How Good Are Your Phrases? Assessing Phrase Quality with Single Class Classification

Nadi Tomeh † *Marco Turchi* ‡ *Guillaume Wisinewski* † *Alexandre Allauzen* † *François Yvon* †

† LIMSI-CNRS and Université Paris-Sud
BP 133, 91403 Orsay, France
{nadi,wisniews,allauzen,yvon}@limsi.fr

‡ European Commission - Joint Research Centre
Via Fermi 2749, 21027 Ispra, Italy
marco.turchi@jrc.ec.europa.eu

Abstract

We present a novel translation quality informed procedure for both extraction and scoring of phrase pairs in PBSMT systems.

We reformulate the extraction problem in the supervised learning framework. Our goal is twofold. First, We attempt to take the translation quality into account; and second we incorporating arbitrary features in order to circumvent alignment errors. One-Class SVMs and the Mapping Convergence algorithm permit training a single-class classifier to discriminate between useful and useless phrase pairs. Such classifier can be learned from a training corpus that comprises only useful instances. The confidence score, produced by the classifier for each phrase pairs, is employed as a selection criteria. The smoothness of these scores allow a fine control over the size of the resulting translation model. Finally, confidence scores provide a new accuracy-based feature to score phrase pairs.

Experimental evaluation of the method shows accurate assessments of phrase pairs quality even for regions in the space of possible phrase pairs that are ignored by other approaches. This enhanced evaluation of phrase pairs leads to improvements in the translation performance as measured by BLEU.

1. Introduction

In phrase-based statistical machine translation (PBSMT) systems, the best translation e^* for the source sentence f is selected by maximizing the conditional probability $P(e|f)$. This term is computed by marginalizing over all possible joint segmentations and alignments of the source and target a, as $P(e|f) = \sum_a P(a, e|f)$, where

$$P(a, e|f) = Z(f, \lambda)^{-1} \exp \sum_{i=1}^{k} \lambda_i h_i(e, a, f), \qquad (1)$$

where Z is a normalization constant and each h_i is a feature function that decomposes over atomic phrase translations and λ_i is the corresponding feature weight which scales each feature's contribution to the final model score. Typical features include language model, reordering and conditional phrase translation probabilities, word and phrase penalties and lexical weights. Additional feature functions are also investigated in the literature [1, 2].

For a PBSMT to produce a good translation, two conditions must be met: (i) good translations must exist in the search space of the decoder, and (ii) the model score must be (positively) correlated with translation quality. The first condition mainly depends on the coverage of the phrase translation candidates that are stored in the *phrase table*. A maximal coverage can be achieved by including all possible phrase pairs encountered in the training corpus: in this setting, the model scores are the only information used to select suitable translations during decoding. Given the sheer number of possible phrase pairs, the vast majority of which are in fact irrelevant, taking all possible phrase pairs into account is impractical, and all methods for constructing phrase tables comprise a first step where the quality of each phrase pair is estimated, and where phrase pairs that look too bad are filtered out.

For this purpose, the standard approach [3] relies on binary scores deduced from the underlying word alignment and discards all phrase pairs that are not consistent with it. This technique, however, does not let the user control the size of the resulting phrase table. More flexibility is gained by employing pruning techniques that need to be applied *a posteriori* as in [4], where a second scoring step is used to filter large phrase tables. An alternative is to use *weighted alignment matrices*, assigning each phrase a smooth score in the interval $[0, 1]$ [5, 6]. Unlike computing the model score, which typically combines several features, the phrase extraction approach is mostly heuristic and relies primarily on word or phrase alignments [7, 8]. These alignments are error-prone and they are obtained as the result of complex optimization programs maximizing an objective function (the likelihood of the training data) that correlates only indirectly with the translation quality. If the same can be said of the feature functions used in the model score, the combined model is however enhanced during *tuning* to better correlate with translation quality, where feature weights are set so as to optimize an automatic quality measure, such as BLEU, on held-out data via Minimum Error-Rate Training (MERT) [9].

As an attempt to improve these procedures, we study in this paper novel extraction and scoring procedures that : (1) can straightforwardly handle arbitrary feature functions; (2) have a direct relationship to translation quality; and (3) give the user a finer control over the size of the phrase table. This study has both practical and methodological implications. From a practical perspective, the scenario we consider is the use of a small set of parallel sentences, from which we would like to extract as much phrases as possible, so as to ensure the larger possible coverage. In this setting, finding better ways to score phrases might prove necessary. From a more methodological perspective, our goal is to better understand the properties that make a good phrase pair.

