Machine Translation manuscript No.
(will be inserted by the editor)

Dimensionality Reduction Methods for Machine Translation Quality Estimation

Jesús González-Rubio · J. Ramón Navarro-Cerdán · Francisco Casacuberta

Received: date / Accepted: date

Abstract Quality estimation for machine translation is usually addressed as a regression problem where a learning model is used to predict a quality score from a (usually highly-redundant) set of features that represent the translation. This redundancy hinders model learning, and thus penalizes the performance of quality estimation systems. We propose different dimensionality reduction methods based on partial least squares regression to overcome this problem, and compare them against several reduction methods previously used in the quality estimation literature. Moreover, we study how the use of such methods influence the performance of different learning models. Experiments carried out on the English-Spanish WMT12 quality estimation task showed that it is possible to improve prediction accuracy while significantly reducing the size of the feature sets.

Keywords Machine translation · Quality estimation · Dimensionality reduction · Partial least squares regression

1 Introduction

Despite an intensive research in the last fifty years, machine translation (MT) systems are still error-prone. Thus, a desirable feature to improve the broader and more effective deployment of (nowadays) imperfect MT technology is the capability of predicting the reliability, namely the quality, of the generated translations. Historically, translation quality assessment has been done manually by human experts. These experts need to read the automatic translation and the source text to be able to judge whether the translation is good or not which, obviously, is a very time consuming task. Therefore, automatical translation quality assessment is a crucial problem, either to present the translations in such way

Jesús González-Rubio (✉) · Francisco Casacuberta
Universitat Politècnica de València, Valencia, Spain
E-mail: jegonzalez@dsic.upv.es · fcn@dsic.upv.es

J. Ramón Navarro-Cerdán
Instituto Tecnológico de Informática, Valencia, Spain
E-mail: jonacer@iti.upv.es

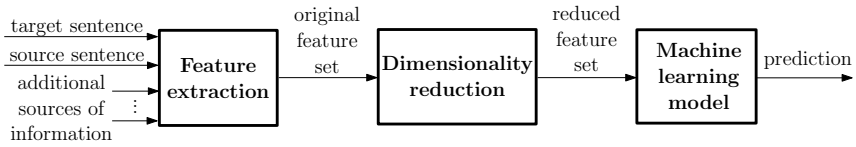

Fig. 1 Dataflow of the proposed two-step quality estimation approach.

as to make end-users aware of the quality [Specia et al., 2009b], or to filter out the translations according to the requirements of a given task and level of expertise of the professional translator, e.g. to avoid professional translators spending time reading / post-editing certain translations [Blatz et al., 2004, Quirk, 2004, Specia et al., 2009a, González-Rubio et al., 2010]. This task, referred to as confidence or quality estimation (QE), is concerned about predicting MT output quality without any information about the expected output. Quality information may be provided for each word [Gandrabur and Foster, 2003, Ueffing and Ney, 2007, Sanchis et al., 2007], sentence [Blatz et al., 2004, Quirk, 2004, Gamon et al., 2005, Specia et al., 2009b] or document [Soricut and Echihabi, 2010]. This article focuses on sentence-level QE.

We distinguish the task of QE from that of MT evaluation by the need, in the latter, of reference translations. The goal of MT evaluation is to compare an automatic translation to reference translation(s) and provide a quality score which reflects how close the two translations are. In QE, the task consist in estimating the quality of the translation given only information about the input and output texts and the translation process.

Sentence-level QE is typically addressed as a regression problem [Quirk, 2004, Blatz et al., 2004, Specia et al., 2009b]. Given a translation generated by an MT system (and potentially other additional sources of information) a set of features is extracted. Then, a model trained using a particular machine learning algorithm is employed to compute a quality score from these features. Most QE works consider a fixed set of features and study the performance of different learning algorithms on those features. However, feature sets tend to be highly redundant, i.e. there is high multicollinearity between the features, and some of the features may even be irrelevant to predict the quality score. Moreover, a set of translations labeled with their "true" quality score is required to train the learning model. Since this labeling process is usually done manually, training sets rarely contain enough labeled samples to accurately train the model. By removing irrelevant and redundant features from the data, dimensionality reduction (DR) methods potentially improve the performance of learning models by alleviating the effect of the "curse" of dimensionality, enhancing generalization capability of the model, and speeding up the learning process. Additionally, DR may also help the researchers to acquire better understanding about their data by telling them which are the important features and how they are related with each other. Despite these potential improvements, works on QE usually put little attention on DR. For example, only six out of the eleven participants to the QE task of the 2012 workshop on statistical machine translation [Callison-Burch et al., 2012] applied DR, and even those participants that used DR only implemented simple feature selection methods.

In this article, we propose two novel DR methods based on partial least squares regression (PLSR) [Wold, 1966]. We consider both a DR method that selects a subset of the original features, namely a feature selection method, and a method

that projects the original data into a space of fewer dimensions, a feature extraction method. Despite being usually more complex, feature extraction methods have a potential advantage over feature selection: they can generate new features that summarize the "information" contained in all original features. In contrast, the information contained in the features discarded by a feature selection method is inevitably lost. The proposed methods are compared to other DR methods previously used in the literature: methods based on statistical multivariate analysis such as PCA [Pearson, 1901] and PLSR regressors selection [Specia et al., 2009b], and heuristic wrapper selection methods [Kohavi and John, 1997]. Moreover, we study how these DR methods affect the performance of different learning models.

The performance of each DR method was evaluated by the prediction accuracy of the models trained in the corresponding reduced feature sets. Figure 1 shows a scheme of the process followed to obtain a quality score from a given translation. First, from the translation, and additional information sources, we compute a (possibly high-dimensional and highly-redundant) set of features that represent the translation. Then, we apply a DR method to obtain a reduced feature set that still contains the relevant information present in the original feature set. Finally, we use a trained learning model to predict the quality score of the translation from this reduced feature set. To assure an accurate comparison between the different DR methods, identical pipelines were used to train the models. By providing a detailed description and a systematic evaluation of these DR methods, we give the reader various criteria for deciding which method to use for a given task.

It should be noted that despite being tested in a QE task, the proposed two-step training and DR methods do not make particular assumptions about the features or the learning model. Thus, they constitute a general methodology that can be applied to a great variety of supervised learning tasks.

The rest of the article is organized as follows. In Section 2, we formalize the regression approach to QE. In Section 3, we state the DR problem and present the different DR methods under study. Section 4 is devoted to describe our experimental setting which include a description of the features extracted for each translation (Section 4.2), and the different learning models used in the experimentation (Section 4.3). In Section 5, we present and discuss the empirical results obtained in the experimentation, and, finally, we conclude with a summary in Section 6.

