (in our case TER) are used.

2.3.1 Linear regression

Linear regression is an approach to modelling the relationship between a scalar dependent variable y and the explanatory variables denoted in X. The case of one explanatory variable is called simple regression. More than one explanatory variable is multiple regression (we have one explanatory variable).

Figure 2.3: Linear regression (from Wikipedia)



In linear regression, data are modelled using linear prediction functions and the parameters of the model are estimated from the data. Most commonly, linear regression refers to a model in which the conditional mean of y given the value of X is an affine function of X. Like all forms of regression analysis, linear regression focuses on the conditional probability distribution of y given X, rather than on the joint probability distribution of y and X, which is the domain of multivariate analysis.

Linear regression is the first type of regression analysis to be studied rigorously, and to be used extensively in practical applications. This is because models which depend linearly on their unknown parameters are easier to fit than models which are non-linearly related to their parameters because the statistical properties of the resulting estimators are easier to determine.

Linear regression has many practical uses. Most applications of linear regression fall into one of the following two broad categories:

- If the goal is prediction, or forecasting, linear regression can be used to fit a predictive model to an observed data set of y and X values. After developing such a model, if an additional value of X is then given without its accompanying value of y, the fitted model can be used to make a prediction of the value of y.
- Given a variable y and a number of variables $\mathbf{x}_1, \ldots, \mathbf{x}_D$ that may be related to y, linear regression analysis can be applied to quantify the strength of the relationship between y and the $\mathbf{x}_j$, to assess which $\mathbf{x}_j$ may have no relationship with y at all, and to identify which subsets of the $\mathbf{x}_j$ contain redundant information about y.

Linear regression models are often fitted using the least squares approach. Conversely, the least squares approach can be used to fit models that are not linear models. Thus, while the terms *least squares* and *linear model* are closely linked, they are not synonymous (Cohen et al., 2002).

Given a data set $\{y_i, x_{i1}, \ldots, x_{iD}\}_{i=1}^{N}$ of n statistical units, a linear regression model assumes that the relationship between the dependent variable y_i and the $D - vector$ of regressors x_i is linear. This relationship is modelled by an error variable ε_i, an unobserved random variable that adds noise to the linear relationship between the dependent variable and the regressors. Thus, the model takes the form:

$$y_i = \beta_1 x_{i1} + \ldots + \beta_D x_{iD} + \varepsilon_i = x_i^T \beta + \varepsilon_i, \tag{2.3.1}$$

where $i = 1, \ldots, N$, The equation usually is represented as:

$$y = X\beta + \varepsilon, \tag{2.3.2}$$

where:

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}, X = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{pmatrix} = \begin{pmatrix} x_1 & \ldots & x_{1D} \\ x_{21} & \ldots & x_{2D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \ldots & x_{ND} \end{pmatrix},$$

$$\beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_D \end{pmatrix}, \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}.$$

- y_i is called the response variable or dependent variable.
- X_i are called regressors, predictor variables, or independent variables.
- β is a $D - dimensional$ parameter vector. Its elements are also called effects, or regression coefficients.
- ε_i is called the error term. This variable captures all other factors which influence the dependent variable y_i other than the regressors x_i.

Simple linear regression

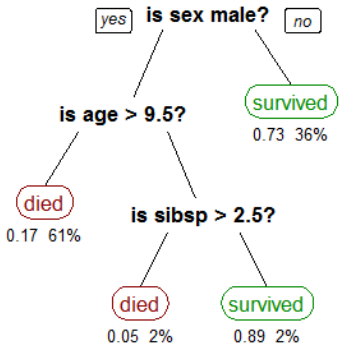
Simple linear regression is the least squares estimator of a linear regression model with a single explanatory variable. In other words, simple linear regression fits a straight line through the set of N points in such a way that makes the sum of squared residuals of the model (that is, vertical distances between the points of the data set and the fitted line) as small as possible.

The adjective 'simple' refers to the fact that this regression is one of the simplest in statistics. The fitted line has the slope equal to the correlation between y and x corrected by the ratio of standard deviations of these variables. The intercept of the fitted line is such that it passes through the center of mass (x, y) of data points (Kenney and Keeping, 1962).

2.3.2 Decision stump tree

A decision stump tree is a decision tree that uses decision stump models. Decision tree learning, used in statistics, data mining and machine learning, uses a decision tree as a predictive model that maps observations about an item to conclusions about the item's target value. More descriptive names for such tree models are classification trees or regression trees. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. An example of decision tree is:

Figure 2.4: A tree showing survival of passengers on the Titanic (from Wikipedia)



A decision stump (the type of decision used) is a machine learning model consisting of a one-level decision tree (Ai and Langley, 1992). That is, it is a decision tree with one internal node (the root) which is immediately connected to the terminal nodes (its leaves). A decision stump makes a prediction based on the value of just a single input feature, also called 1-rules (Holte, 1993).

Depending on the type of the input feature, several variations are possible. For nominal features, one may build a stump which contains a leaf for each possible feature value or a stump with two leaves, one of which corresponds to some chosen category, and the other to the remaining categories. For binary features these two schemes are identical. A missing value may be treated as a yet another category.

For continuous features, usually, some threshold feature value is selected, and the stump contains two leaves for values below and above the threshold. However, rarely, multiple thresholds may be chosen and the stump therefore contains three or more leaves.

Decision stumps are often called *weak learners* in machine learning ensemble techniques such as bagging and boosting (Reyzin and Schapire, 2006).

The term *decision stump* was coined in a 1992 ICML paper by Wayne Iba and Pat Langley (Ai and Langley, 1992; Oliver and Hand, 1994).

2.3.3 Radial Basis Function network (RBF network)

Other technique to train a model is the Radial Basis Function network (RBF network). A RBF network is an artificial neural network that uses radial basis functions as activation functions.

A radial basis function (RBF) is a real-valued function whose value depends only on the distance from the origin, so that $\phi(x) = \phi(||x||)$; or alternatively on the distance from some other point c, called a center, so that $\phi(x) = \phi(||x-c||)$. Any function that satisfies the property $\phi(x) = \phi(||x||)$ is a radial function. The norm is usually Euclidean distance, although other distance functions are also possible. Sums of radial basis functions are typically used to approximate given functions. This approximation process can also be interpreted as a simple kind of neural network. RBF has different types, but we use widespread approximation called Gaussian.

A Gaussian function is a function of the form:

$$\phi(x) = ae^{-\frac{(x-b)^2}{2c^2}} \tag{2.3.3}$$

For some real constants $a, b, c > 0$, and $e \approx 2.718281828$ (Euler's number).

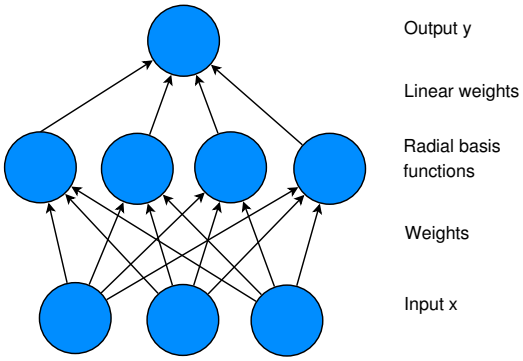
RBF network is a linear combination of RBFs. They are used in function approximation, time series prediction, and control. It is composed of three layers: an input, hidden and output layers connected by weights between layers. The output of the network is:

$$\varphi(x) = \sum_{i=1}^{I} w_i \phi(||\mathbf{x} - \mathbf{c}_i||) \tag{2.3.4}$$

where I is the number of neurons in the hidden layer, $\mathbf{c}_i$ is the center vector for neuron i and w_i are the weights of the linear output neuron. In the basic form all inputs are connected to each hidden neuron. The norm is typically taken to be the Euclidean distance and the basis function is taken to be Gaussian.

RBF networks are universal approximators on a compact subset of $\mathbb{R}^D$. This means that an RBF network with enough hidden neurons can approximate any continuous function with arbitrary precision. The weights w_i, $\mathbf{c}_i$, and β are determined in a manner that optimizes the fit between ϕ and the data (Moody and Darken, 1989). The next image illustrates the radial function network architecture:

Figure 2.5: Architecture of a radial function network

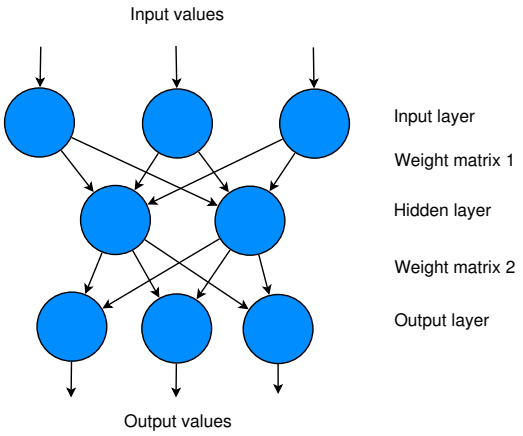


2.3.4 Multilayer perceptron

A multilayer perceptron (MLP) is a feedforward artificial neural network model that maps sets of input data onto a set of appropriate output. An MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. Except for the input nodes, each node is a neuron (or processing element) with a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training the network (Rosenblatt, 1962; Rumelhart et al., 1986). MLP is a modification of the standard linear perceptron and can distinguish data that is not linearly separable (Cybenko, 1992).

The architecture of an MLP is what follows:

Figure 2.6: Multilayer perceptron



MLP is composed of an input, hidden and output layers connected by weights between layers. Each node is a perceptron that is a binary classifier which maps its input (a real-valued vector) to an output value (a single binary value):

$$\varphi(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.3.5}$$

where w is a vector of real-valued weights, $w \cdot x$ is the dot product (which here computes a weighted sum), and b is the *bias*, a constant term that does not depend on any input value. The value of $f(x)$ (0 or 1) is used to classify x as either a positive or a negative instance, in the case of a binary classification problem. If b is negative, then the weighted combination of inputs must produce a positive value greater than $|b|$ in order to push the classifier neuron over the 0 threshold. Spatially, the bias alters the position (though not the orientation) of the decision boundary. The perceptron learning algorithm does not terminate if the learning set is not linearly separable. Learning occurs in the perceptron by changing connection weights after each piece of data is processed. It is based on the amount of error in the output compared to the expected result. This is an example of supervised learning, and is carried out through backpropagation, a generalization of the least mean squares algorithm in the linear perceptron.

MLP using a backpropagation algorithm is the standard algorithm for any supervised learning pattern recognition process and the subject of ongoing research in computational neuroscience and parallel distributed processing. They are useful in research in terms of their ability to solve problems stochastically, which often allows one to get approximate solutions for extremely complex problems like fitness approximation.

2.3.5 Support vector machine (SVM)

A support vector machine (SVM) is a supervised learning method that analyze data and recognize patterns, used for classification and regression analysis. SVM can be interpreted as an extension of the perceptron. The standard SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the input, making the SVM a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of the two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall into.