To fulfill these goals, we propose to reformulate the extraction problem in a supervised learning framework. Extracting or discarding a phrase pair is indeed a binary decision, which can be learned, using a set of labeled training examples. We would like to make such decisions based on the expected utility of phrase pairs in a translation pipeline. This leaves us with two problems: (a) defining an operational notion of utility, and (b) finding examples of useful

and useless phrase pairs with respect to this definition.

As discussed below, a good approximation of (a) will be to consider that phrase pairs participating in derivations of good translations are useful; such phrase pairs are relatively easy to collect by looking at good derivations of test data, and will provide us with sets of *positive* training examples. Obtaining negative examples proves more challenging, and would require to examine all the (non-optimal) derivations of our test data, which is clearly unrealistic. A nice walk-around is to use *single-class classification* [10] techniques, which aim at learning concepts in the absence of counter examples, by distinguishing one class of (positive) instances from all other possible instances. Such techniques can handle arbitrary feature functions to represent candidate phrase pairs, thus making the extraction procedure more robust to alignment errors. A useful by-product of the model is the computation of an *accuracy-based feature*, analogous to the proposal in [11]. In short, our main contribution can be viewed as a *novel translation quality informed procedure for both extraction and scoring of phrase pairs*.

The rest of the paper is organized as follows. In Section 2, we motivate the formulation of phrase pair extraction as a single-class classification problem and describe a practical extraction pipeline. The *One-Class SVM (OC-SVM)* [12] and the *Mapping Convergence (MC)* [13] algorithms, which are used to train the single-class classifier are presented in Section 3. In Section 4, we describe the oracle decoder used to label positive examples. Our feature functions, describing various facets of a phrase pair are detailed in Section 5. Experiments are reported in Section 6 before we conclude in Section 7 and 8 with related and future works.

2. Supervised Phrase-Pair Extraction

2.1. Single-Class Classification (SCC)

In this section, we would like to learn the binary decision of extracting or discarding a phrase pair as a supervised classification problem, in which we aim to discriminate *useful (positive)* from *useless (negative)* phrase pairs from a translation perspective. The model decision can then be used as a new feature function to score candidate phrase pairs.

An obvious way to recast this problem as a supervised classification problem requires labeled training examples of both classes, which subsumes an understanding of what makes a useful phrase pair. Such a task is tricky even for humans. From a phrase-based model point of view, a phrase pair is useful if (1) each phrase is an appropriate translation of the other and (2) it combines well with neighbor phrase pairs to produce a good translation. While scores associated to a phrase pair provide evidence regarding the validity of the translational equivalence, the combination aspect is difficult to judge without involving the translation process itself.

We therefore define a positive phrase pair as one that participates in best scoring derivations of good translations, which are easy to obtain. Unfortunately, negative phrase pairs are not so easily found: to identify bad phrase pairs, we would need to examine all the possible translations where they occur and make sure none is acceptable. In fact, the very concept of a negative phrase pair is hard to define, and it is therefore difficult to develop a representative sample of this class. Hopefully, unlabeled examples of phrase pairs abound, a fact that will prove useful later.

A particularly attractive solution in this setting is Single-Class Classification (SCC), which seeks to distinguish one class, for which positive instances exist, from data in a universal set containing one or several classes, for which no sample is available. We assume that the very large set of all possible phrase pairs contains a small set of positive examples $P = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l)$ completed with a large unlabeled set $U = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{l+u})$. The ratio between positive and negative phrase pairs is *unknown*.

2.2. Phrase Translation Training Algorithm

The algorithm described here takes as input a parallel corpus, and uses an oracle decoder and some other resources to compute phrase pair features and output a phrase translation model, in the form of phrase table.