2 Quality Estimation

We formalize QE as a regression problem where we model the relationship between a dependent variable y (the quality score), and a vector of m explanatory variables $\mathbf{x}^T = (x_1, \ldots, x_m)$ (the features that represent the translation). Given a data set with n samples $\{y_i, \mathbf{x}_i\}_{i=1}^n$, our goal is to build a predictive model $\mathbb{M}_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}$ with free parameters $\boldsymbol{\theta}$. The data set is usually represented in matrix form where $\mathbf{y}$ is a vector that contains the quality scores, and $\mathbf{X}$ is a matrix where each row is the feature vector of one training sample:

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \qquad \mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{pmatrix} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{pmatrix}$$

To carry out the regression, the form of the model $\mathbb{M}_{\boldsymbol{\theta}}$ must be specified. Since we do not know how $\mathbf{y}$ and $\mathbf{X}$ actually relate, we use different flexible models (see Section 4.3) whose free parameters $\boldsymbol{\theta}$ can be estimated to fit the data. Typically, these models include a regularization term [Tibshirani, 1996] that facilitates the learning process in the presence of noisy and collinear data. One of the goals of the experimentation will be to study if regularized models can also benefit from an explicit DR of the feature space.

3 Dimensionality Reduction

3.1 Motivation

The proposed QE formalization assumes that translation quality can be described by a number of independent variables. Since these underlying variables are unknown, in practice, we instead extract a (possibly larger) set of features that aim at describing the prediction information contained in the underlying variables. This approach implies to consider translation quality as governed by more variables than it really is, which results in several learning problems due to the addition of irrelevant features, or the multicollinearity between them. However, provided the influence of this "extra" features is not too strong as to completely mask the original structure, we should be able to "filter" them out and recover the original variables or an equivalent set of them. DR methods aim at somehow strip off this redundant information, producing a more economic representation of the data.

DR can also be seen as a method to overcome the so-called "curse" of dimensionality. This term, coined in [Bellman, 1961], refers to the fact that, in the absence of simplifying assumptions, the sample size needed to estimate a function of several variables to a given degree of accuracy grows exponentially with the number of variables. Responsible for the "curse" of dimensionality is the fact that high-dimensional spaces are inherently sparse which is known as the empty space phenomenon [Scott and Thompson, 1983]. This is a difficult problem in model estimation, as regions of relatively very low density can contain a considerable part of the distribution, whereas regions of apparently high density can be completely devoid of observations in a sample of moderate size. DR technology address these problems, by reducing the input dimension of the function to be estimated.

3.2 Problem Statement and Approaches

The DR problem can be stated as follows: given a regression problem $P_1 : \mathbb{R}^m \to \mathbb{R}$, we want to obtain an equivalent problem $P_2 : \mathbb{R}^r \to \mathbb{R}$ where $r << m$. In other words, we want to obtain a low-dimensional, compact representation of the input data that still retains the information required to perform an accurate prediction. Formally, DR is defined by a function Δ that transforms an m-dimensional space into an r-dimensional space:

$$\Delta : \mathbb{R}^m \to \mathbb{R}^r \tag{1}$$

The determination of the dimension r of this compact representation is central to the DR problem, because knowing it would eliminate the possibility of over- or under-fitting. All the methods studied in this article take this intrinsic dimension

as a parameter to be given by the user; a trial-and-error process is thus necessary to obtain a satisfactory value for it.

Next, we describe the different DR methods tested in the experimentation. For a more clear presentation, we distinguish between heuristic methods and methods derived from statistical multivariate analysis.

3.3 Heuristic Feature Selection Methods

We consider heuristic wrapper [Kohavi and John, 1997] methods to address the problem of feature selection. In the wrapper methodology, the learning model is considered a perfect black box. In its most general formulation, this methodology consists in using the prediction accuracy of a given learning model to assess the relative usefulness of subsets of features. In practice, the different wrapper methods are defined by the search strategy implemented to explore the space of possible subsets. An exhaustive search can conceivably be performed if the number of features is not too large. For example, all the subsets for 24 features (2^{24}) were explored in [Soricut et al., 2012]. However, the problem is known to be NP-hard [Amaldi and Kann, 1998] and the search quickly becomes computationally intractable.

In out experimentation, we tested two search strategies that define two different heuristic feature selection methods: ranking of feature selection, and greedy forward selection. Since the computational complexity of these simple methods depends on the complexity of the chosen learning model, we use symbol $\zeta(n, m)$ to denote the time complexity to train the actual learning model with n samples of m-dimensional feature vectors.

3.3.1 Rank of Feature

Rank of feature selection (RFS) generates subsets of features by selecting the top-scoring features according to the prediction accuracy of a QE system trained solely with that feature [González-Rubio et al., 2012]. RFS is typically used as a baseline selection mechanism because of its simplicity, scalability and (somewhat) good empirical success [Guyon and Elisseeff, 2003]. The computational complexity of RFS to generate the first reduced feature set is given by $O(m \cdot \zeta(n, 1))$; once the scores for the features are computed, we can generate reduced groups of different sizes with no further calculations. For example, the complexity of RFS if we use a linear model[1] is in $O(m \cdot n)$ given that $\zeta(n, 1)$ is proportional to n.

Since RFS selects the features according to their individual prediction accuracy, we expect to obtain subsets of features that also provide good prediction accuracy. However, RFS does not take into account the correlations that may exist between the different features, thus, these subsets will probably contain a large number of redundant features.

3.3.2 Greedy Forward

Greedy forward selection [Kohavi and John, 1997, Avramidis, 2012] (GFS) incrementally creates subsets of features by selecting at each iteration the feature that,

[1] This particular setup can be considered as a lower bound of the complexity of RFS.

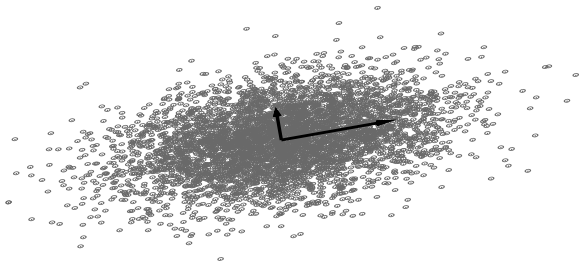

Fig. 2 PCA example for a 2-dimensional gaussian distribution. The vectors represent the two principal components of the data.

when added to the current set, yields the learned model that performs best. In contrast to RFS, GFS recomputes the importance of each feature at each step having into account the current subset of features. Thus, the computational complexity of GFS to compute a reduced set of size r is $O(\sum_{i=1}^{r}\sum_{j=1}^{m-i+1}\zeta(n,i))$ that is upper bounded by $O(r\cdot m\cdot\zeta(n,r))$. For example, if we use a linear model the temporal complexity of GFS is in $O(r^2\cdot m\cdot n)$ given that $\zeta(n,r)\propto n\cdot r$.

Since GFS selects at each step the feature that improves most the QE model performance, we expect to obtain subsets with lower redundancy in comparison to RFS. However, it requires to re-compute the contribution of each feature to the QE model at each step, $O(\zeta(n,r))$, which penalizes GFS complexity.