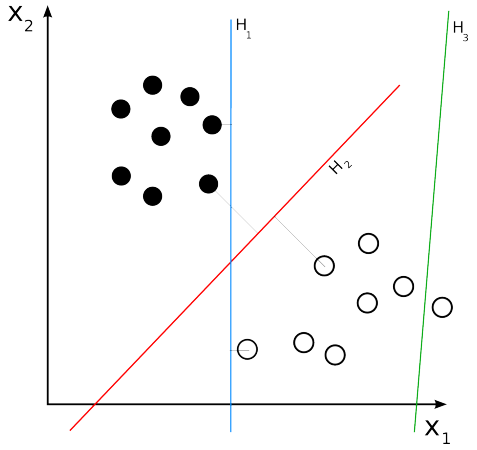
More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class, since in general terms the larger the margin the lower the generalization error of the classifier. The sets usually are not linearly separable, for this reason, it is proposed that the original space be mapped into a much higher-dimensional space. The vectors defining the hyperplanes can be chosen to be linear combinations with parameters of images of feature vectors that occur in the data base.

The support vector (SV) algorithm is a nonlinear generalization of the Generalized Portrait algorithm developed in Russia in the sixties (Vapnik and Lerner, 1963; Vapnik and Chervonenkis, 1964). Finally, it is firmly grounded in the framework of statistical learning theory which has been developed by Vapnik (1995).

There are SVMs for binary class, multiclass and regression. We focus on SVM for regression because we want to predict the TER values of the test set and this variable is continuous.

A version of SVM for regression is proposed in 1996 by Drucker et al. (1996). The model produced by support vector classification depends only on a subset of the training data, because the cost function for building the model does not care about training points that lie beyond the margin. Analogously, the model produced by SVM regression depends only on a subset of the training data, because the cost function for building the model ignores any training data close to the model prediction. An example of SVM (H3 (green) does not separate the two classes. H1 (blue) does, with a small margin and H2 (red) with the maximum margin):

Figure 2.7: Support Vector Machine (from Wikipedia)



2.3.6 Partial least squares regression (PLSR)

Partial least squares regression (PLSR) is a statistical learning method. The partial least squares (PLS) family of methods are known as bilinear factor models. Partial least squares Discriminant Analysis (PLS-DA) is a variant used when the predicted variables Y are binary.

Partial Least Squares (PLS) is used to find the fundamental relations between two matrices X and Y. A PLS model will try to find the multidimensional direction in the X space that explains the maximum multidimensional variance direction in the Y space of latent variables (or components).

The general underlying model of multivariate PLS is:

$$\begin{aligned} X &= TP^T + E \\ Y &= UQ^T + F \end{aligned} \tag{2.3.6}$$

- X is an $N \times D$ matrix of predictors, N represents the total number of features and D the number of samples.
- Y is an $N \times p$ matrix of responses, p is the number of responses.
- T and U are $N \times l$ matrices that are, respectively, projections of X and projections of Y, l is the number of loadings (the weight by which each standardized original variable should be multiplied to get the component score).
- P and Q are, respectively, $D \times l$ and $p \times l$ orthogonal loading matrices.
- E and F are the error terms. The decompositions of X and Y are made to maximise the covariance of T and U.

There are some variants of PLS for estimating the factor and loading matrices T, P and Q. The next variant is used in this master thesis. It constructs estimates of the linear regression between X and Y as (Lindgren et al., 1993; De Jong and Ter Braak, 1994; Dayal and MacGregor, 1997; de Jong, 1993; Rännar et al., 1994; Abdi, 2010):

$$Y = X\tilde{B} + \tilde{B}_0 \tag{2.3.7}$$

- X is a matrix of predictors (input variables). It is standardized.
- Y is a vector of responses (TER values in this case). It is standardized.
- $\tilde{B}$ is the regression matrix. It is computed with the optimum number of components.
- $\tilde{B}_0$ is the residual matrix.

2.4 Feature selection methods

It is known that feature selection can be useful in NLP, and even using learning methods that implicitly perform a feature selection, such as Support Vector Machines. By removing most irrelevant and redundant features from the data, feature selection helps improve the performance of learning models. The ideal selected group of features could be calculated testing all their combinations of them, but those experiments would have a high cost $O(2^D)$. Using feature selection methods allow to select features incrementally. Three feature selection methods are carried out, they are explained in the next subsections:

- Individual Performance-driven Selection (IS).
- Principal Component Analysis (PCA).
- Partial Least Squares Regression (PLSR).

2.4.1 Individual Performance-driven Selection (IS)

Individual performance-driven selection (IS) method creates subsets of increasing size with the best-scoring individual features. The calculation of the scoring of each feature comprises the results of the training the system using only one feature. In this way, the correlation between each feature and the prediction (TER value) is known, we measure the correlation with Pearson correlation coefficient explained in 3.3.2. The features that obtain best results are kept to train the model (Sanchis, 2004).

Despite selecting features according to their individual performance, correlations between different features are not taken into account. This could result in redundant features being still selected. Therefore, we also test other feature selecting techniques that solve this problem.

2.4.2 Principal Component Analysis (PCA)

Principal component analysis (PCA) is a mathematical procedure that uses an orthogonal transformation to convert a set of possibly correlated variables into a set of linearly uncorrelated variables called principal components (the number of principal components is less or equal than the number of features). The first principal component has the largest possible variability in the data and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. Principal components are guaranteed to be independent only if the data set is jointly normally distributed. PCA is sensitive to the relative scaling of the original variables.

The procedure of PCA is (Juan, 2011):

- Input:
  - Original feature space dimension: D
  - Unlabelled data set: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^D$
  - Reduced dimensionality to transform: $M < D$
- Output:
  - Linear transformation matrix: $U = (\mathbf{u}_1, \ldots, \mathbf{u}_M) \in \mathbb{R}^{D \times M}$
  - Projected data: $\tilde{\mathbf{x}}_n = U^t \mathbf{x}_n \; where \; n = 1, \ldots, N$
- Method:
  - Compute the sample mean and covariance matrix of the original data:

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_n \mathbf{x}_n \tag{2.4.1}$$

$$S = \frac{1}{N} \sum_n (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^t \tag{2.4.2}$$

  - $\mathbf{u}_1, \ldots, \mathbf{u}_M$ are the M eigenvectors of S with largest eigenvalues

PCA was developed in 1901 by Karl Pearson (1901). Now, it is mostly used as a tool in data analysis and for making predictive models. PCA can be done by eigenvalue decomposition of a data covariance matrix or singular value decomposition of a data matrix, usually after mean centering and normalizing the data matrix for each attribute (Abdi, 2010). The results of a PCA are usually discussed in terms of component scores (the transformed variable values corresponding to a particular data point), and loadings (the weight by which each standardized original variable should be multiplied to get the component score) (Shaw, 2003).

PCA method eliminates the problem of IS about the redundant features, but it does not take into account the correlation between the set of principal components and the prediction, which differs from our objective of predicting. Our goal is to predict the TER value, but this method cannot obtain the best-performing subset of features. For this reason, we test another method called PLSR that try to solve this problem.

2.4.3 Partial Least Squares Regression (PLSR)

Partial least squares regression (PLSR) is a statistical method explained in 2.3.6. PLSR bears some relation to principal components regression. Instead of finding hyperplanes of minimum variance between the response and independent variables, it finds a linear regression model by projecting the predicted variables and the observable variables to a new space explaining the maximum variance with predicted variables.

The PLS variant we use (explained in subsection 2.3.6) is defined as:

$$Y = X\tilde{B} + \tilde{B}_0 \tag{2.4.3}$$

- X is a matrix of predictors (input variables). It is standardized.
- Y is a vector of responses (TER values in this case). It is standardized.
- $\tilde{B}$ is the regression matrix. It is computed with the optimum number of components.
- $\tilde{B}_0$ is the residual matrix.

PLS is also able to provide information on the importance of individual features in X. An element of $\tilde{B}$ with large absolute value indicates an importance of the variable because it has been standardized previously, so the ordering (from highest to lowest) of the values in $\tilde{B}$ permits to select the best features of X. PLS is particularly indicated when the features in X are correlated with Y, where techniques like PCA usually fail (Rosipal et al., 2001). Morever, PLSR takes into account correlation of subsets of features, not just the individual correlation of each feature, unlike IS method. PLS has been used to extract qualitative information from different types of data (Frenich et al., 1995) and specifically for NLP (Specia et al., 2009a).

2.5 Summary

We have described the method proposed in this chapter. First of all, we have explained basic concepts and have introduced machine learning techniques to the reader. Secondly, we describe the different phases of the method. Thirdly, the phase of feature extraction is explained in more detail (all the groups of features are described). Then, we have listed and have given details about the classification methods tested and finally, we have also explained the features selection methods proven.

The procedure comprises the extraction of 156 features that are trained with different regression classifiers in order to train a model to predict the TER (Translation Error Rate) of a given translated sentence. In order to improve the method, a feature selection method is also applied. The user that use the method proposed, will be able to get automatically additional information about the quality of the translated sentence and classify it as good or bad translation.

Chapter 3

Experimental framework

3.1 Introduction

The experimental framework developed for testing the effectiveness of the method is described in the following sections. First of all, the dataset and evaluation measures used in the experiments are described.

Secondly, we compute experiments to select the methods that fit better. We compare number of features considered, classification methods and feature selection methods.

Thirdly, we explain the regression experiments where we calculate the regression error and we compare our results with other methods. Furthermore, we describe results in a translation application choosing TER thresholds and the desired number of post-edited sentences. As a result, we show the examples of use in appendix II. Moreover, we carry out an extra experiment removing anomalous data. Then, we describe the classification methods comparing TER values with human scores and detecting good and bad translated sentences. At last, we comment some conclusions of this chapter.

3.2 Dataset

The experiments have been tested with the dataset available and further described in Specia et al. (2010a).

The characteristics of the dataset are:

Table 3.1: Dataset (from Europarl)

Languages	EN-ES
Total sentences	4001
SMT systems	4
Human scores	1-4
Vocabulary	13411
Running words	564616

Each data point contains the following elements:

- Source (English) sentences.
- Reference (Spanish) sentences.
- 4 translations (Spanish) produced by each SMT system.
- Human scores for translations produced by each SMT system.

Each translator is produced by a different SMT system trained on Europarl:

- System 1: Matrax (Simard et al., 2005). It is a standard phrase-based SMT system, allows for gaps in phrases.
- System 2: Portage (Johnson et al., 2006). It is a standard phrase-based SMT system.
- System 3: Sinuhe (Kääriäinen, 2009). It is a phrase-based SMT system, but is not standard system by allowing phrases to overlap during decoding, and by training individual phrase weights applying a regularized conditional random fields on the full parallel aligned corpus.
- System 4: MMR (Maximum Margin Regression) (Saunders, 2008). It is a rather distinct approach to MT based on using predictions with structured output.

The reference translations (manually human productions without any MT system) provided by WMT08 (Callison-Burch et al., 2008) are used. The sentences are lowercased and tokenized. Each translation is also evaluated with a human annotation done by professional translators trained on the task and based on clearly defined guidelines about the interpretation of quality scores in the range $[1-4]$. The agreement of the evaluations is measured using the Kappa coefficient (Cohen, 1960) derived from three human judgements. This range $[1-4]$ is commonly used by professional translators to indicate the quality of translations with respect to the need for post-editing and means:

- 1 = requires complete retranslation.
- 2 = post editing quicker than retranslation.
- 3 = little post editing needed.
- 4 = fit for purpose.

The first experiments are focused on the translations of system 1. The training set is 75% of the each dataset and the remaining 25% is for testing randomly splitted and using a uniform distribution.

3.3 Evaluation measures

The automatic evaluation do not include humans to score the translation, it just include a software that execute an approximation of the human criterion. A metric is a measurement. One metric that evaluates machine translation output represents the quality of the output.