In *step (1)*, we build the set U of phrase pairs that are going to be considered by the algorithm. For each one of them, a set of feature functions is calculated. U can be constructed naively by considering all possible phrase pairs found in the parallel corpus, or by applying some prior knowledge, such as word alignments, to filter the set;

In *step (2)*, the set (or a subset) of phrase pairs in U is used to build a phrase table, using the calculated features as scores. An oracle decoder (Section 4) uses this phrase table on a held-out parallel corpus, to produce the best phrasal derivations of this corpus. The best derivation is the one that maximizes a combination of model score and translation quality metric. All phrase pairs involved in these derivations are labeled as positive phrase pairs in P;

In *step (3)*, we seek to generalize beyond the scope of the phrase pairs actually used by the decoder. As discussed in Section 2.1, the oracle decoder acquires a subset of positive phrase pairs, that we want to expand, by learning its characteristics using a classifier. In the next section, we introduce a One-Class Support Vector Machines (OC-SVM) algorithm, designed to learn from positive examples P only, by estimating the support of their distribution [12]. In practice, this approach is sensitive to the choice of features and parameter settings and is likely to underfit or overfit easily [14]. Therefore, we employ the Mapping Convergence (MC) algorithm [13], a semi-supervised framework, which, in addition to learning from positive examples, exploits unlabeled data to improve the accuracy of the classifier.

In *step (4)*, the best classifier, output of the previous step, is applied to the unlabeled phrase pairs $(U - P)$, estimating to what extent they resemble the positive samples, and which ones ought to be extracted. The distance to the decision boundary (the hyperplane in the SVM feature space) is interpreted as a confidence measure, and used for two purposes: it is thresholded to extract phrase pairs; and injected into the final phrase table as an accuracy-based feature function, similar to [11, 15]. Final phrase table contains all phrase pairs labeled as positive either by the oracle decoder or by the learned classifier. Any subset of the calculated features, in addition to the standard phrase translation probabilities (normalized frequencies) can be used to score phrase pairs in the output phrase table.

Training phrase translation model needs to address precision and recall issues, following an information retrieval scheme [16]. High precision requires that extracted phrase pairs are accurate, while high recall seeks to increase coverage by extracting as much valid phrase pairs as possible. Precision of standard phrase tables can be improved by filtering out most of the entries, using some statistical significance test [4, 17]. On the other hand, there are valid translation pairs in the training corpus that are not learned due to word alignment errors [6]. The algorithm presented here attempts to circumvent alignment errors and increase accuracy by integrating multiple features and combining them discriminatively. At the same time, the threshold on the classifier score presents a control point over the balance between precision and recall, and introduces an additional parameter that can be tuned *via* grid search, for an optimal performance on a specific translation task.

3. Learning the Single-Class Classifier

3.1. One-Class SVM (OC-SVM)

Extensions of SVM have been proposed to allow learning in single-class setting. Both the Support Vector Data Description algorithm (SVDD) [10] and the One-Class SVM (OC-SVM) [12] are shown to be equivalent when data vectors are scaled to unit length. We use OC-SVM, in which the optimization problem is formulated as in ν-SVM parametrization [18], which allows us to control the fraction of outliers.

Given some dataset drawn from an underlying probability distribution, the task of OC-SVMs is to estimate a subset $\mathcal{S}$ of the input space such that the probability of a test point drawn from outside of $\mathcal{S}$ equals some *a priori* specified value between 0 and 1. This is done by estimating a function f which is positive on $\mathcal{S}$ and negative on its complementary, which is an easier problem than full density estimation. The functional form of f is given by a kernel expansion in terms of a subset of the training data; it is regularized by controlling the length of the weight vector in an associated feature space. The expansion coefficients are found by solving a quadratic programming problem, which carries out sequential optimization over pairs of input patterns.

The main idea behind OC-SVMs is to create a hyperplane in the feature space where the projections of data points are separated from the origin with a large margin. The data is separable from the origin if there exists a vector $\boldsymbol{w}$ such that $\forall i, K(\boldsymbol{w}, \boldsymbol{x}_i) > 0$. In such a case, there exists a unique supporting hyperplane. This is always true for the special case of a Gaussian (Radial Basis Function) kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma||\boldsymbol{x}_i - \boldsymbol{x}_j||}$.