3.4 DR Methods Based on Statistical Multivariate Analysis

Statistical multivariate analysis is a generic term for any statistical technique concerned with analyzing data in high dimensions [Anderson, 1958]. In particular, we focus on statistical techniques to partition the variability of the data into components attributable to different sources of variation. In this work, we consider two of these techniques: principal component analysis (PCA), and partial least squares regression (PLSR). Given a number of dimensions r, both PCA and PLSR compute a transformation of the original data space into an orthogonal r-dimensional space. However, they differ in the criteria followed to compute this transformation.

The main advantage of these methods stems in the orthogonality of the output space; which means that the transformed features will be linearly independent by construction. Therefore, using these transformations we obtain reduced feature sets with almost no redundant information. Moreover, statistical multivariate methods are mathematically well-founded and independent of the choosen learning model. However, these methods also have an obvious drawback, i.e. new features are computed as a linear combination of all original features which makes it often difficult to interpret them.

3.4.1 Principal Component Analysis

Principal component analysis (PCA) [Pearson, 1901] defines a transformation of the original data into a new space of features, known as principal components. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the

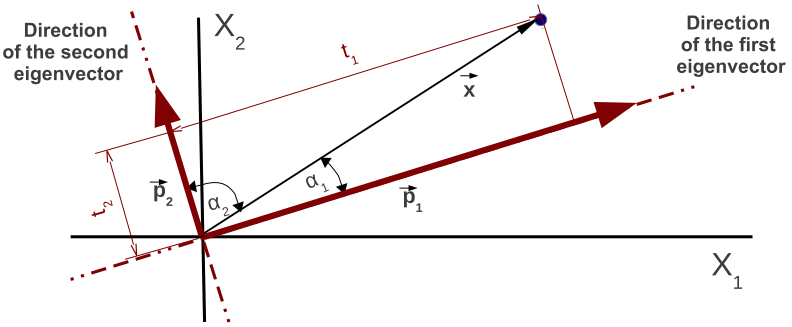

Fig. 3 Example of the principal component values (t_1, t_2) for a data point $\mathbf{x}$ in Figure 2. Values t_1, t_2 are computed by projecting $\mathbf{x}$ over the corresponding eigenvectors $(\mathbf{p}_1, \mathbf{p}_2)$.

data as possible), and each succeeding component in turn has the highest variance possible under the constraint of being uncorrelated with the preceding components. Therefore, each of these principal components represent one of the individual latent factors that actually govern the variability of the data, as exemplified in Figure 2.

Given a matrix $\mathbf{X}$ whose rows represent the n samples and each column represents one of the m features, PCA is formalized by the following decomposition:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T \tag{2}$$

where $\mathbf{P}$ is the space transformation matrix that contains the eigenvectors of the covariance matrix $\mathbf{X}^T\mathbf{X}$, and the rows of $\mathbf{T}$ represent the principal components of each training sample. The nonlinear iterative partial least squares (NIPALS) algorithm [Wold, 1966] is commonly used to obtain the eigenvectors.

Given that the eigenvectors in $\mathbf{P}$ are unitary and orthogonal ($\mathbf{P}^T\mathbf{P} = \mathbf{I}$), we can multiply both sides of Equation (2) by $\mathbf{P}$ to obtain the principal components $\mathbf{T}$ of the data:

$$\mathbf{X}\mathbf{P} = \mathbf{T} \tag{3}$$

Figure 3 shows a graphical example of the computation of two principal components $\mathbf{t} = (t_1, t_2)$ for a single data point $\mathbf{x}$. Each principal component t_k is computed by projecting $\mathbf{x}$ over the corresponding unitary eigenvector $\mathbf{p}_k$. Specifically, $t_k = \mathbf{x} \cdot \mathbf{p}_k = ||\mathbf{x}|| \cdot ||\mathbf{p}_k|| \cdot \cos(\alpha_k) = ||\mathbf{x}|| \cdot \cos(\alpha_k)$, where α_k is the angle between $\mathbf{x}$ and $\mathbf{p}_k$.

PCA Projection

The principal components are linearly independent, and each of them accounts for the maximum variability in $\mathbf{X}$ not explained by previous components, thus we follow [González-Rubio et al., 2012] and select the first r components to create the reduced feature sets. Since each of these components is a linear combination of the original features, this is a feature extraction method. In the experiments, we use PCA-P to denote this DR approach.

The complexity of PCA-P to compute a reduced set of size r is given by the complexity of the NIPALS algorithm: $O(r \cdot m \cdot n)$. Note that in contrast to the previously presented heuristic methods, the cost of PCA-P does not depend on the complexity of the chosen learning model.

3.4.2 Partial Least Squares Regression

PCA generates sets of orthogonal features where each feature explains the variability of the data $\mathbf{X}$ in one principal direction. However, this transformation ignores the scores $\mathbf{y}$ to be predicted. Thus, although the features generated by PCA-P contain almost no redundancy, they do not necessarily have to be the best set of features to perform the prediction. Partial least squares regression (PLSR) [Wold, 1966] is an alternative to PCA that takes into account $\mathbf{y}$ when computing the transformation of $\mathbf{X}$. Specifically, PLSR computes a ordered set of latent variables such that each of them account for the maximum co-variability between $\mathbf{X}$ and $\mathbf{y}$ under the constraint of being uncorrelated with previous latent variables. Formally, PLSR builds the following model where $\mathbf{b}$ is a vector of regressor coefficients, and $\mathbf{f}$ is a vector of zero-centered Gaussian errors:

$$\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{f} \tag{4}$$

Even though this is a linear regression model the estimation of the regression coefficients $\mathbf{b}$ for PLSR is different from the conventional least squares regression, see Section 4.3.1. The intuitive idea of PLSR is to describe $\mathbf{y}$ as well as possible, hence to make $||\mathbf{f}||$ as small as possible, and, at the same time, take advantage of the relation between $\mathbf{X}$ and $\mathbf{y}$. To do that, PLSR defines two independent PCA-like transformations $\mathbf{P}$ and $\mathbf{q}$ (for $\mathbf{X}$ and $\mathbf{y}$ respectively) with $\mathbf{E}$ and $\mathbf{f}$ being the corresponding residual errors, and a linear relation $\mathbf{R}$ linking both blocks:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E} \qquad \mathbf{y} = \mathbf{U}\mathbf{q}^T + \mathbf{f} \tag{5}$$

$$\mathbf{U} = \mathbf{T}\mathbf{R} \tag{6}$$

where matrices $\mathbf{T}$ and $\mathbf{U}$ are the projections from $\mathbf{X}$ and $\mathbf{y}$ respectively. Specifically, each of the columns of the $\mathbf{T}$ matrix represents one of the latent variables of $\mathbf{X}$.