The quality of a translation is inherently subjective, there is no objective or quantifiable way to know if a translation is good. Therefore, any metric must assign quality scores so they correlate with human judgement of quality. That is, a metric should score highly translations that humans score highly, and give low scores to those humans. Human judgement is the benchmark for assessing automatic metrics, as humans are the end-users of any translation output.

The evaluation measures that are used to carry out the experiments of this project are:

- Translation measures: BLEU and TER.
- Regression error measures: RMSPE and Pearson.
- Classification error measures: CER, ROC curve and AUC.

These measures are explained in the following subsections.

3.3.1 Translation measures

BLEU

BLEU (Bilingual Evaluation Understudy) is an algorithm for evaluating the quality of text which has been machine-translated from one natural language to another. BLEU is not an error rate, is a score, i.e. the higher the BLEU score, the better.

BLEU measures the precision of unigrams, bigrams, trigrams, and four-grams with respect to a set of reference translations, with a penalty for too short sentences (Papineni et al., 2002). In practise, BLEU implements a geometrical average of n-gram precision. The consequence of this is that BLEU is often only well-defined at corpus level, but not at sentence level. If it is considered, for instance, a sentence of three words, such sentence will never share a common four-gram with the reference sentence, and BLEU will score zero even when the hypothesis produced by the system and the reference sentence are identical.

BLEU can be single or multi-reference, but in the present master thesis only single-reference BLEU will be used due to the corpus restrictions.

Quality is considered to be the correspondence between a machine's output and that of a human: the closer a machine translation is to a professional human translation, the better it is. It can take values between 0% and 100% or 0 and 1. In this master thesis it is scored as a percentage, the higher the value indicates that the translation is better. The metric is currently one of the most popular in the field.

TER

TER (Translation Error Rate) measures the effort of a user to correct MT output to make it a good translation. Therefore, the human post-edited version is considered the reference translation. This measure is defined as the minimum number of edits needed to change the MT output so that it matches exactly the reference, normalized by the length of the reference. Edits include insertion, deletion and substitution of single words, as any standard edit distance metric, as well as shifts of word sequences.

It can be multi-reference but in this master thesis it is used with a single-reference and it is represented as a percentage, but it can reach a value over 100. The formula that define TER is:

$$TER = \frac{\#edits}{\#reference\_words} \tag{3.3.1}$$

We use a tokenized and lowercased corpus to avoid more edits in TER due to punctuation and case. We have computed TER using a the software tex-tittercom by BBN Technologies and the University of Maryland (Snover et al., 2006)

3.3.2 Regression error measures

Root Mean Squared Prediction Error (RMSPE)

Root Mean Squared Prediction Error (RMSPE) is used to measure the performance of the confidence measure of the translation in a system. RMSPE compute the average error in the estimation of TER scores. The formula that define RMSPE is:

$$RMSPE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{3.3.2}$$

N is the number of test sentences, $\hat{y}$ is the TER predicted by the learning algorithm and y is the actual TER value for that test case. RMSPE calculate the root of the average deviation of the estimator with respect to the expected score: the lower the value, the better the performance of the confidence measure system.

Pearson correlation coefficient

Pearson correlation coefficient (Pearson) is used to evaluate the performance of the confidence measure system. Pearson is computed between the predicted score $\hat{y}$ and the expected score y. Pearson measures their linear dependence and is defined as the covariance of these two variables divided by the product of their standard deviations, giving a value between +1 and -1 inclusive. It is independent of the scale of variables measurement. The higher its absolute value, the better performance of the confidence measure system. This metric is commonly used for the measurement of the machine translation evaluation metrics.

$$Pearson = \frac{\sum_{i=1}^{N}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}\sqrt{\sum_{i=1}^{N}(\hat{y} - \bar{\hat{y}})^2}} \tag{3.3.3}$$

N is the number of test sentences, $\hat{y}$ is the TER predicted by the learning algorithm, y is the actual TER value and $\bar{y}$ is the average value for that test case.

3.3.3 Classification error measures

Classification Error Rate (CER)

The Classification Error Rate (CER) is used in a binary classifiers. CER is defined as the number of errors made divided by the total number of test samples (we use the percentage):

$$CER = \frac{\#errors}{\#total} \cdot 100 \tag{3.3.4}$$

ROC curve

Receiver Operating Characteristic (ROC) curve is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. It plots the fraction of true positives out of the positives or True Positive Rate (TPR), also called *sensitivity* vs. the fraction of false positives out of the negatives or False Positive Rate (FPR), also called *1-specificity* at various thresholds. TPR and FPR are defined as:

$$\begin{aligned} TPR &= \frac{TP}{P} = \frac{TP}{TP + FN} \\ FPR &= \frac{FP}{N} = \frac{FP}{FP + TN} \end{aligned} \tag{3.3.5}$$

The concepts are explained as follows:

- Positive (P) is one of the classes of the binary classifier system.
- Negative (N) is the other class of the binary classifier system.
- True Positive (TP) is a sample that is actually P and the prediction is also P.
- False Positive (FP) is the actual value is N but the prediction is P.
- True Negative (TN) is when the prediction and the actual value are N.
- False Negative (FN) is when the prediction is N but the actual value is P.

There are three principal cases in a ROC curve:

- The best case: The value of TPR is equal to al the possible values of FPR. This means that all the samples can be classified appropriately.
- The worst case: when $TPR = FPR$ for all possible threshold.
- Usual case: the ROC curve is between the best case and the worst case.

The system is better when is nearer to the best case and the most interesting zone of the curve is when FPR is more aproximate to 0. If the FPR value is elevate, the system could be not useful.

Area Under Curve (AUC)

Area Under Curve (AUC) is defined as the area under the ROC curve divided by the area under the ROC curve of the worst case. AUC, when using normalized units, is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one.

We take values from 0% to 100% and the value 0% means that the model is the worst case and 100% the best case. If we compare two systems and the value of the area of the first system is higher than the area of the second one, the first system is better than the other one.

3.4 Testing of the method

The first experiments are executed to decide the number of features, the classification method and the feature selection method that fit better with results.

The experiments undertaken in this section are:

- Comparing number of features: are done just with the first features and gradually more features are considered. In these experiments we test the performance of the use of distinct number of features. They are processed using SVM (explained in 2.3.5).

- Comparing classification methods: testing different types of classifiers in order to find the classifier which fits better.

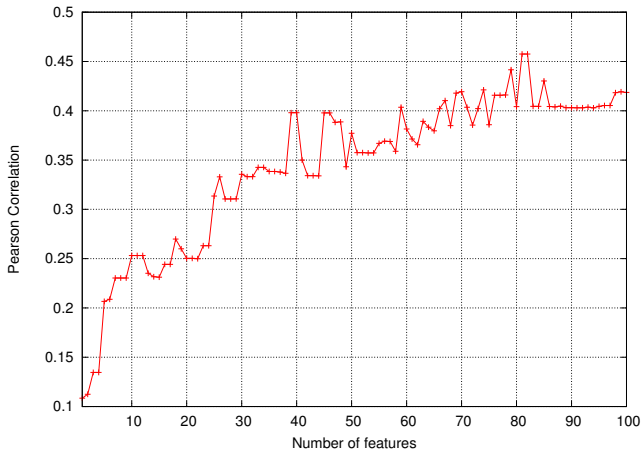
- Comparing feature selection methods: we test several feature selection methods to obtain the best results. It is known that if you give to the classifiers the optimized features the samples will be better classified.

3.4.1 Comparing number of features

The preliminary experiments have been processed with Support Vector Machines (SVM) for regression taking the options epsilon-SVR algorithm with radial basis function kernel from libSVM library optimizing its parameters using downhill simplex (J. A. Nelder, 1965). For these experiments we apply cross-validation using five random subsamples of the training set. Moreover, we apply a standard normalization to the data. We have used SVM because other similar works (Specia and Farzindar, 2010) train the models with SVM but in the next subsection we prove more classifiers.

In the next figure, results for 1 to 100 number of features in cross-validation are shown:

Figure 3.1: Results in cross-validation comparing number of features



We can see that the growth in the number of features improves the results. Despite that, the last features do not improve the results. For this reason, we believe that a selection feature method can improve the results selecting best features and removing the features that do not correlate well with TER (the value we want to predict). Moreover, we think that other classifiers can fit better.

3.4.2 Comparing classification methods

We have tested in cross-validation some classifiers (SVM, PLSClassifier, RBFNetwork, Decision Stump Tree, Simple Linear Regression and linear regression). LibSVM classifier is executed from libSVM software (chung Chang and Lin, 2001) and the rest of classifiers are executed from the software Weka (Hall et al., 2009). All these classifiers are explained in section 2.3.

The next table shows the results:

Table 3.2: Results in cross-validation comparing classification methods

Classifier	**RMSPE**	**Pearson**
libSVM	0.26	0.30
PLSClassifier	0.26	0.38
MultilayerPerceptron	0.34	0.27
RBFNetwork	0.28	0.06
Decision Stump Tree	0.28	0.16
Simple Linear Regression	0.27	0.21
SVMreg	0.27	0.35
Linear Regression	0.26	0.37

We can see that the PLSClassifier obtains the best results in Pearson Correlation and ties with libSVM and Linear Regression in RMSPE. The good results for PLSClassifier could be because this classifier provides more predictive accuracy and a much lower risk of chance correlation than others classifiers (Cramer, 1993). After having this result, we have chosen PLSClassifier as the best classifier method that we have tested and we use it for the next experiments.

3.4.3 Comparing feature selection methods

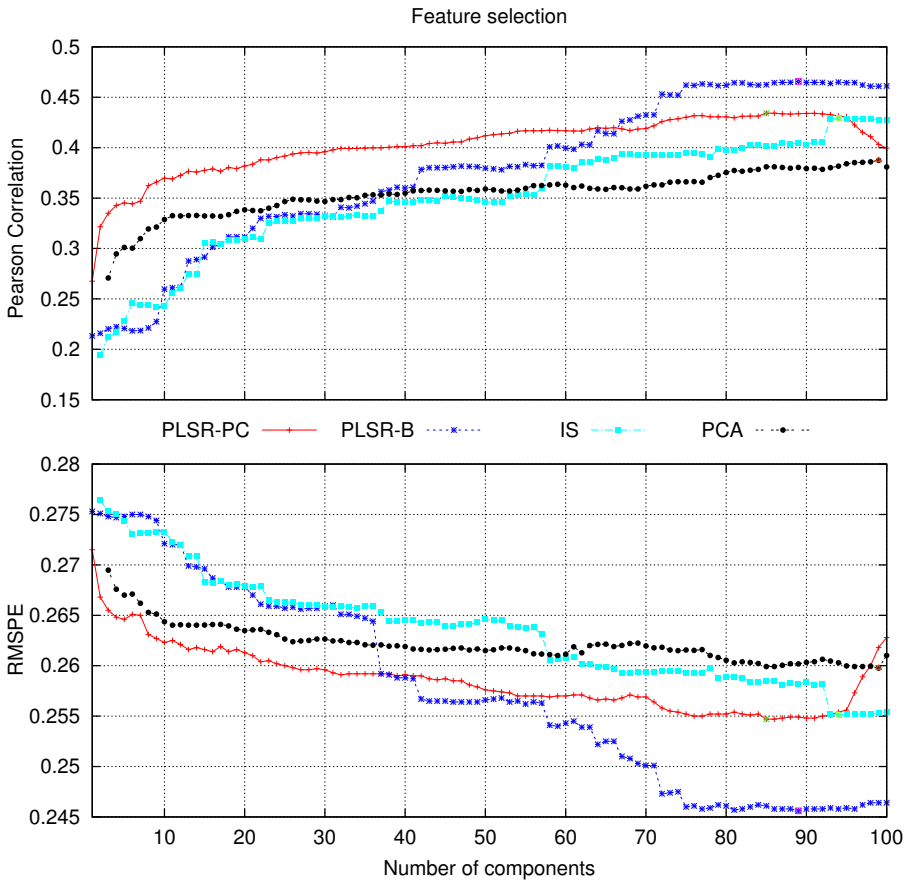
In order to improve the system results, we make a study of feature selection methods. It is known that if the model is trained with a optimized number of features and the features that correlate better with the desired predicted value the results will be better.