As pointed out in [12], there exists a strong connection between OC-SVMs and binary classification. Assuming we have a parametrization $(\boldsymbol{w}, \rho)$ for the supporting hyperplane of a data set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l\}$, then $(\boldsymbol{w}, 0)$ is the parametrization of the maximally separating hyperplane for the labeled data set $\{(\boldsymbol{x}_1, +1), \ldots, (\boldsymbol{x}_l, +1), (-\boldsymbol{x}_1, -1), \ldots, (-\boldsymbol{x}_l, -1)\}$. Also, assuming that we have a maximally separating hyperplane parametrized by $(\boldsymbol{w}, 0)$ for a data set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_l, y_l)\}, (y_i \in \{\pm 1\})$ and with a margin $\rho / ||\boldsymbol{w}||$, we know that the supporting hyperplane for the unlabeled dataset $\{y_1\boldsymbol{x}_1, \ldots, y_l\boldsymbol{x}_l\}$ is parametrized by $(\boldsymbol{w}, \rho)$. For the non-separable case, margin errors in the binary setting correspond to outliers in the one-class case. This connection allows us to reuse the optimization problem of ν-SVM to find the supporting hyperplane, such that in the one-class setting, ν represents an upper bound on the fraction of outliers (margin errors) and a lower bound on the fraction of support vectors.

3.2. Mapping Convergence (MC)

OC-SVM draws a nonlinear boundary around the positive data set in the feature space using two parameters: ν (to control the number of outliers) and γ (to control the smoothness of the boundary). They have the same advantages as regular SVMs, such as efficient handling of high dimensional spaces and nonlinear classification using the kernel trick.

Several attempts to take advantage of large sets of unlabeled data have been studied (see [19] for a survey). The *Mapping Convergence (MC)* algorithm [13] assumes the presence of a "gap" between positive and negative points in the feature space and uses it by incrementally marking as negative unlabeled samples using the *margin maximization* property of SVM. MC has been shown to generate as accurate boundaries as standard SVM with fully labeled data. A key intuition of MC is that negative examples can be sorted by their distance to the decision boundary, the farthest ones being called the *strong negatives*.

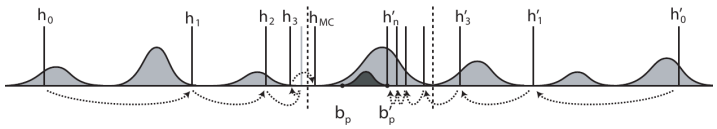

Figure 1: Mapping Convergence in 1-dimensional space

Algorithm 1 Mapping Convergence (MC)

Require: positive data set P; unlabeled data set U; negative data set $N = \phi$; OC-SVM: C_1; SVM: C_2
Ensure: boundary function h_i

1: $h_0 \leftarrow$ train $C1$ on P
2: $\hat{N}_0 \leftarrow$ strong negatives ($\leq$10%) from U by h_0
   $\hat{P}_0 \leftarrow U - \hat{N}_0$
3: $i \leftarrow 0$
4: **while** $\hat{N}_i \neq \phi$ **do**
5: $\quad N \leftarrow N \cup \hat{N}_i$
6: $\quad h_{i+1} \leftarrow$ train C_2 on P and N
7: $\quad \hat{N}_{i+1} \leftarrow$ negatives from $\hat{P}_i$ by h_{i+1}
8: $\quad \hat{P}_{i+1} \leftarrow$ positives from $\hat{P}_i$ by h_{i+1}
9: $\quad i \leftarrow i + 1$
10: **end while**

MC, described in Algorithm 1, is thus composed of two stages: the mapping stage and the convergence stage. In the mapping stage, the algorithm uses OC-SVM (C_1) to compute an initial approximation of strong negatives in U. Based on this initial approximation, the convergence stage runs iteratively using a binary SVM (C_2) to maximize the margin in order to make a progressively better approximation of negative data. When no new negatives are discovered, MC converges and the boundary comes to a hold.

Figure 1, adopted from [13, 20], illustrates the MC process on a data set U composed of seven data clusters in 1-D, of which only the middle one is positive. Everything is unlabeled except for the dark subset of positives. The optimal boundary is represented by the dashed lines. OC-SVM would end up tightly fitting the positive data on (b_p, b'_p). MC starts with identifying strong negative examples, by employing a OC-SVM with a high threshold to favor higher recall and install the initial hypothesis (h_0, h'_0) far from the positive data. Subsequent convergence steps improve boundary (h_i, h'_i) toward the optimal one by adding unlabeled data recognized as negatives to N, then employing binary SVM and taking advantage of its margin maximization property, which avoids stopping in an arbitrary gap in the feature space.