The NIPALS algorithm [Wold, 1966] is also used to solve this optimization problem. In this case, $\mathbf{b}$ is estimated as:

$$\mathbf{b} = \mathbf{R}\mathbf{q}^T \quad \text{where} \quad \mathbf{R} = \mathbf{W}(\mathbf{P}^T\mathbf{W})^{-1} \tag{7}$$

where $\mathbf{W}$ is an internal weight matrix used by the algorithm that accounts for the correlation between $\mathbf{X}$ and $\mathbf{U}$. An exhaustive description of the NIPALS algorithm for PLSR can be found in [Geladi and Kowalski, 1986].

Since PLSR is a much more sophisticated model than PCA, different elements of the PLSR model can be used to obtain reduced feature sets. In addition to the regressors-based selection method previously described in [Specia et al., 2009b], we propose one new feature selection method, variance importance in projection, and one new feature extraction method, PLSR projection. Similarly to PCA-P, the computational complexity of these three PLSR-based DR methods is also given by the complexity of the NIPALS algorithm, $O(r \cdot m \cdot n)$.

Feature Importance in Regression

Let us consider a linear model such as the one used by PLSR:

$$\hat{y} = b_0 + b_1 x_1 + \cdots + b_i x_i + \cdots + b_m x_m \tag{8}$$

Regressor scores b_i denote the expected value increment of the predicted quality score $\hat{y}$ by unitary increment of feature x_i, i.e., they denote the importance of each feature in the regression. However, due to the usually different scale of the features, these values cannot be directly compared; first data need to be standardized by subtracting the feature mean from the raw data values and dividing the difference by the standard deviation. Standardized features become dimensionless, and then regressors are directly comparable. We thus can create a reduced set of features by selecting them in descending regressor absolute value ($\mathbf{b}$ in Equation (4)). This method, first proposed by [Specia et al., 2009b], is labeled FIR in the experiments.

Variance Importance in Projection

Given the weight matrix $\mathbf{W}$, we can compute the variance importance in projection (VIP) [Chong and Jun, 2005] of the features. VIP is a score that evaluates the importance of each feature to find the r latent variables. Therefore, similarly as done for RFS in Section 3.3.1, we propose to select subsets of top-scoring features according to their VIP. The VIP score for the k^{th} feature is given by:

$$\text{VIP}_k = \sqrt{\frac{m \sum_{j=1}^{r} \left(\frac{w_{kj}}{||\mathbf{w}_j||} \right)^2 \text{ESS}_j}{\sum_{j=1}^{r} \text{ESS}_j}} \tag{9}$$

where m is the number of original features, $\text{ESS}_j = b_j^2 \mathbf{t}_j^T \mathbf{t}_j$ is the square of the contribution of the j^{th} latent variable to the score predicted by the PLSR model, $\mathbf{t}_j$ is the j^{th} column of matrix $\mathbf{T}$, b_j is the j^{th} regressor coefficient in $\mathbf{b}$, and $\frac{w_{kj}}{||\mathbf{w}_j||}$ is the normalized value of weight w_{kj}.

PLSR Projection

The latent variables are linearly independent, and each of them accounts for the maximum co-variability between $\mathbf{X}$ and $\mathbf{y}$ not explained by previous latent variables. Thus, we propose to obtain a reduced feature set by extracting the first r latent variables, i.e. the first r columns in matrix $\mathbf{T}$. In contrast to PCA, the latent variables computed by PLSR take into account the relation between the features $\mathbf{X}$ and the quality scores $\mathbf{y}$. Therefore, in addition of being linearly independent, we expect the latent variables to attain more predictive potential than the equivalent number of principal components. This feature extraction method is labeled PLS-P in the experiments.

4 Experimental Setting

4.1 Data

We computed quality scores for translations of the English-Spanish news evaluation data used in the shared QE task[2] featured at the 2012 workshop on statistical MT [Callison-Burch et al., 2012]. Those translations were generated by a phrase-based MT system [Koehn et al., 2007] trained on the Europarl and News Commentaries corpora as provided for the shared translation task[3]. Evaluation

[2] `http://statmt.org/wmt12/quality-estimation-task.html`

[3] `http://statmt.org/wmt12/translation-task.html`

data contains 1832 translations for training and 422 translations for test. Each translation was manually scored by several professional translators in terms of post-editing effort according to the following scheme:

1 - The translation is incomprehensible. It must be translated from scratch.
2 - About 50%–70% of the translation needs to be edited to be publishable.
3 - About 25%–50% of the translation needs to be edited.
4 - About 10%–25% of the translation needs to be edited.
5 - The translation is clear and intelligible. It requires little to no editing.

The final quality score of each translation (a real number in the range $[1, 5]$) is the average of the scores given by the different experts. Additionally, for each translation the corresponding source sentence, and decoding information (decoding graph and 1000-best translations) are available. We used these and the training data of the shared translation task to compute the features of each translation.

4.2 Features

We extract a total of 480 features described in previous works for translation QE [Blatz et al., 2004, Ueffing and Ney, 2007, Sanchis et al., 2007]. Some of these features are highly-correlated, for example, we consider both the translation probability and the perplexity given by a language model. As described in Section 3.1, working with such redundant features involves several learning issues. However, these inherent learning issues make translation QE a task where DR techniques may lead to important improvements in prediction accuracy.

Following [Specia et al., 2009b], we consider both black-box and glass-box features. On the one hand, black-box features (B) can be extracted given only the input sentence and the translation produced by the MT system, i.e. the source and target sentences, and possibly additional monolingual or parallel data. On the other hand, glass-box (G) features may also depend on some aspect of the translation process.

We distinguish between sentence- and subsequence-based features. Sentence-based features consider the translated sentence as an atomic unit and represent the translation as a whole. In contrast, subsequence-based features consider the translation as a sequence, and are computed by combining the feature scores of the subsequences (words or sequences thereof) contained in each translation.

4.2.1 Sentence-Based Features

- Source and translation lengths, and their ratio. (B, 3 features)
- Source and translation probabilities, probabilities divided by length, and perplexities computed by language models of order one to five. (B, 30 features)
- Translation probability, probability divided by translation length, and perplexity computed by language models of order one to five trained on the complete 1000-best file, and in the particular 1000-best translations of each source sentence. (G, 3 indicators $\times$ 5 orders $\times$ 2 training corpora = 30 features)
- Average length of the 1000-best translations, vocabulary size of the 1000-best translations divided by average length, and 1000-best vocabulary size divided by source length. (G, 3 features)

- Proportion of death nodes in the decoding search graph. (G, 1 feature)
- Number of source phrases of sizes one to six used in decoding. (G, 6 features)
- Number and average size of the alternative translations considered in decoding for source phrases of sizes one to six. (G, 12 features)

4.2.2 Subsequence-Based Features

We represent each subsequence feature by five sentence-level indicators: the average value of the subsequence scores in the translation, and the percentage of scores belonging to each frequency quartile[4]. Each method represent a different approach to summarize the subsequence scores. The average value is a rough indicator that measures the "middle" value of them, while the quartile percentages are more fine-grained indicators that denote how spread out the scores are. We compute the following features for subsequences of sizes one to four:

- Number of translation options for each source word in a Model-1 lexicon trained on the translation task data. (B, $1 \times 5 = 5$ features)
- Frequencies of source sentence subsequences in the training data of the translation task. (B, 4 sizes $\times$ 5 = 20 features)
- Confidence score of each translation word computed by a Model-1 lexicon as in [Ueffing and Ney, 2007]. (B, $1 \times 5 = 5$ features)
- Posterior probabilities of translation subsequences computed on the 1000-best translations [Ueffing et al., 2003]. We follow [Sanchis et al., 2007] and use four different criteria to align the subsequences of the translation to the subsequences of the alternative 1000-best translations, and three different weighting schemes to score each aligment. The accumulated score of the alignments of each subsequence is normalized to obtain the posterior probability of the subsequence. (B, 4 sizes $\times$ 4 criteria $\times$ 3 weightings $\times$ 5 = 240 features)
- Confidence scores of the translation subsequences computed from the corresponding posterior probabilities by a smoothed naïve Bayes classifier as in [Sanchis et al., 2007]. We used three position correctness criteria to automatically generate the reference correctness labels required to train the classification model. (B, 4 sizes $\times$ 3 criteria $\times$ 5 = 60 features)

We also compute the number of words in the translation with zero ($< 10^{-7}$) confidence according to the Model-1 lexicon (B, 1 feature), the number of source subsequences that do not appear in the training data of the translation task (B, 4 sizes = 4 features), the number of translation subsequences with zero ($< 10^{-7}$) posterior probability (B, 4 sizes $\times$ 4 criteria $\times$ 3 weightings = 48 features), and the number of translation subsequences classified as correct by the naïve Bayes classifier (B, 4 sizes $\times$ 3 criteria = 12 features).

4.3 Machine Learning Models

Now, we describe the particular learning models ($\mathbb{M}_{\boldsymbol{\theta}}$ in Section 2) tested in the experiments. We use the WEKA [Hall et al., 2009] package to estimate the values of the free parameters $\boldsymbol{\theta}$ that best fit training data.

[4] Frequency quartiles were computed on the training data of the shared translation task.

4.3.1 Linear Regression

Linear regression assumes a linear relationship between the prediction value y_i and the vector of features $\mathbf{x}_i$ which is modeled by a vector of weights $\boldsymbol{\theta}^T = (\theta_1, \ldots, \theta_m)$. Formally, linear regression models take the form of a set of equations:

$$y_i = \theta_1 x_{i1} + \cdots + \theta_m x_{im} + \epsilon_i, \quad i = 1, \ldots, n \tag{10}$$

where n is the number of training samples, m is the number of features, and ϵ_i are zero-centered Gaussian error variables. Often all equations are stacked together and written in matrix form:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\epsilon} \tag{11}$$

The most common technique to estimate the free parameters $\boldsymbol{\theta}$ of linear models is known as least squares estimation. This method minimizes the sum of squared errors, and leads to a closed-form expression for the optimum values of $\boldsymbol{\theta}$:

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{12}$$

Additionally, different regularization techniques are usually implemented to prevent ill-posed learning problems when multicollinearity is present. Regularization techniques deliberately introduce bias into the estimation of $\hat{\boldsymbol{\theta}}$ to penalize complex models. In the experiments, we used ridge and LASSO regression [Tibshirani, 1996]. Both methods constraint the norm of the parameter vector (L^2-norm ridge and L^1-norm LASSO) to be lower than a given value γ.

4.3.2 Support Vector Machines

In practice, few natural phenomena exhibit a linear relationship between their explanatory variables $\mathbf{x}$ and the corresponding dependent variable y. Thus, linear regression cannot adequately describe such nonlinear phenomena.

Support vector machines [Cortes and Vapnik, 1995] (SVMs) are a class of machine learning models that, as linear regression, assume a linear relationship between $\mathbf{X}$ and $\mathbf{y}$. However, prior to any calculation, SVMs project the data into an alternative space. This projection, defined by a kernel function $\varphi(\mathbf{x})$, may be nonlinear; thus, though a linear relationship is learned in the projected feature space, this relationship may be nonlinear in the original input space. Choice of the kernel determines whether the resulting SVM is a polynomial regressor, a two-layer neural network, a radial basis function machine, or some other learning machine.

The linear relationship is estimated as a regularized (L^2-norm) optimization problem. In contrast to linear regression, the SVM model depends only on a subset of the training data, because the cost function for building the model does not care about those training samples that already lie within a given margin. There exist several specialized algorithms for solving the quadratic programming problem that arises. For example, sequential minimal optimization [Platt, 1999] breaks the problem down into 2-dimensional sub-problems that can be solved analytically.

Preliminary experiments studying different kernels showed that radial basis kernel obtained among the best results and additionally was easier to train than other kernels such as polynomial kernels. Therefore, in the experimentation we used SVMs with a radial basis kernel.

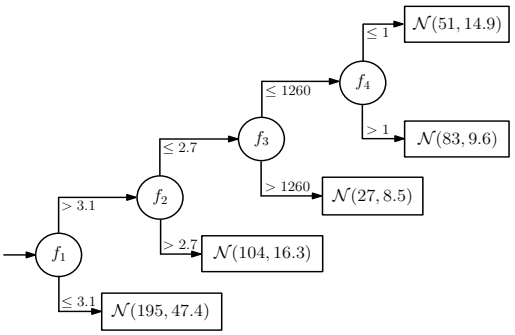

Fig. 4 Example of a regression tree. It uses four feature comparisons to partition the data-space, and gaussian normal distributions to model the data on each of the five partitions.

4.3.3 Regression Trees

Typical regression models, such as linear regression or SVMs, are global. In other words, there is a single predictive formula holding over the entire data-space. When the data has lots of features which interact in complicated, nonlinear ways, assembling a single global model can become a very difficult problem. An alternative regression approach is to recursively partition the data-space into smaller regions, until they are simple enough to fit elemental models to them.

Regression trees use a tree structure to represent such a recursive partition. Each of the terminal nodes of the tree represents a region of the partition, and has attached to it a simple model which applies in that region only. We start at the root node of the tree, and ask a sequence of questions about the features. The interior nodes are labeled with questions, and the edges between them are labeled with the answers. Typically, each question refers to only a single feature, and has a yes or no answer, e.g., "Is Horsepower > 50?" or "Is GraduateStudent $==$ FALSE?". Features can be of different types (continuous, discrete, categorical, etc), and more-than-binary questions can be done, but these can always be accommodated as a larger binary tree. Figure 4 shows an example of a regression tree using gaussian normal distributions to model the data on each partition.