Different feature selection methods (explained in detail in section 2.4) have been tested:

- Individual Performance-driven Selection (**IS**) has been executed programming scripts that calculate Pearson Correlation Coefficient of each separated feature.
- Principal Component Analysis (**PCA**) has been executed with a script in R (R Development Core Team, 2008) using the function *eigen*. The features are reduced to n components in a new space.
- Partial Least Squares Regression (**PLSR**). We have applied two techniques with PLSR:
  - One of them is just changing the number of components that are used to reduce the dimensionality, we call it principal components PLSR (**PLSR-PC**).
  - The other one consists, in addition to the above, of using the sorted features in the following way: PLSR model is trained in Matlab (MATLAB, 2012); the matrix called *PCTVAR* informs about the relevance components to predict the output in its second row; taking for account the relevant components, in the Matrix B we can see the best features calculating the average of the components values. Then, the features are sorted from the major absolute value to the minor absolute value. We call this technique **PLSR-B**.

In the next figure, we can see the results of RMSPE and Pearson Correlation Coefficient changing the number of features used for the different feature selection methods:

Figure 3.2: Results in cross-validation comparing feature selection methods



The best points to each methods for Pearson Correlation are:

Table 3.3: Best points feature selection methods

Method	**RMSPE**	**Pearson**
PLSR-B	0.25	0.47
PLSR-PC	0.25	0.43
IS	0.25	0.43
PCA	0.26	0.39

The results show that the best feature selection method is **PLSR-B** because reaches the best value in Pearson and RMSPE. Despite that, using few components until 35 components, the best technique would be **PLSR-PC** in RMSPE and until 65 components in Pearson. **PCA** is better than **IS** for few number of components (until 55 components) and **IS** is better than **PCA** using more number of components and yields better result.

In any case, PLSR is the best technique according to the results. The reason is because this method reduces the dimensionality explaining the maximum variance with predicted variables (unlike PCA that reduces dimensionality but does not consider the prediction) and considers correlation within subset of features (unlike IS that just consider correlation of each feature individually).

The first 10 features selected with PLSR (the best method) are some of the extracted from 1000-best, some number of mismatches punctuation symbols, the number of names in the target sentence and the LM probability of the unigrams in the target sentence.

We also tested the feature selection techniques: Linear discriminant analysis (LDA) and Canonical correlation analysis (CCA). The results have not been good because these two last techniques are difficult to apply in regression approaches.

3.5 Regression experiments

In the regression experiments we evaluate the regression error that we have obtained. We comment the results for each dataset and compare the method with other confidence measures methods. Moreover, we show the results in a translation application thresholding the TER or the desired number of post-edited sentences. Lastly, we describe an extra experiment removing anomalous data.

We consider the model with which we have obtained best results (PLSR optimized in number of features and components).

3.5.1 Comparison with other methods

In this section, we comment the results for each dataset and compare the method with other confidence measures methods.

In the next table, we can see the results for each dataset of the corpora (represented on the table with S+number of dataset). We have compared with the results in Specia and Farzindar (2010) using the same corpora and partition test:

Table 3.4: Results comparing with Specia and Farzindar (2010)

Dataset	**Method**	**RMSPE**	**Pearson**
En-Es Europarl S1	Specia and Farzindar (2010)	0.18	0.33
	Method proposed	0.22	0.44
En-Es Europarl S2	Specia and Farzindar (2010)	0.17	0.39
	Method proposed	0.21	0.49
En-Es Europarl S3	Specia and Farzindar (2010)	0.17	0.36
	Method proposed	0.20	0.50
En-Es Europarl S4	Specia and Farzindar (2010)	0.14	0.40
	Method proposed	0.18	0.49

We can see that the method proposed is better than (Specia and Farzindar, 2010) in Pearson correlation and worse in RMSPE. Pearson correlation is a metric more reliable than RMSPE because is independent of the scale of the confidence measure system.

It has to be considered that the TER values are computed using reference translations as opposed to post-edited translations so they may not reflect post-editing effort appropriately.

As in Specia and Farzindar (2010), we can compare our approach with other criteria used to select good translations from post-editing:

- *Size*: the size of the source sentence in words. Long sentences are likely to be incorrectly translated.

- *LM*: trigram language model score of the source sentence using the source of the SMT traning corpus. Common segments in the training corpus are likely to be well translated.

We also have compared our method with the method in Specia and Farzindar (2010). The results for the dataset S1 are:

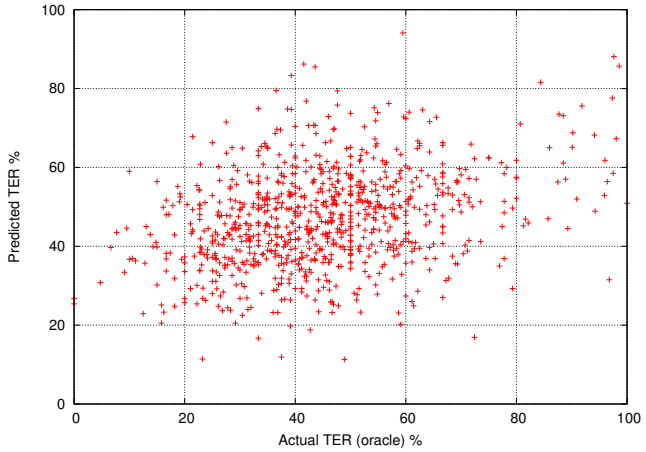
Table 3.5: Comparing confidence measures methods

Method	**Pearson**
Size in words	0.09
LM	0.14
Specia and Farzindar (2010)	0.33
Method proposed	0.44

The results show that our method has obtained better performance than the other methods. We can see that the methods Size and LM are very simple and method in (Specia and Farzindar, 2010) and our method obtain better results because they have taken into account more useful features.

In the next figure is presented the predicted TER on the $x - axis$ and the actual TER (oracle, obtained with the reference of the test samples) on the $y - axis$ of the test samples.

Figure 3.3: Comparing actual TER (oracle) and predicted TER



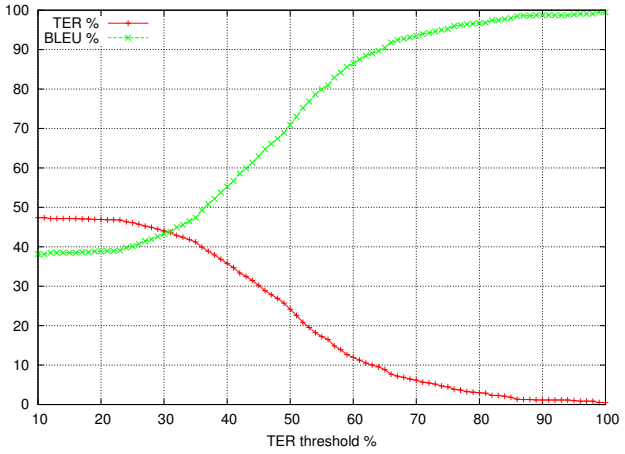
We can see a tendency of the predicted values to follow the actual values, there is not a diagonal line from the point $0-0$ to $100-100$ but there are not samples in the corners, near the points $0-100$ and $100-0$. Moreover, values from 40% to 60% in TER are better predicted, and they are very similar to the oracle. If the method is improved with more features, try more options (other classifiers, selection features methods) and exhaustive adjustment of parameters the results could be better.

3.5.2 Results in a translation application

There are several alternatives to select translations, one of them could be establishing thresholds on the edit distance or TER. We could then check which of the thresholds is the best to select the largest number of translated sentences with the lowest effort and the minimum possible error.

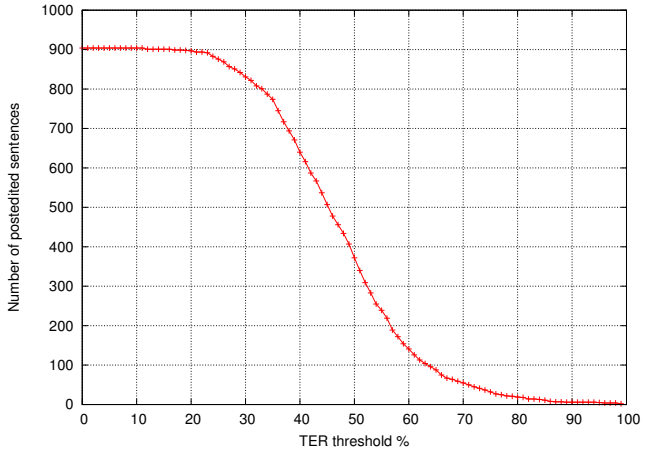
In the next figure is shown the TER and BLEU obtained for TER thresholds from 0% to 100%. The translated sentences that have the predicted TER higher than the TER threshold are considered as bad translations, the rest of the translated sentences are considered as good translations. The TER and BLEU are computed changing the selected sentences as bad translation for the reference sentences to try to simulate the correction of the translation or post-edition.

Figure 3.4: Results for TER and BLEU changing the TER threshold



The next figure shows the results for the number of post-edited sentences for each TER threshold from 0% to 100%:

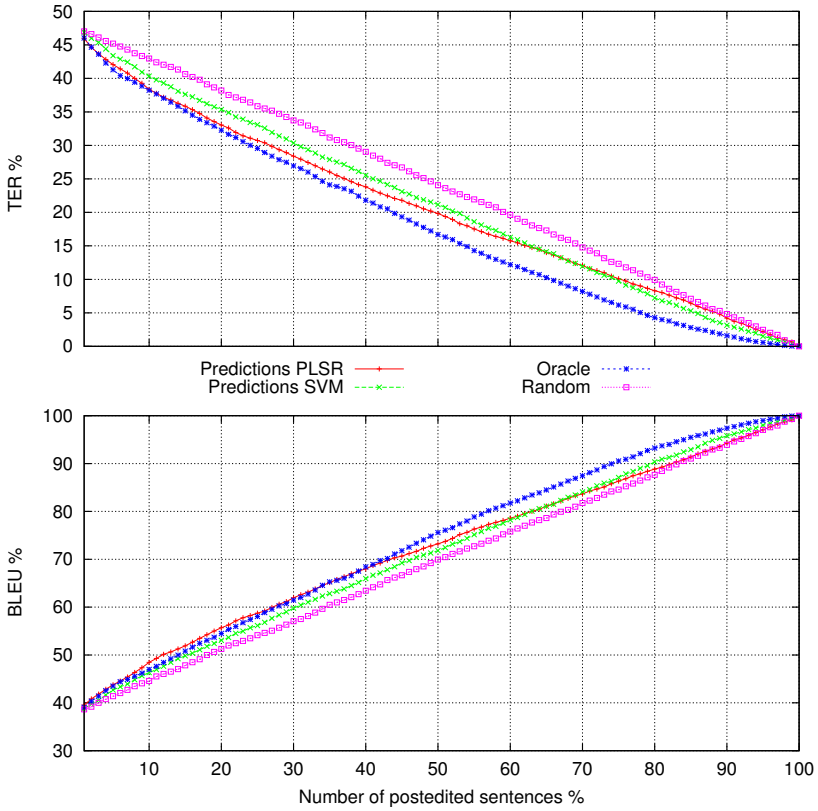
Figure 3.5: Results for number of sentences post-edited changing the TER threshold



We can see that in the TER threshold 60% there is a tendency to decrease the error of BLEU and TER and the effort to post-edit less sentences.