3.3. $\hat{P}P$ measure and classifier selection

Cases when the positive class is severely undersampled or when too much unlabeled items act as noise would result in incapacity of detecting the gap between positive and negative data in the feature space, which makes MC over-iterate and overfit. An example is illustrated in [20], where a measure called $\hat{P}P$ is introduced and shown to be effective in detecting convergence and is hence employed as stopping criterion. Here, we describe this approach with a slight modification to incorporate a parameter to control the SVM decision threshold. This parameter regulates the size of rejected unlabeled data and hence the size of the resulting phrase table. Hereafter, a classification hypothesis h and a threshold α identify a classifier h_α, using the following decision function

$$f_{h_\alpha}(\boldsymbol{x}) = sgn(\delta_h(\boldsymbol{x}) - \alpha) \quad (2)$$

where $\delta_h(\boldsymbol{x})$[1] is the SVM decision value, on which α acts as a threshold and allows to shift the decision boundary in the feature space. The standard SVM decision function f_h is obtained at $\alpha = 0$.

Every such classifier h_α is represented as a point in the $\hat{P}P$ plot as shown in Figure 2. On the x-axis, we plot the percentage of the entire data set positively classified $|\hat{P}_\alpha|/|U|$, while on the y-axis, we plot the percentage of positive data positively classified $|h^+_\alpha(P) \cap P|/|U|$. In Figure 2, each dotted curve depicts the performance of different threshold values α for a certain classifier h_i, resulting from MC iterations. The standard curve (solid line) traces the performance of the standard classifier ($\alpha = 0$) for each MC iteration. It can be interpreted in a ROC-like fashion. The upper left corner represents a perfect classifier, while points on the diagonal are hypothesis performing random selection from U. The first point in the convergence will be close to the upper right corner: the mapping stage is about selecting a small part of the data set containing only near-certain negatives. Subsequent convergence steps will try to produce a smaller selection, moving left on the curve, while maintaining performance on the positive subset. Contrary to a genuine ROC curve, the $\hat{P}P$-curve is not continuous, and the step-like behavior of points (x_i, y_i) is not guaranteed, which makes it impossible to calculate the area under the curve (AUC). An alternative is to identify the point in the curve that discards most of the data in U, while keeping a large part of the positive data P. The point on the curve that is closest to (0, 100) is considered as the best classifier. To assign more importance to precision or recall, the distance measure can be weighted and/or different values of α can be used.

In every iteration of MC the resulting classifier scores are thresholded with several values of α and the corresponding $\hat{P}P$ points are calculated. Convergence is achieved when degradation or no more improvements of $\hat{P}P$ are observed.

4. Oracle Decoder for Building the Set of Positive Examples

The approach for supervised learning of phrase extraction introduced in Section 2.2 relies on a set of positive phrase pairs. In the PBSMT paradigm, good phrase pairs are required to fulfill two criteria: (1) participate in derivations of good translations with the highest BLEU scores (or another translation quality measure) with respect to some reference translation(s); and (2) have a good intrinsic quality as measured by the phrase-based model score.

This implies that, for identifying positive examples, we need to search among all possible translations, represented as a scored lattice, for the one that jointly optimizes the model score and the translation quality. Once the optimal path in the lattice is found we harvest all phrase pairs used in the derivation to be labeled as positive and added to P.

There are several methods to find the best path, of which we employ two in our experiments. The first method is *constrained decoding*, as implemented in MOSES[2]: the lattice is searched for the path with the highest model score that exactly matches the reference, and thus has a local BLEU score of one. However, if the reference is not attainable the sentence is discarded. The second method relaxes this constraint by using an oracle decoder that searches for the hypothesis that explicitly optimizes an approximation of the BLEU score at the sentence level as an objective. We implemented the lattice oracle decoder from [21], which, while being less conservative than constrained decoding (all source sentences are decoded), is agnostic about the model score which, therefore, needs to be optimized indirectly by pruning the lattice input of the decoder.

[1] $\delta_h(\boldsymbol{x}) = \boldsymbol{w}.\Phi(\boldsymbol{x}) - \rho$, where $\boldsymbol{w}$ is the classifier weight vector.

[2] `http://www.statmt.org/moses/`

5. Feature Functions

One of the main motivation of this work is to incorporate features into phrase pairs extraction, so as to smooth the conventional, alignment-based, phrase scores. We consider features from the literature [6, 16, 22, 4, 23], which evaluate various aspects of the association between a source and a target chunk. Most features are data-driven and language-independent, based on statistical word alignment and language models. A small set of language-dependent morpho-syntactic features is also used.