Once we fix the tree structure, local models are completely determined, and easy to find, so all the effort should go into finding a good tree structure, which is to say into finding a good partitioning of the data-space. In our experiments, we specifically use M5 regression tree [Quinlan, 1992] because one of the best submissions to the 2012 QE task [Callison-Burch et al., 2012] used such tree model.

5 Experiments

5.1 Methodology

We extracted the 480 features described in Section 4.2 for each of the automatic translations in the evaluation data of the QE task. As a result, we obtained a training and a test set of 480-dimensional real vectors with 1832 and 422 samples respectively. All features were standardized by subtracting the feature mean from the raw values, and dividing the difference by the standard deviation.

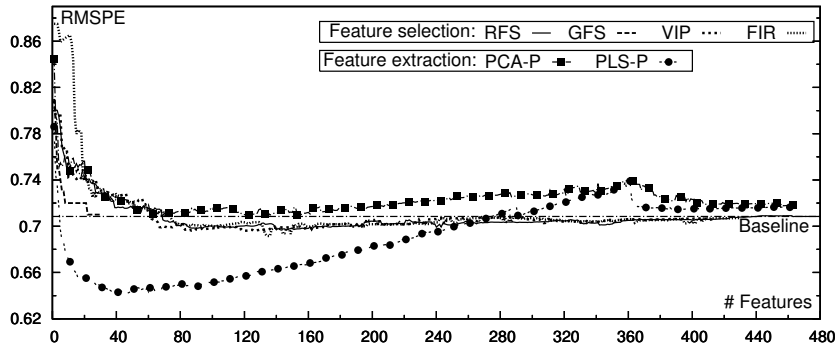

Fig. 5 SVMs cross-validation training results for different DR methods as a function of the size of the reduced feature set. In comparison, the baseline SVM trained on the 480 original features obtained 0.71 RMSPE. Best PLS-P results were statistically better than the rest.

Then, we carried out an exhaustive experimentation to test the different DR methods described in Section 3, and to study how their use affect the prediction performance of the different learning models presented in Section 4.3. We tested all 18 combinations of a DR method and a learning model in a series of two-step experiments as depicted in Figure 1. Since we did not know the optimum size r of the reduced feature set (see Section 3.2), each experiment involved several trains of the model with reduced feature sets of different sizes. For each size, we performed a cross-validation training with ten randomly-chosen data splits to learn the meta-parameters of the models, e.g. the γ parameter of ridge regression.

5.2 Evaluation Criteria

Since we view DR as a way to build robust prediction models, we evaluated each DR method by the prediction accuracy of the regression models trained on the corresponding reduced feature sets. The performance of a regression model is usually measured by the average error of the predictions $\hat{\mathbf{y}} = \{\hat{y}_1, \ldots, \hat{y}_n\}$ with respect to the actual scores $\mathbf{y} = \{y_1, \ldots, y_n\}$. Specifically, we compute the root mean squared prediction error (RMSPE) as in [Specia et al., 2009b]:

$$\text{RMSPE}(\mathbf{y}, \hat{\mathbf{y}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{13}$$

where n is the number of test samples. RMSPE quantifies the average deviation of the estimation with respect to the expected score. I.e. the lower the value, the better the performance of the learning model.

5.3 Cross-Validation Training Results

We now present the results for cross-validation training experiments. The conclusions were similar for all learning models. Thus, to keep the presentation clear, we only show RMSPE results using SVMs as learning model. Figure 5 shows SVMs cross-validation RMSPE for the different DR methods presented in Section 3.

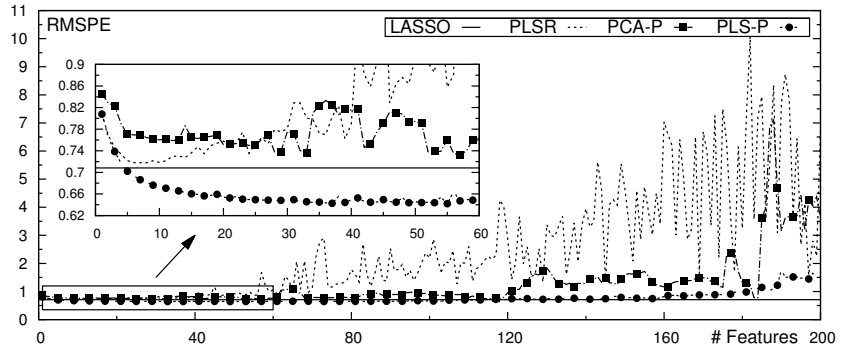

Fig. 6 Cross-validation training results for linear ridge regression using PCA-P and PLS-P DR methods. We also display baseline results for LASSO regression, and PLSR (Equation (4)). As for SVMs in Figure 5, PLS-P outperforms any other tested approaches. Additionally, note that the use of the proposed two-step training procedure, see Figure 1, allows to smooth the rough learning curves obtained by conventional PLSR, compare PLSR and PLS-P learning curves.

The results of the four feature selection methods were very close, and all of them slightly outperformed the baseline SVM model trained with the whole 480-dimensional feature set (0.71 RMSPE). Rank of feature selection (RFS), variance importance in projection (VIP), and feature importance in regression (FIR) obtained virtually the same results. Their performance improved as more features were selected, and they required to select above 100 features to reach their top performance. Then, as more features were selected their results slowly converged to the performance of the baseline model. Since these methods do not take into account the correlations that may exist between the features, their reduced feature sets were highly-redundant; which explains the large number of features they needed to stabilize. In contrast, greedy forward selection (GFS) obtained great improvements with few features. However, its higher computational complexity complicates its practical deployment; reason why we carried out experiments only up to 30 features. Nevertheless, with only these 30 features it was able to equal the performance of the baseline model trained on the original 480 features.

Regarding the two feature extraction methods, they exhibited important differences in performance. PCA projection (PCA-P) obtained worse results than the four feature selection methods, moreover it did not even improve the results of the baseline model. PCA-P reached its top performance when $\sim$ 120 principal components were generated, and it slightly deteriorated as the number of features increased. In contrast, PLSR projection (PLS-P) obtained much better results consistently outperforming PCA-P and all feature selection methods. Moreover, with only five latent variables PLS-P was able to outperform the baseline SVM model trained with 480 features, and it only required 44 features to reach its top performance. Additionally, the performance difference observed between the best result of PLS-P and the rest of the DR methods was significant with a probability of improvement of 95% according to a pair-wise bootstrap analysis [Bisani and Ney, 2004]. These results indicate that PLS-P generates more "information-dense" features that constitute a better summary the original high-dimensional feature set.