Other alternative to select translations is to establish the percentage of translated sentences wanted to be postedited. The next figure reflects the results of TER and BLEU for this criteria:

Figure 3.6: Results comparing number of post-edited sentences



The figure represents the results for the test samples for the oracle (actual values of TER), random values and predictions with the models using SVM and PLSR with the parameters optimized and features selected. We can see that both methods (PLSR and SVM) are better than the random one because they are more aproximate to the oracle. Moreover, PLSR method is better than SVM to values from 0% to 65% of post-edited sentences and SVM is better from 65% to 100%. The best threshold for PLSR method that seems to be the best method is for TER 10% and for BLEU from 30% to 40% of post-edited sentences.

It is worth noting that for values from 0% to 20% of post-edited sentences, the PLSR method has practically equal values than the TER oracle. The better results (measured in BLEU) that obtained PLSR with respect to the oracle can be explained given that the oracle is based on TER.

Moreover, the appendix II shows examples of sentences with well predicted and the first 21 sentences of the test set used using a threshold of 50% of TER. The sentences in bold represent the sentences whose error is higher than the threshold indicated; these sentences would be the worst translated.

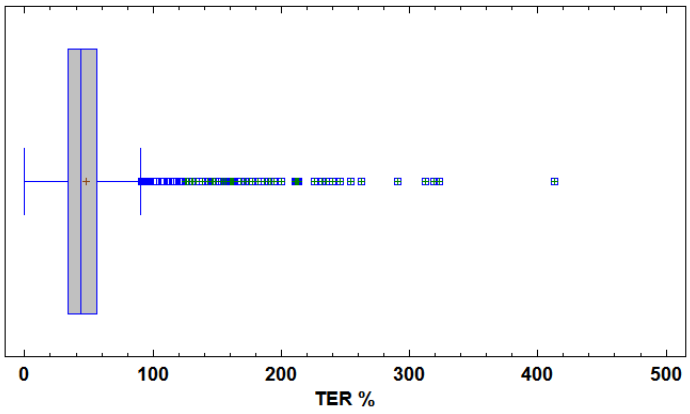
3.5.3 Removing anomalous data

We have analyzed the training data and have detected the anomalous and super anomalous data according to its TER value. We have constructed a box and whisker graphic.

A box and whisker graphic is based on quartiles and permits to visualize a dataset. It is composed of a rectangle, the box and two arms or whiskers. This graphic shows information about the minimum and maximum values, the quartiles Q1, Q2 or median (represented in the middle of the box) and Q3, and about the existence of anomalous values and the symmetry of the distribution.

The next figure shows the box and whisker graphic for the data used:

Figure 3.7: Anomalous training samples according to the actual TER



We can see that the normal data are between values 0% and 90% of TER, the anomalous data are between 90% and the maximum (420%) and super anomalous (the furthest data from the median) data are between 130% and the maximum (420%).

We have experimented removing just the super anomalous and removing anomalous samples in the partitions of cross-validation and the results are:

Table 3.6: Results removing anomalous data in cross-validation

Data	**RMSPE**	**Pearson**	**Number of samples**
Without anomalous	0.14	0.49	2963
Without super anomalous	0.17	0.48	3043
All samples	0.25	0.47	3095

The best results are for the data without anomalous samples and data without super anomalous is better than data with all samples.

We have tested with the test samples and the results are:

Table 3.7: Results removing anomalous data in test

Data	**RMSPE**	**Pearson**
Without anomalous	0.24	0.28
Without super anomalous	0.23	0.34
All samples	0.22	0.44

We can see that the results for the test data are worse removing anomalous and super anomalous samples than with all samples. This is because in cross-validation the model has been overtrained, possible for the limited training data.

3.6 Classification experiments

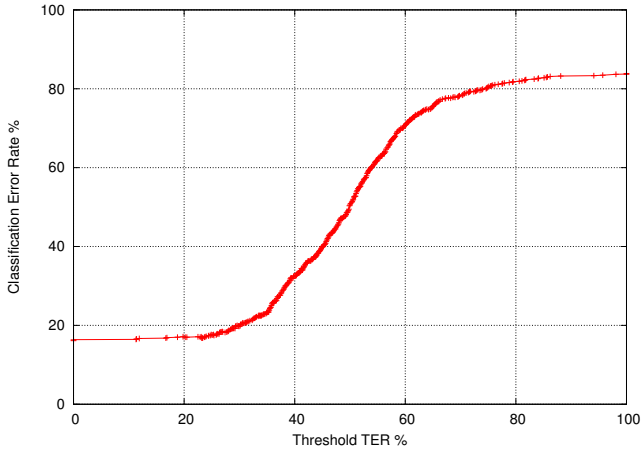
The dataset used also provides scores assigned by professional translator to each translated sentence. The translators have scored the translations as:

- 1 = requires complete retranslation.
- 2 = post editing quicker than retranslation.
- 3 = little post editing needed.
- 4 = fit for purpose.

We have used these scores to check whether our predictions are good. For this purpose, we consider that sentences scored from 1 to 3 are considered as bad translations and sentences scored with 4 are considered as good translation.

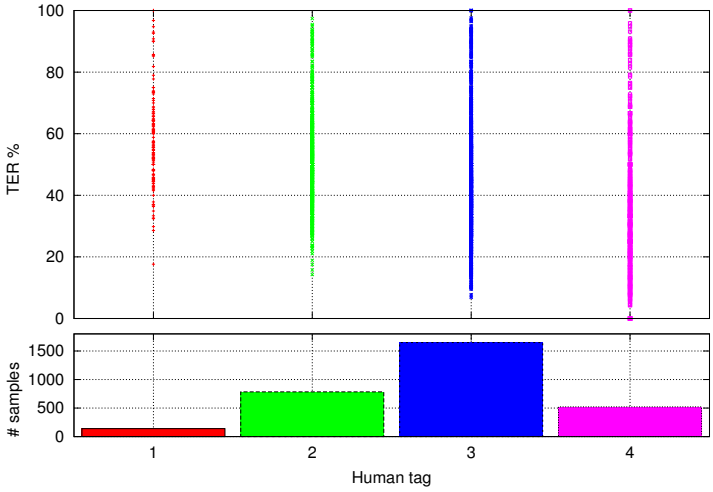
As previous experiments, we have considered TER thresholds. The sentences which prediction is higher than the threshold are considered as bad translations and sentences with lower value are considered as good translations. We have compared these results with the human scores and have calculated the Classification Error Rate (CER). The results are illustrated in the following figure:

Figure 3.8: CER



The best TER threshold is 0%, that is a bad result because involves that all the samples are considered as bad translations and the method would not be useful. The reason for this bad result, is that the data are not balanced with respect to the human scores. In other words, the number of the samples of each score or tag is out of balance. Moreover, the criteria of the TER threshold is not very correlated with the scores. The next figure shows the dispersion of the human scores or tags of the training samples, the bottom figure reflects the number of samples of each tag and the upper figure compares the TER value and the human tags:

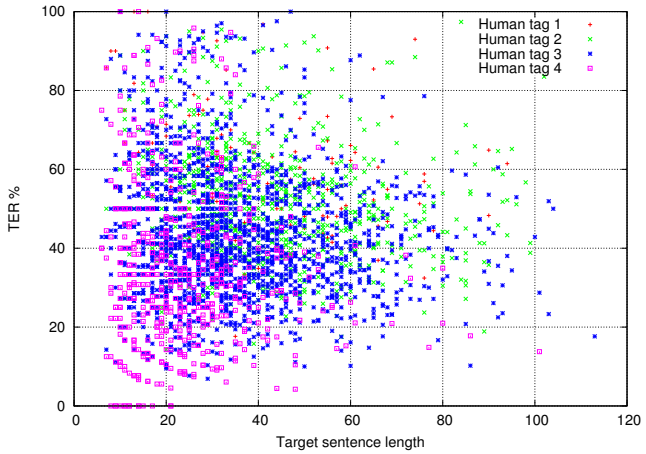
Figure 3.9: Human tags dispersion



There are very few samples of the score 1 and a lot of them for score 3. We also can see that it is true that the samples with less calification (1 and 2 scores) have more TER and 3 and 4 scores have less TER but it is not clearly defined, there are samples of score 3 that have values of TER from 10% to 100% and some samples of score 4 that have TER 100% or near 100%.

Furthermore, the next figure compares the TER values and target sentence length of the training samples. We have chosen the target sentence length because there is a simility between this value and TER (if the sentence is long there are more possibilities that the translation error could be worse).

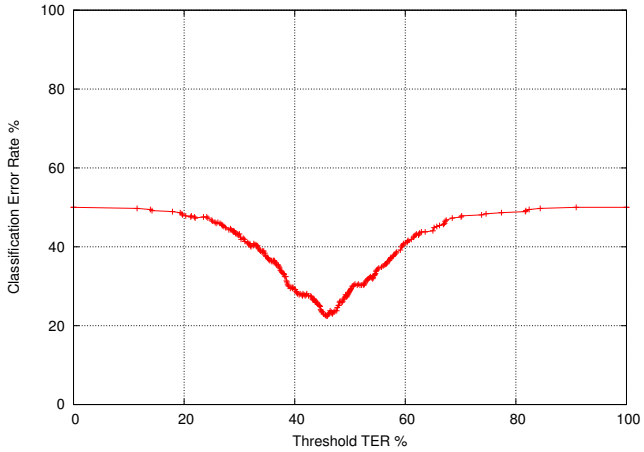
Figure 3.10: Human tags vs. target sentence length



We can see that the majority of the samples of scores 2, 3 and 4 have values of TER from 0% to 50%, 20% to 60% and 50% to 70%, respectively. These scores are clustered in their respective values but they are also samples that have a TER value out of this range. In addition, score 1 has not a defined range, it has samples in all the values of TER.

We decided to balance the data having the same samples for each class (good translated sentences would have the same samples as bad translated sentences). The problem of the criteria chosen (just score 4 for good translations) is that there are very few samples of this score so we cannot train with such a few samples. So we choose the values 1 and 2 as bad translated samples, and 3 and 4 as good translated samples with the same number of good and bad samples for balance data and have the most number of samples to train. Moreover, we separate a partition, also balanced, for testing purposes. The figure below shows the results for Classification Error Rate (CER):

Figure 3.11: CER with balanced data



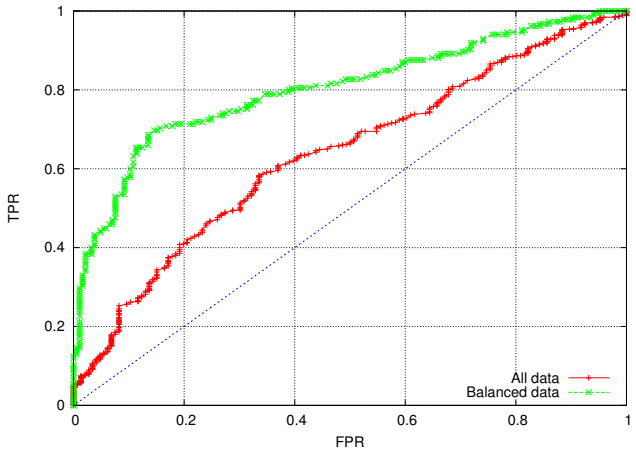
Having these results, we can select the best TER threshold for this experiments. As the figure shows the best TER threshold is 46%.