Weighted Alignment Matrix (WAM) feature is a score computed using discriminative Weighted Alignment Matrices [6] similar to the model-based phrase pair posterior metric described in [16]. Each cell in a weighted matrix [5] contains the posterior probability of aligning the corresponding source and target words. A phrase pair splits the underlying weighted matrix in two areas: *inside* and *outside* the phrase pair, where *consistent* and *inconsistent* links respectively live. The WAM feature is a score that combines two factors characterizing these areas and quantifies the consistency of the given phrase pair with respect to the entire probability distribution over all possible alignments.

Word Alignments (WA) induced features, similar to [16, 22], allow the evaluation of the quality of the association between source and target phrases according to the number of consistent and inconsistent word links. Given a standard alignment matrix obtained by thresholding the weighted matrix, and a phrase pair, a *consistent* link associates words inside the phrase pair boundary, while an *inconsistent* link crosses the phrase pair boundary. This feature is the ratio between the number of consistent links and the sum of the number of consistent and inconsistent links.

Bilingual and Monolingual Information (BI, MI) features are proposed in [16] as measures of the reliability of a phrase pair. Extracting candidate translations for every phrase to maximize coverage, is not always feasible and might hurt precision. Some phrases could not be accuratly aligned due to data sparsity and limitations of alignment models; while other phrases carry no linguistic meaning, such as phrases that are parts of non-compositional phrases or metaphorical expressions.

The BI feature addresses the first issue by estimating how reliably the model aligns a phrase pair. Given a weighted alignment matrix, we calculate the WAM score for all phrase pairs, and normalize them to estimate for every source phrase $(f_{i_1}^{i_2})$ a conditional distribution over all target phrases: $P_{\text{WAM}}(.|f_{i_1}^{i_2})$. The same computation is performed for every target phrase $(e_{i_1}^{i_2})$. The BI score of a source or a target phrase is defined as the entropy of the corresponding distribution. Low entropy implies a high confidence that the source/target phrase can be reliably aligned by the model. Conversely, high value of the entropy signals the impossibility to correctly identify the right alignment.

The MI feature addresses the second issue by evaluating to what extent a certain n-gram is a "good" phrase, and to measure how the choice of the phrase boundaries affects the quality of the phrase. The boundaries of a good phrase are assumed to be the right places to break. This feature evaluates the quality of the phrase pair boundaries using monolingual language models. Given a sentence of length N and a history of n words before a boundary (between words i and $i+1$), the forward language model probability $p(*|w_{i-(n-1)} \dots w_i)$ defines a conditional distribution. A similar distribution is defined for the "history" after the boundary, and, in this case, a backward language model is used. The predictive uncertainty (PU) of the boundary between word i and $i+1$ is computed

as the sum of the entropy of the forward and backward language models conditional distributions. The larger the predictive uncertainty is, the more likely is the boundary to be located in a "reasonable" place in the sentence. A good phrase pair is hence characterized with high PU values on its four boundaries, the product of which is the value of the MI feature. This feature captures how well a phrase pair combines with its neighbors to form new parallel sentences.

Statistical Significance (Pval) feature, as proposed in [4], captures the fact that not all phrase pairs are equally supported by the training data. By including corpus level statistics, this feature gives an overall view of the statistical properties of phrase pair. For a given phrase pair and a parallel corpus, we compute the source, target and joint source/target frequencies and draw a 2x2 contingency table representing the unconditional relationship between source and target phrases. We then calculate the one-tail p-value of the Fisher's Exact test, interpreted as the probability that the observed table or a more extreme one could occur by chance assuming a model of independence. We take $|log(\text{p-value})|$ as the value of this feature: that means that the higher it is, the more significant is the phrase pair.

Morphosyntactic Similarity (SIM) feature, unlike the previous ones, is language dependent and takes morphosyntactic information into account. This feature resembles the measure proposed in [23], which captures morphosyntactic Part-Of-Speech (POS) similarity between source and target phrases. We use here co-occurrence statistics of source/target POS tags, linked in the word aligned parallel corpus, to build a matching table similar to an IBM model's translation table, which provides association scores between source and target POS tags. The sum of these scores, for aligned words inside the phrase pair, normalized by the number of consistent links, is used as the value of the POS similarity feature. The higher this value is, the stronger the syntactic similarity of the given phrase pair.