Although results in Figure 5 are representative for all learning models, we observed important differences in the stability of the learning curves of the different

	Ridge regression		Support vector machines		Regression trees	
	NF	RMSPE	NF	RMSPE	NF	RMSPE
Original features	480	0.79	480	0.97	480	0.87
RFS	69	0.83	162	0.84	72	0.91
GFS	22	0.82	22	0.83	16	0.89
VIP	67	0.83	126	0.83	57	0.88
FIR	82	0.83	136	0.82	71	**0.86**
PCA-P	57	0.83	122	0.81	31	0.90
PLS-P	55	**0.78**	44	**0.78**	9	0.88

Table 1 Prediction results (RMSPE) on the test set for the different DR methods and learning models under study. NF denotes the number of features of the reduced test sets. Best results for each learning model are displayed in bold. As a comparison, the result for a linear LASSO regression model was 0.82 RMSPE.

models. Figure 6 displays training cross-validation results for linear ridge regression using PCA-P and PLS-P as DR methods. We present results only for these two DR methods for simplicity. Since the baseline ridge model (trained with the original 480 features) obtained a dreadful RMSPE of 16.73, we present results for two alternative linear regression baselines: a LASSO regression model also trained with all the original 480 features, and for the predictions directly generated by the PLSR model according to Equation (4). In contrast to the results for SVMs, we now obtained rougher learning curves with large performance variations, particularly as we increased the number of features. However, the proposed two-step training procedure (see Figure 1) partially addresses this problem. This is exemplified in the comparison between PLSR and PLS-P. Both methods use a linear model to predict the quality scores from the projected data, however PLS-P obtains a much smother learning curve than PLSR. Finally, we could extract the same conclusion as for SVMs: among all the tested DR methods, PLS-P is the best performing one allowing us to improve the performance of even sophisticated regularized models such as SVMs or linear LASSO regression.

These results show that the proposed two-step training is an efficient procedure to deal with noisy and correlated input features, and it can outperform models such as LASSO regression and PLSR that integrate DR in their formulation.

5.4 Blind Test Results

Next, for each combination of a DR method and a learning model, we built a new model using the full training set and the best configuration (size of the reduced feature set, and values of the meta-parameters of the learning model) observed in the corresponding cross-validation experiments. Then, we reduced the test set to the optimal dimension estimated by cross-validation training, and tested the performance of the new trained model for the reduced test set. Table 1 displays these results. In contrast to the previous cross-validation experiments, results on the test set were quite different for the three learning models. While for SVMs, the use of DR improved the performance of the baseline model trained on the 480 original features, no improvement was obtained at all for linear ridge regression, or for regression trees. This was quite a surprising result. Given the large improve-

ments over the baseline obtained in the cross-validation experiments, we expected to obtain similar improvements over baseline in test.

To better understand these results, we carried out a multivariate Hotelling's two-sample T-squared test [Hotelling, 1931, Anderson, 1958] to study the possible differences that may exist between the training and test feature sets. The objective of such tests is to determine whether two samples, in our case the training and test sets, have been sampled from the same population or not. The result of the test indicated that there were a statistically significant difference between the two feature sets ($p < 0.01$), and thus they seemed to come from different populations. Since the training and test translations come from a similar news domain [Callison-Burch et al., 2012], we hypothesize that the difference between the feature sets was mainly due to the specific chosen features. In fact, results of individual Student's two-samples t-tests for each feature showed that 260 of the 480 extracted features were significantly different ($p < 0.01$) between training and test. For example, the number of words with zero posterior probability is significantly different between the samples in training ($\mu = 1.7, \sigma = 1.39$) and test ($\mu = 0.90, \sigma = 0.80$).

In addition to the relatively small number of training samples (1832), this mismatch between the distribution of the features values in the training and test sets may be the explanation for the unintuitive results displayed in Table 1, compared to the cross-validation results in Figure 5 and Figure 6 where PLS-P largely improved Baseline. DR methods obtain a reduced feature set based on the training set, thus, if the training set is not representative of the test set, as proved by the Hotteling's test, the computed reductions cannot be adequate for test. Also, the fact that SVMs actually improved baseline test results when DR methods were used can be explained by the fact that SVMs are more complex models than ridge regression and regression trees. SVMs performance is more heavily penalized due to the lack of data. Thus, we hypothesize that the use of reduced feature sets, even if they are inadequate, allows to improve SVMs performance[5]. Despite these problems, Table 1 shows that PLS-P was the top-performing DR method for linear regression and SVMs. However, for regression trees, all methods obtained similar results. This fact indicates that regression trees were not able to fully exploit the more "information-dense" features generated by PLS-P. Since these "information-dense" features are the combination of several of the original features, we hypothesize that they are also more difficult to be partitioned into regions to create the tree structure of the model. Nevertheless, even in this pessimistic setting PLS-P generated reduced sets of features that performed similarly as the original 480 features. We consider that, given the cross-validation results in Section 5.3, larger performance improvements could be expected whenever an adequate set of features, and/or a large enough training set are provided.

Additionally, since the time required to train the model and to perform the prediction are directly related to the number of features, an additional advantage of DR methods is that they can improve the practical deployment of QE technology by reducing training / test time. For example, training an SVM model (including meta-parameter optimization) using the original 480 features typically required $\sim$30 hours in our test machine, while the training time using the optimal 44 latent variables extracted by PLS-P was below three hours.

[5] Few features imply few parameters to be estimated with the same amount of data.

5.5 Feature Analysis

We perform a final analysis on the features that contribute more to create the reduced feature sets. For feature selection methods, we simply looked for the most frequently selected features. For PCA-P and PLS-P, that combine the original features into new features (the principal components and the latent variables respectively) by a matrix transformation ($\mathbf{P}$ in Equations (2) and (6)), we computed the contribution of each feature by summing up the absolute value of the scores in the corresponding column of $\mathbf{P}$. We then can highlight the following features:

- Source and translation lengths and language model probabilities.
- Vocabulary of the 1000-best translations divided by their average length.
- Number of source phrases of size one used in decoding.
- Number of source phrases used in decoding.
- Frequencies of source subsequences (sizes one to four). †
- Posterior probabilities of translation subsequences (sizes one and two).†
- Probability of the translation subsequences (sizes one and two) by a naïve Bayes' classifier.†

Additionally, for the subsequence-based features (marked with †) the most important sentence-level indicators were specifically the average value of the feature, and the number of subsequences in the first and fourth quartile.

Despite this general result, we observed slight differences in the importance of each feature according to the different methods. For example, the simple RFS method tended to add lots of similar features, such as the posterior probabilities of the target subsequences, which independently are quite informative but together are highly redundant. In contrast, the more computationally complex GFS method selected only one or two features that represent all features of the same type.

6 Summary and Future Work

We have proposed two novel DR methods based on PLSR and compared them against several DR methods previously used in the QE literature. The DR methods under consideration can be classified by their theoretical background: statistical multivariate analysis or heuristic methods, or by how they perform the reduction: feature selection or feature extraction methods. Moreover, we have studied how DR affect the prediction performance of different learning models.