We also have calculated the *True Positive Rate* (TPR) and *False Positive Rate* (FPR) and have constructed a *Receiver Operating Characteristic* curve or ROC curve (explained in 3.3.3). Moreover, we have calculated the AUC (explained in 3.3.3) of the ROC curve.

If we compare two systems and the value of the area of the first system is higher than the area of the second one, the first system is better than the other one.

The ROC curve for the system with all (unbalanced) data and with the balanced data is:

Figure 3.12: ROC curve



The AUC values for both systems are:

Table 3.8: Area Under Curve (AUC) results

System	**AUC %**
Unbalanced data	63.64%
Balanced data	80.31%

As we can see in the results, both systems are higher than the diagonal line (the worst case) and the area of the system with balanced data is higher than the area of the system with all (unbalanced) data.

We have also proven other criteria to select good and bad translations as in Specia and Farzindar (2010). This criteria consists of considering samples scored with 2 to 4 as good translated and samples with the score 1 as bad translated considering that is better to remove bad translated sentences and translate them just with the source and the good translations should be kept for post-editing. The results, as expected, are similar to the criteria we have chosen (just score 4 as good translated sentences).

3.7 Summary

We have proven in the experiments that features used are useful to predict the TER value of a translated sentence. Moreover, we have tested several classification and features selection methods to improve the method and we have selected the best methods to test the test samples. We have detected that the classifier PLSR is the one which fits better and the best feature selection method is also PLSR. We have got better results with these improvements.

We also have tested regression experiments comparing our results with other methods, we have obtained better results in correlation. Moreover, we have described results in a translation application selecting thresholds of TER and number of post-edited sentences. As a result, we have shown examples of use in the appendix II. Furthermore, we have tested an extra experiment removing anomalous data but we have detected that in cross-validation the data are overtrained and the results with the test data are not satisfactory.

Finally, we have compared TER predictions with human scores of the sentences samples (classification experiments) and we have seen that the dataset is unbalanced with respect to the human scores. We conclude that this comparison is hard to be carried out.

Chapter 4

Conclusions

We have presented an approach to estimate the translator's effort to post-edit a translated sentence. In order to help and save time to the translators we predict a value to each given translated sentence to facilitate its post-edition. Moreover, users can indicate a threshold selecting sentences as good or bad translations, in this manner, users can take to ways:

- Keeping the translated sentences without post-edition saving translator's time and post-editing or removing bad translated sentences.
- Removing bad translations saving time post-editing bad translations (so bad that is better to translate without the SMT translation) and post-editing just the good translations.

We have applied other way to train a model. Instead of training with the human annotations, we have trained with the TER (Translation Error Rate obtained calculating number of edits to correct the translation) of each translated sentence. Using this approach, effort and money of humans tagging the samples are saved.

On chapter 1, we have made an introduction to the method proposed, on chapter 2 we have explained the techniques applied and on chapter 3 we have shown the experiments executed.

Summarising this master thesis, we have extracted 156 features just given the source and translation (like a "black-box") of the translated sentences we want to predict their quality. In addition, we have study different classifiers and feature selection methods (selection of the best features) to better train a model to predict the TER or error value of given translated sentences.

We have obtained that the classifier which better fits is PLSR and the feature selection method is also PLSR. These good PLSR results could be derived from the fact that it provides more predictive accuracy and a much lower risk of chance correlation than other classifiers. Moreover, feature selection takes into account correlation of subgroups of features and correlation with the prediction.

In conclusion, the results obtained show that the method proposed achieves a significant benefit in terms of user effort post-editing translations. As we have seen, the use of the CM method makes the results better in Pearson correlation than other similar methods. The prediction of the error (TER) improves the results of data that are processed with the method proposed like randomly selection of the best translated sentences. Moreover, we have executed an extra experiment removing anomalous data. It was not good because the model was overtrained.

Furthermore, we have carried out classification experiments comparing the results about TER prediction and human scores taking into account two classes: good and bad translated sentences. It was not good because the data are unbalanced with respect to the human scores.

The utility of the method proposed is considerable. For example, in post-edition translation platform it is possible to add this method giving the predicted value to each translated sentence. In this way, the user can easily see the predicted error of the sentence. This predicted value is similar to other common measure used in translation framework known as *fuzzy match threshold* that gives a degree of match between a source document segment and a translation memory segment. Other utility is that, if the user wanted, it could indicate a threshold of quality to select sentences as good or bad translations. For instance, good translations could be coloured with green and bad translation with red.

4.1 Future work

The method proposed could be improved on taking the following items:

The future work is based on the integration of the method to an actual translation platform, for example, in a translator's company. Moreover, the method could be improve taking into account:

- Evaluation of the method with experts and improve execution time:
  - Train with human scores to see if the results are good as other alternative for corpus that have the scores available.
  - Evaluate the method with experts instead of using the reference.
  - Optimize software for execution time.
- Test the method with new aspects:
  - Test the method using other pairs of languages.
  - Test other corpora.
  - Use a corpus with several references instead of just one reference.

- Add new characteristics to the method and improve it doing:
  - Calculate more features. Also, the features can be combined extracting more features, like multiplying them. Moreover, other combination of word-level sentences can be taken to get the sentence-level as ranking the words. We just calculate the average of the values per word in each sentence.
  - Calculate the confidence measure per word. This could be using the word-level extracted features.
  - Predict more values, for example, the modified BLEU for a sentence.
  - Prove more classifiers. For instance, M5P decision tree, we have taken promising results in preliminaries experiments with this classifier.

It is worth noting, that executing the proposed improvements the results would be better.

Chapter 5

Appendix I: List of features

Sorted list with the description of each extracted feature:

1. Number of tokens in the source sentence.
2. Number of tokens in the target sentence.
3. Average source token length.
4. LM probability of source sentence.
5. LM probability of the target sentence.
6. Average number of occurrences of the target word within the target sentence.
7. Average number of translations per source word in the sentence, as given by probabilistic dictionaries produced by GIZA++ (Och and Ney, 2000) extracted from the parallel corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010), thresholded using percentage 0.2.
8. Average number of translations per source word in the sentence, as given by probabilistic diccionaries produced by GIZA++ (**?**) extracted from the parallel corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010), thresholded using percentage 0.01 weighted by the inverse frequency of the words in the source sentence.
9. Percentage of unigrams in quartile 1 of frequency (lower frequency words) in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
10. Percentage of unigrams in quartile 4 of frequency (higher frequency words) in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
11. Percentage of bigrams in quartile 1 of frequency of source words in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
12. Percentage of bigrams in quartile 4 of frequency of source words in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
13. Percentage of trigrams in quartile 1 of frequency of source words in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
14. Percentage of trigrams in quartile 4 of frequency of source words in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
15. Percentage of unigrams in the source sentence seen in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).

16. Number of punctuation marks in the source sentence.
17. Number of punctuation marks in the target sentence.
18. Source sentence length.
19. Target sentence length.
20. Ratio source target lengths.
21. Average source word length.
22. Source perplexity model probabilities of the unigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
23. Source perplexity model probabilities of the bigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
24. Source perplexity model probabilities of the trigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
25. Target perplexity model probabilities of the unigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
26. Target perplexity model probabilities of the bigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
27. Target perplexity model probabilities of the trigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
28. Source probability model probabilities of the unigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
29. Source probability model probabilities of the bigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
30. Source probability model probabilities of the trigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
31. Target probability model probabilities of the unigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
32. Target probability model probabilities of the bigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
33. Target probability model probabilities of the trigrams in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010).
34. Target sentence trigram language model probability trained on a POS-tagged (extracted with the software Freeling (Padró, 2011)) corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the target language extracted with the software Freeling (Padró, 2011).
35. Average frequency of unigrams in the source sentence belonging to the first quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
36. Average frequency of unigrams in the source sentence belonging to the second quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
37. Average frequency of unigrams in the source sentence belonging to the third quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
38. Average frequency of unigrams in the source sentence belonging to the fourth quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
39. Average frequency of bigrams in the source sentence belonging to the first quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
40. Average frequency of bigrams in the source sentence belonging to the second quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.

41. Average frequency of bigrams in the source sentence belonging to the third quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
42. Average frequency of bigrams in the source sentence belonging to the fourth quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
43. Average frequency of trigrams in the source sentence belonging to the first quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
44. Average frequency of trigrams in the source sentence belonging to the second quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
45. Average frequency of trigrams in the source sentence belonging to the third quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
46. Average frequency of trigrams in the source sentence belonging to the fourth quartile of corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source language.
47. Percentage of distinct unigrams in the source sentence seen in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source.
48. Percentage of distinct bigrams in the source sentence seen in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source.
49. Percentage of distinct trigrams in the source sentence seen in the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the source.
50. Percentage of punctuation symbols in the source sentences.
51. Percentage of punctuation symbols in the target sentences.
52. Ratio of punctuation symbols between source and target sentences.
53. Percentage of numbers in the source sentences.
54. Percentage of numbers in the target sentences.
55. Ratio of numbers between source and target sentences.
56. Percentage of stop words in the source sentences.
57. Percentage of stop words in the target sentences.
58. Ratio of stop words between source and target sentences.
59. Number of mismatching opening/closing brackets in the target sentences.
60. Whether target sentence contains mismatched quotation marks.
61. Number of mismatches of numbers between the source and target sentences in absolute terms.
62. Number of mismatches of numbers between the source and target sentences normalized by sentence length.
63. Number of mismatches of the symbol ! between the source and target sentences in absolute terms.
64. Number of mismatches of the symbol ! between the source and target sentences normalized by sentence length.
65. Number of mismatches of the symbol # between the source and target sentences in absolute terms.
66. Number of mismatches of the symbol # between the source and target sentences normalized by sentence length.
67. Number of mismatches of the symbol ” between the source and target sentences in absolute terms.

68. Number of mismatches of the symbol ” between the source and target sentences normalized by sentence length.
69. Number of mismatches of the symbol % between the source and target sentences in absolute terms.
70. Number of mismatches of the symbol % between the source and target sentences normalized by sentence length.
71. Number of mismatches of the symbol $ between the source and target sentences in absolute terms.
72. Number of mismatches of the symbol $ between the source and target sentences normalized by sentence length.
73. Number of mismatches of the symbol ’ between the source and target sentences in absolute terms.
74. Number of mismatches of the symbol ’ between the source and target sentences normalized by sentence length.
75. Number of mismatches of the symbol & between the source and target sentences in absolute terms.
76. Number of mismatches of the symbol & between the source and target sentences normalized by sentence length.
77. Number of mismatches of the symbol) between the source and target sentences in absolute terms.
78. Number of mismatches of the symbol) between the source and target sentences normalized by sentence length.
79. Number of mismatches of the symbol (between the source and target sentences in absolute terms.
80. Number of mismatches of the symbol (between the source and target sentences normalized by sentence length.
81. Number of mismatches of the symbol + between the source and target sentences in absolute terms.
82. Number of mismatches of the symbol + between the source and target sentences normalized by sentence length.
83. Number of mismatches of the symbol * between the source and target sentences in absolute terms.
84. Number of mismatches of the symbol * between the source and target sentences normalized by sentence length.
85. Number of mismatches of the symbol - between the source and target sentences in absolute terms.
86. Number of mismatches of the symbol - between the source and target sentences normalized by sentence length.
87. Number of mismatches of the symbol , between the source and target sentences in absolute terms.
88. Number of mismatches of the symbol , between the source and target sentences normalized by sentence length.
89. Number of mismatches of the symbol / between the source and target sentences in absolute terms.
90. Number of mismatches of the symbol / between the source and target sentences normalized by sentence length.
91. Number of mismatches of the symbol . between the source and target sentences in absolute terms.
92. Number of mismatches of the symbol . between the source and target sentences normalized by sentence length.
93. Number of mismatches of the symbol ; between the source and target sentences in absolute terms.

94. Number of mismatches of the symbol ; between the source and target sentences normalized by sentence length.
95. Number of mismatches of the symbol : between the source and target sentences in absolute terms.
96. Number of mismatches of the symbol : between the source and target sentences normalized by sentence length.
97. Number of mismatches of the symbol = between the source and target sentences in absolute terms.
98. Number of mismatches of the symbol = between the source and target sentences normalized by sentence length.
99. Number of mismatches of the symbol ¡ between the source and target sentences in absolute terms.
100. Number of mismatches of the symbol ¡ between the source and target sentences normalized by sentence length.
101. Number of mismatches of the symbol ? between the source and target sentences in absolute terms.
102. Number of mismatches of the symbol ? between the source and target sentences normalized by sentence length.
103. Number of mismatches of the symbol ¿ between the source and target sentences in absolute terms.
104. Number of mismatches of the symbol ¿ between the source and target sentences normalized by sentence length.
105. Number of mismatches of the symbol @ between the source and target sentences in absolute terms.
106. Number of mismatches of the symbol @ between the source and target sentences normalized by sentence length.
107. Number of mismatches of the symbol [between the source and target sentences in absolute terms.
108. Number of mismatches of the symbol [between the source and target sentences normalized by sentence length.
109. Number of mismatches of the symbol] between the source and target sentences in absolute terms.
110. Number of mismatches of the symbol] between the source and target sentences normalized by sentence length.
111. Number of mismatches of the symbol \between the source and target sentences in absolute terms.
112. Number of mismatches of the symbol \between the source and target sentences normalized by sentence length.
113. Number of mismatches of the symbol _ between the source and target sentences in absolute terms.
114. Number of mismatches of the symbol _ between the source and target sentences normalized by sentence length.
115. Number of mismatches of the symbol ^ between the source and target sentences in absolute terms.
116. Number of mismatches of the symbol ^ between the source and target sentences normalized by sentence length.
117. Number of mismatches of the symbol ` between the source and target sentences in absolute terms.
118. Number of mismatches of the symbol ` between the source and target sentences normalized by sentence length.
119. Number of mismatches of the symbol { between the source and target sentences in absolute terms.

120. Number of mismatches of the symbol { between the source and target sentences normalized by sentence length.
121. Number of mismatches of the symbol } between the source and target sentences in absolute terms.
122. Number of mismatches of the symbol } between the source and target sentences normalized by sentence length.
123. Number of mismatches of the symbol | between the source and target sentences in absolute terms.
124. Number of mismatches of the symbol | between the source and target sentences normalized by sentence length.
125. Number of mismatches of the symbol ˜ between the source and target sentences in absolute terms.
126. Number of mismatches of the symbol ˜ between the source and target sentences normalized by sentence length.
127. Number of mismatches of all the punctuation symbols between the source and target sentences in absolute terms.
128. Number of mismatches of all the punctuation symbols between the source and target sentences normalized by sentence length.
129. Average words of frequency according to Levenstein criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
130. Average words of frequency according to the target position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
131. Average words of frequency according to any position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
132. Average words of frequency according to the average position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
133. Average words of frequency considering ranking according to Levenstein criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
134. Average words of frequency considering ranking according to the target position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
135. Average words of frequency considering ranking according to any position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
136. Average words of frequency considering ranking according to the average position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
137. Average words of frequency considering probabilities according to Levenstein criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
138. Average words of frequency considering probabilities according to the target position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
139. Average words of frequency considering probabilities according to any position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).

140. Average words of frequency considering probabilities according to the average position criteria of the N-best (Sanchis, 2004) translated with a model trained with Europarl version 2 (Koehn, 2006).
141. Geometric average of IBM1 probability (Ueffing et al., 2003) trained with the corpora Europarl version 5 (Koehn, 2010) and News Commentary (Callison-Burch et al., 2010) of the words in the target sentence.
142. Number of verbs in target sentence.
143. Number of nouns in target sentence.
144. TER between POS-tagged (extracted with the software Freeling (Padró, 2011)) source and target.
145. Average of words probability that the word is correct given by Naïve Bayes (Sanchis, 2004) classifier of the source sentence belonging to the first quartile using the corpus Europarl version 2 (Koehn, 2006) of the source language. The optimization of the parameters was done with a separate development set Europarl version 2.
146. Average of words probability that the word is correct given by Naïve Bayes (Sanchis, 2004) classifier of the source sentence belonging to the second quartile using the corpus Europarl version 2 (Koehn, 2006) of the source language. The optimization of the parameters was done with a separate development set Europarl version 2.
147. Average of words probability that the word is correct given by Naïve Bayes (Sanchis, 2004) classifier of the source sentence belonging to the third quartiles using the corpus Europarl version 2 (Koehn, 2006) of the source language. The optimization of the parameters was done with a separate development set Europarl version 2.
148. Average of words probability that the word is correct given by Naïve Bayes (Sanchis, 2004) classifier of the source sentence belonging to the fourth quartiles using the corpus Europarl version 2 (Koehn, 2006) of the source language. The optimization of the parameters was done with a separate development set Europarl version 2.
149. Number of adjectives in target sentence.
150. Number of verbs in source sentence.
151. Number of verbs in source sentence.
152. Number of names in source sentence.
153. Ratio of number of verbs.
154. Ratio of number of names.
155. Ratio of number of adjectives.
156. TER computed with source sentence as reference and the machine translation of target sentence to the source language as hypothesis. The translation is done using a model trained with the corpus Europarl version 2 (Koehn, 2006) and the software Moses (Koehn et al., 2007).

Chapter 6

Appendix II: Use of the method

In this appendix we present some examples of use of the method proposed. The examples are from the test samples used in our experiments:

- Sentence that would be well predicted:
  - Source sentence: *i would like to end with a slogan for the future , when disabled people will not just be ‘ getting on board ’ but also ‘ staying on board ’ .*
  - Reference sentence: *quiero acabar con un lema para el futuro , cuando las personas discapacitadas no solo vayan a ≪ subir a bordo ≫ , sino también a ≪ permanecer a bordo ≫ .*
  - Translated sentence: *quisiera terminar con un lema para el futuro , cuando las personas discapacitadas no será sólo ‘ llegar a bordo , sino también permanecer a bordo ’ ‘ ’ .*
  - Actual TER value = 41%
  - Predicted TER value = 42 %
- Sentence that could be kept (depends on the threshold used) without post-edition because is predicted with a low TER value:
  - Source sentence: *madam president , unfortunately , i could not resolve the problems of the structural funds with eur 10 .*
  - Reference sentence: *señora presidenta , por desgracia yo no podría resolver los problemas de los fondos estructurales con 10 euros .*
  - Translated sentence: *señora presidenta , desgraciadamente , no podría resolver los problemas de los fondos estructurales con 10 euros .*
  - Actual TER value = 16%
  - Predicted TER value = 20%

- Sentence that could be selected (depends on the threshold used) to post-edition or elimination (depends on the criteria):
  - Source sentence: *and what was the background to that ? the stuff had been exported to iraq – for civilian purposes – from europe .*
  - Reference sentence: *hay que decir que esta guerra preventiva de iraq ha convencido , contra toda lógica , a los dictadores de que la posesión de armas nucleares es una especie de salvoconducto .*
  - Translated sentence: *y lo que era el trasfondo de esos materiales ? que habían sido exportadas a iraq – para fines civiles de europa – .*
  - Actual TER value = 84%
  - Predicted TER value = 82%

Results for the 21 first sentences of the data set. It is shown in order: the source text, translation and error of each sentence. The sentences selected as bad translations are in bold. We have used a TER threshold of 50%.

Src: the next item is the report (a6-0216 / 2006) by mr ransdorf on behalf of the committee on industry , research and energy , on nanoscience and nanotechnologies : an action plan for europe 2005-2009 [2006 / 2004 (ini)] .

Trg: de conformidad con el orden del día es el informe (a6-0216 / 2006) formulada por el señor ransdorf en nombre de la comisión de industria , investigación y energía , sobre la nanociencia y nanotecnologías : un plan de acción para europa 2005-2009 (2006 / 2004 (ini)] .

Err: 30.2%

Src: as regards patents , however , the united states ' worldwide share is 42 % , whereas the eu stands at 36 % .

Trg: en lo que respecta a las patentes , sin embargo , los estados unidos comparten el 42 % , es ' mundial mientras que la ue se sitúa en el 36 % .

Err: 45.3%

Src: us federal expenditure is approximately equivalent to that of the whole of the eu in the area of nanotechnologies and nanosciences , and the individual member states have unequal spending levels .

Trg: nosotros federal gastos es aproximadamente equivalente a la del conjunto de la ue en el ámbito de la y nanotecnologías , y cada uno de los estados miembros han desigual los niveles de gasto .

Err: 41.2%

Src: in this regard , i should like to quote two great scholars . the first is johann wolfgang von goethe .

Trg: a este respecto , quisiera citar dos grandes maestros . la primera es johann wolfgang von goethe .

Err: 44.8%

Src: let me say that it is vitally important to emphasise the social aspect of nanotechnologies .

Trg: **permítanme decir que es sumamente importante hacer hincapié en el aspecto social de nanotecnologías .**

Err: 50.6%

Src: they are comparable in scope with microelectronics in the 1960s , 70s and 80s .

Trg: no son comparables en el ámbito de la microelectrónica en los años sesenta y setenta y ochenta .

Err: 36.9%

Src: ladies and gentlemen , these are my introductory remarks and i look forward to the debate .

Trg: señoras y señores , estas son mis observaciones preliminares y espero con interés el debate .

Err: 37.0%

Src: mr president , i am here today to talk about the big issue of small technologies .

Trg: señor presidente , hoy estoy aquí para hablar de la gran cuestión de pequeñas tecnologías .

Err: 39.4%

Src: europe is in a leading position in the world today , partly thanks to the commission ' s framework programme .

Trg: europa está en una posición de liderazgo en el mundo de hoy , en parte gracias a la comisión ' del programa marco .

Err: 37.5%

Src: european industry should now reap the benefits of that knowledge through innovative products and processes .

Trg: la industria europea debería ahora cosechar los beneficios de ese conocimiento a través de los productos y procesos innovadores .

Err: 45.2%

Src: that is a key area , because , as well as the benefits , we must also recognise the potential risks .

Trg: este es un ámbito clave , porque , además de los beneficios , también debemos reconocer los riesgos potenciales .