We have evaluated each DR method by the prediction performance of the learning models trained on the corresponding reduced feature set. This quality measure has the advantage of automatic evaluation, and, using identical pipelines to train the models, it allows us to accurately compare the different DR methods. The key results of the experiments are as follows:

- Feature extraction methods can outperform feature selection methods.
- Methods based on multivariate analysis can outperform heuristic methods.
- To obtain a good prediction performance, DR methods have to take into account the scores to be predicted.
- The performance-wise ranking of the DR methods is to a great extent independent of the chosen learning model.

– However, for simple models such as linear regression the use of some DR methods may result in erratic learning curves.

One of the proposed DR methods, PLS-P, can be seen as a summary of the conclusions: a feature extraction method based on multivariate analysis that takes into account the values to be predicted to perform the reduction. Thus, it consistently obtained the best results in the cross-validation training experiments. Additionally, the unintuitive results observed in test (where PLS-P did not improve baseline) can be explained by a difference between the distribution of the features in training and test. The use of statistical tests to detect this problem is then a necessary tool to build robust QE systems.

As future work, we plan to explore additional feature selection methods based on redundancy minimization and relevancy maximization, and new feature extraction methods based in nonlinear projections, and also to integrate statistical tests over the features as a preliminary step to filter out problematic features. Additionally, we also plan to investigate automatic techniques to estimate the internal dimension r of the problem, interactions between the features, and outliers detection methods to efficiently use of the (usually) scarce training data.

Acknowledgements Work supported by the European Union Seventh Framework Program (FP7/2007-2013) under the CasMaCat project (grans agreement n^{o} 287576), by Spanish MICINN under TIASA (TIN2009-14205-C04-02) project, and by the Generalitat Valenciana under grant ALMPR (Prometeo/2009/014).

References

[Amaldi and Kann, 1998] Amaldi, E. and Kann, V. (1998). On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1-2):237–260.

[Anderson, 1958] Anderson, T. W. (1958). *An Introduction to Multivariate statistical Analysis*. Wiley, New York.

[Avramidis, 2012] Avramidis, E. (2012). Quality estimation for machine translation output using linguistic analysis and decoding features. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 84–90.

[Bellman, 1961] Bellman, R. E. (1961). *Adaptive control processes: a guided tour*. Rand Corporation Research studies. Princeton University Press.

[Bisani and Ney, 2004] Bisani, M. and Ney, H. (2004). Bootstrap estimates for confidence intervals in asr performance evaluation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 409–412.

[Blatz et al., 2004] Blatz, J., Fitzgerald, E., Foster, G., Gandrabur, S., Goutte, C., Kulesza, A., Sanchis, A., and Ueffing, N. (2004). Confidence estimation for machine translation. In *Proceedings of the international conference on Computational Linguistics*, pages 315–321.

[Callison-Burch et al., 2012] 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.

[Chong and Jun, 2005] Chong, I. and Jun, C. (2005). Performance of some variable selection methods when multicollinearity is present. *Chemometrics and Intelligent Laboratory Systems*, 78(1–2):103–112.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

[Gamon et al., 2005] Gamon, M., Aue, A., and Smets, M. (2005). Sentence-Level MT evaluation without reference translations: beyond language modeling. In *Proceedings of the conference of the European Association for Machine Translation*.

[Gandrabur and Foster, 2003] Gandrabur, S. and Foster, G. (2003). Confidence estimation for text prediction. In *Proceedings of the Conference on Computational Natural Language Learning*, pages 315–321.

[Geladi and Kowalski, 1986] Geladi, P. and Kowalski, B. R. (1986). Partial least-squares regression: a tutorial. *Analytica Chimica Acta*, 185(1):1–17.

[González-Rubio et al., 2010] González-Rubio, J., Ortiz-Martínez, D., and Casacuberta, F. (2010). Balancing user effort and translation error in interactive machine translation via confidence measures. In *Proceedinss of the meeting of the Association for Computational Linguistics*, pages 173–177.

[González-Rubio et al., 2012] González-Rubio, J., Sanchís, A., and Casacuberta, F. (2012). Prhlt submission to the wmt12 quality estimation task. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 104–108.

[Guyon and Elisseeff, 2003] Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Machine Learning Research*, 3:1157–1182.

[Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: an update. *SIGKDD Explorations Newsletter*, 11(1):10–18.

[Hotelling, 1931] Hotelling, H. (1931). The Generalization of Student's Ratio. *Annals of Mathematical Statistics*, 2(3):360–378.

[Koehn et al., 2007] 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 Association for Computational Linguistics, demonstration session.*

[Kohavi and John, 1997] Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324.

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

[Platt, 1999] Platt, J. C. (1999). Using analytic QP and sparseness to speed training of support vector machines. In *Proceedings of the conference on Advances in neural information processing systems II*, pages 557–563.

[Quinlan, 1992] Quinlan, R. J. (1992). Learning with continuous classes. In *Proceedings of the Australian Joint Conference on Artificial Intelligence*, pages 343–348.

[Quirk, 2004] Quirk, C. (2004). Training a sentence-level machine translation confidence measure. In *Proceedings of conference on Language Resources and Evaluation*, pages 825–828.

[Sanchis et al., 2007] Sanchis, A., Juan, A., and Vidal, E. (2007). Estimation of confidence measures for machine translation. In *Proceedings of the Machine Translation Summit XI*, pages 407–412.

[Scott and Thompson, 1983] Scott, D. W. and Thompson, J. R. (1983). Probability density estimation in higher dimensions. *Computer Science and Statistics: Proceedings of the Fifteenth Symposium on the Interface*, pages 173–179.

[Soricut et al., 2012] Soricut, R., Bach, N., and Wang, Z. (2012). The SDL Language Weaver Systems in the WMT12 Quality Estimation Shared Task. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 145–151, Montréal, Canada.

[Soricut and Echihabi, 2010] Soricut, R. and Echihabi, A. (2010). TrustRank: inducing trust in automatic translations via ranking. In *Proceedings of the Meeting of the Association for Computational Linguistics*, pages 612–621.

[Specia et al., 2009a] Specia, L., Saunders, C., Wang, Z., Shawe-Taylor, J., and Turchi, M. (2009a). Improving the confidence of machine translation quality estimates. In *Proceedings of the Machine Translation Summit XII.*

[Specia et al., 2009b] Specia, L., Turchi, M., Cancedda, N., Dymetman, M., and Cristianini, N. (2009b). Estimating the sentence-level quality of machine translation systems. In *Proceedins of the meeting of the European Association for Machine Translation*, pages 28–35.

[Tibshirani, 1996] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288.

[Ueffing et al., 2003] Ueffing, N., Macherey, K., and Ney, H. (2003). Confidence measures for statistical machine translation. In *Proceedings of the MT Summit IX*, pages 394–401.

[Ueffing and Ney, 2007] Ueffing, N. and Ney, H. (2007). Word-level confidence estimation for machine translation. *Computational Linguistics*, 33:9–40.

[Wold, 1966] Wold, H. (1966). *Estimation of Principal Components and Related Models by Iterative Least squares*, pages 391–420. Academic Press, New York.