Err: 43.1%

Src: special projects and publicity in many languages will provide information and communication .

Trg: proyectos especiales y publicidad en muchos idiomas proporcionará información y la comunicación .

Err: 37.7%

Src: we are assessing how adequate and appropriate that legislation is to deal with the increasing use of nanotechnologies .

Trg: **estamos evaluando cómo suficiente y adecuado que la legislación es abordar la creciente utilización de nanotecnologías .**

Err: 57.3%

Src: we also need to consider potential regulatory issues .

Trg: también necesitamos considerar potencial cuestiones reglamentarias .

Err: 49.8%

Src: it is equally important that it stresses the importance of creating the right climate for innovation in europe as well as emphasising the importance of ‘ speaking with one voice ’ internationally in this highly promising research area .

Trg: **es igualmente importante que hace hincapié en la importancia de crear el clima adecuado para la innovación en europa , así como de hacer hincapié en la importancia de ‘ hablar con una sola voz ’ internacionalmente en esta muy prometedor de investigación .**

Err: 50.7%

Src: firstly , nanoscience and nanotechnology is permeated with ethical issues .

Trg: en primer lugar , nanociencia y nanotecnología está impregnado con las cuestiones éticas .

Err: 44.0%

Src: with strong growth projected in the field of nanosciences and nanotechnology , it is important that the eu accept the commission proposal to adopt new approaches to this industry , from education to r [amp] d. such actions will contribute to heightened competitiveness and development in our member states .

Trg: **con un fuerte crecimiento previsto en el ámbito de la nanotecnología , y es importante que la ue aceptar la propuesta de la comisión de adoptar nuevos enfoques para este sector , de la educación al i+d . estas acciones contribuyen a agravar la competitividad y el desarrollo en nuestros estados miembros .**

Err: 50.5%

Src: . mr president , first of all i would like to congratulate mr ransdorf , my colleague and vice-president of the committee on industry , research and energy , for his excellent report .

Trg: . señor presidente , en primer lugar quiero felicitar al señor ransdorf , mi colega y vicepresidente de la comisión de industria , investigación y energía , por su excelente informe .

Err: 29.2%

Src: having said that , i wish to enter a note of regret at the rather negative and fearful approach characterised in the verts / ale group ' s amendments .

Trg: **dicho esto , quiero entrar en una nota de lamentar la bastante negativo y temible enfoque caracterizado de los verdes / ale grupo ' las enmiendas .**

Err: 71.6%

Src: i would urge caution , therefore , on the requirements for labelling in advance of scientific evidence and on applying the precautionary principle .

Trg: yo recomendaría precaución , por tanto , sobre los requisitos de etiquetado de antemano de pruebas científicas y sobre la aplicación del principio de precaución .

Err: 38.1%

Src: if we always applied this principle , then innovation , invention and inquiry would all go out the window and we would make no progress at all .

Trg: **si no siempre se aplica este principio , entonces invención y la innovación , la investigación no todos salir la ventana y no avanzaremos en absoluto .**

Err: 61.3%

Bibliography

Abdi, H. (2010). Partial least squares regression and projection on latent structure regression (pls regression). *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):97–106.

Ai, W. I. and Langley, P. (1992). Induction of one-level decision trees. In *Proceedings of the Ninth International Conference on Machine Learning*, pages 233–240. Morgan Kaufmann.

Blatz, J., Fitzgerald, E., Foster, G., Gandrabur, S., Goutte, C., Kulesza, A., Sanchis, A., and Ueffing, N. (2004). Confidence estimation for machine translation. In *in M. Rolling (ed.), mental imagery*. Yale University Press.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19(2):263–311.

Callison-Burch, C., Fordyce, C., Koehn, P., Monz, C., and Schroeder, J. (2008). Further meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 70–106, Stroudsburg, PA, USA. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., Peterson, K., Przybocki, M., and Zaidan, O. (2010). Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, Uppsala, Sweden. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., Monz, C., Post, M., Soricut, R., and Specia, L. (2012). Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada. Association for Computational Linguistics.

chung Chang, C. and Lin, C.-J. (2001). Libsvm: a library for support vector machines.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cohen, J., Cohen, P., West, S. G., and Aiken, L. S. (2002). *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Routledge Academic, third edition.

Cox, S. (2004). Confidence measures in speech recognition. volume 3361. Springer Bengio, Samy and Bourlard, HervéEditors.

Cramer, R. (1993). Partial least squares (pls): Its strengths and limitations. *Perspectives in Drug Discovery and Design*, 1:269–278.

Cybenko, G. (1992). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 5:455–455.

Dayal, B. S. and MacGregor, J. F. (1997). Improved pls algorithms. *Journal of Chemometrics*, 11(1):73–85.

de Jong, S. (1993). Simpls: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18(3):251 – 263.

De Jong, S. and Ter Braak, C. J. F. (1994). Comments on the pls kernel algorithm. *Journal of Chemometrics*, 8(2):169–174.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, HLT '02, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Drucker, H., Burges, C. J., Kaufman, L., C, C. J., Kaufman, B. L., Smola, A., and Vapnik, V. (1996). Support vector regression machines.

Duch Guillot, J. (2007). European parliament - never lost in translation. Press service.

Frenich, A. G., Jouan-Rimbaud, D., Massart, D. L., Kuttatharmmakul, S., Galera, M. M., and Vidal, J. L. M. (1995). Wavelength selection method for multicomponent spectrophotometric determinations using partial least squares. *Analyst*, 120:2787–2792.

Gamon, M., Aue, A., and Smets, M. (2005). Sentence-level mt evaluation without reference translations: Beyond language modeling. In *European Association for Machine Translation (EAMT)*.

Gandrabur, S. and Foster, G. (2003). Confidence estimation for translation prediction. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 95–102, Stroudsburg, PA, USA. Association for Computational Linguistics.

Google (2001). Stop-words google code.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

He, Y., Ma, Y., van Genabith, J., and Way, A. (2010). Bridging SMT and TM with translation recommendation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 622–630, Uppsala, Sweden. Association for Computational Linguistics.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. In *Machine Learning*, pages 63–91.

Hutchins, W. J. (2003). Machine translation. In *Encyclopedia of Computer Science*, pages 1059–1066. John Wiley and Sons Ltd., Chichester, UK.

J. A. Nelder, R. M. (1965). A simplex method for function minimization.

Johnson, H., Sadat, F., Foster, G., Kuhn, R., Simard, M., Joanis, E., and Larkin, S. (2006). Portage: with smoothed phrase tables and segment choice models. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, pages 134–137, Stroudsburg, PA, USA. Association for Computational Linguistics.

Juan, A. (2011). Feature extraction, lecture slides.

Kääriäinen, M. (2009). Sinuhe: statistical machine translation using a globally trained conditional exponential family translation model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 1027–1036, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kenney, J. F. and Keeping, E. S. (1962). *Linear Regression and Correlation.* Princeton, third edition.

Koehn, P. (2006). Europarl: A parallel corpus for statistical machine translation, version 2.

Koehn, P. (2010). Europarl: A parallel corpus for statistical machine translation, version 2.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lindgren, F., Geladi, P., and Wold, S. (1993). The kernel algorithm for pls. *Journal of Chemometrics*, 7(1):45–59.

MATLAB (2012). *version 7.14.0.739 (R2012a).* The MathWorks Inc., Natick, Massachusetts.

Moody, J. and Darken, C. J. (1989). Fast learning in networks of locally-tuned processing units. *Neural Comput.*, 1(2):281–294.

Nagao, M. (1984). A framework of a mechanical translation between japanese and english by analogy principle. In *Proc. of the international NATO symposium on Artificial and human intelligence*, pages 173–180, New York, NY, USA. Elsevier North-Holland, Inc.

Och, F. J. and Ney, H. (2000). Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 440–447, Stroudsburg, PA, USA. Association for Computational Linguistics.

Oliver, J. J. and Hand, D. (1994). Averaging over decision stumps. In *Proceedings of the European conference on machine learning on Machine Learning*, ECML-94, pages 231–241, Secaucus, NJ, USA. Springer-Verlag New York, Inc.

Padró, L. (2011). Analizadores multilingües en freeling. *Linguamatica*, 3(2):13–20.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572.

Quirk, C. B. (2004). Training a sentence-level machine translation confidence measure. In *Proceedings of LREC*.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Reyzin, L. and Schapire, R. E. (2006). How boosting the margin can also boost classifier complexity. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 753–760, New York, NY, USA. ACM.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan.

Rosipal, R., Be, P. P., Trejo, L. J., Cristianini, N., Shawe-taylor, J., and Williamson, B. (2001). Kernel partial least squares regression in reproducing kernel hilbert space.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. *Mit Press Computational Models Of Cognition And Perception Series*, pages 318–362.

Rännar, S., Lindgren, F., Geladi, P., and Wold, S. (1994). A pls kernel algorithm for data sets with many variables and fewer objects. part 1: Theory and algorithm. *Journal of Chemometrics*, 8(2):111–125.

Sanchis, A. (2004). *Estimación y aplicación de medidas de confianza en reconocimiento automático del habla*. PhD thesis, Departamento de Sistemas Informáticos y Computación.

Saunders, C. (2008). Application of markov approaches to statistical machine translation.

Shaw, J. A. (2003). *Multivariate statistics for the Environmental Sciences*. Hodder-Arnold.

Simard, M., Cancedda, N., Cavestro, B., Dymetman, M., Gaussier, E., Goutte, C., Yamada, K., Langlais, P., and Mauser, A. (2005). Translating with non-contiguous phrases. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05,

pages 755–762, Stroudsburg, PA, USA. Association for Computational Linguistics.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

Specia, L., Cancedda, N., and Dymetman, M. (2010a). A dataset for assessing machine translation evaluation metrics. In *LREC'10*, pages –1–1.

Specia, L. and Farzindar, A. (2010). Estimating machine translation post-editing effort with hter. 2010. *AMTA2010 Workshop Bringing MT to the User MT Research and the Translation Industry*, page 33–41.

Specia, L., Raj, D., and Turchi, M. (2010b). Machine translation evaluation versus quality estimation. *Machine Translation*, 24(1):39–50.

Specia, L., Turchi, M., Cancedda, N., Dymetman, M., and Cristianini, N. (2009a). Estimating the sentence-level quality of machine translation systems. *Proceedings of the 13th Annual Conference of the EAMT*, pages 28–35.

Specia, L., Turchi, M., Shawe-Taylor, Z., and Saunders, C. (2009b). Improving the confidence of machine translation quality estimates. *Proceedings of MT Summit XII*.

Tillmann, C., Vogel, S., Ney, H., Zubiaga, A., and Sawaf, H. (1997). Accelerated dp based search for statistical translation. In *In European Conf. on Speech Communication and Technology*, pages 2667–2670.

Ueffing, N., Macherey, K., and Ney, H. (2003). Confidence measures for statistical machine translation. In *Proc. MT Summit IX*, pages 394–401. Springer-Verlag.

Ueffing, N. and Ney, H. (2007). Word-level confidence estimation for machine translation. *Comput. Linguist.*, 33(1):9–40.

Vapnik, V. and Chervonenkis, A. (1964). A note on one class of perceptrons. *Automation and Remote Control*, 25.

Vapnik, V. and Lerner, A. (1963). *Automation and Remote Control*, 24(6):774–780.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Wessel, F., Schlüter, R., Macherey, K., and Ney, H. (2001). Confidence measures for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 9:288–298.