Lexical Probability (LEX) features, as described in [3], and found in standard phrase tables, use word translation probabilities to quantify the extent to which words inside the phrase pair translate each others. These features are similar to POS similarity, but computed using surface word instead of POS tags.

6. Experiments

The experiments presented here aim to evaluate and compare the performance of different methods of training the translation model, including heuristics and the single-class classifier, first according to $\hat{P}P$ measure (Section 3.3), and second according to translation performance.

6.1. Data and experimental setup

For one-class SVM and the mapping convergence algorithm we use LIBSVM[3]. In our method we need to estimate a confidence measure in the classifier's output, for which we would ideally use a calibrated posterior probabilities. Although standard SVMs do not produce such probabilities, they can be estimated using a method proposed in [24] that fits a logistic function to the output of an SVM. This algorithm assumes equal distribution of positive and negative examples in training and test sets. This is not the case in a one-class setting, nor in the convergence steps where the distribution in the training set actually changes on each step and converges to the actual one. Therefore, we slightly altered the code of LIBSVM so that it directly outputs the distance to the decision hyperplane and used it as a confidence measure of prediction.

[3] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

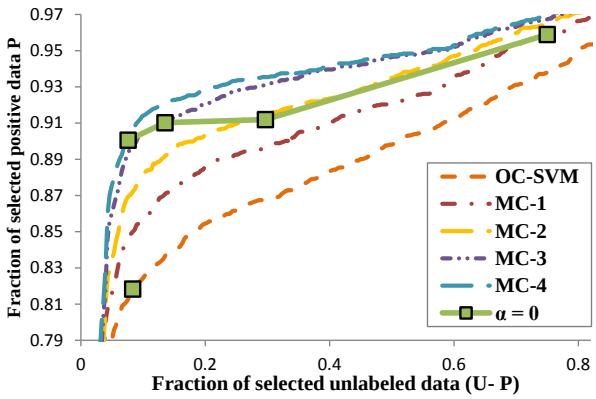

Figure 2: $\hat{P}P$ measure: OC-SVM and MC

For translation experiments we built several phrase-based, Arabic to English state-of-the-art SMT systems with Moses, of which the parameters are tuned with Minimum Error-Rate Training [9] on a held-out development corpus. For this corpus, we used the NIST MT'06 evaluation test set, containing 1,797 Arabic sentences (46K words) with four English references (53K words). The performance of each system is assessed by calculating the multi-reference BLEU on NIST MT'08 evaluation test set, which contains 1,360 Arabic sentences (43K words), each with four references (53K words). Training data for translation, reordering and language models, are subsets of the LDC resources made available by the NIST MT'09 constrained track[4]. For the language model, we train a 4-gram back-off model with SRILM[5] on the NIST MT'09 constrained English data. Arabic sentences are pre-processed using MADA+TOKAN[6] [25] and segmented according to the D2 tokenization scheme as an attempt to reduce the sparseness problem associated with the rich morphology of the Arabic language. The IBM Arabic-English Word Alignment Corpus [26] is used to train the discriminative alignment models. Part-of-speech tags for English are generated using the Stanford Tagger[7], while Yamcha[8] is used for Arabic.

In the experiments we randomly select 30K sentences from the NIST'09 training data as an input to the algorithm described in Section 2.2. U contains all possible phrase pairs with maximum phrase size of 3, for each of which we compute the set of features described in Section 5. We then use a phrase table built from U with both oracle decoders presented in Section 4 to decode a held-out parallel corpus of 2K sentences. Phrase pairs used by the decoder constitute positive examples, of which 80% are added to the training set P while the remaining are used for evaluation P_{test}.

6.2. Classification performance: $\hat{P}P$

We use positive examples in P to train a one-class SVM that is employed to score all phrase pairs in U, of which the worst scoring 10% examples are considered strong negatives, and used to boost the MC algorithm. The parameters ν and γ of OC-SVM and all classifiers trained in MC iterations are tuned using grid search and cross-validation to maximize the $\hat{P}P$ measure. The performance of

[4] http://www.itl.nist.gov/iad/mig/tests/mt/2009/
[5] http://www-speech.sri.com/projects/srilm/
[6] http://www1.ccls.columbia.edu/~cadim/MADA.html
[7] http://nlp.stanford.edu/software/tagger.shtml
[8] http://www.chasen.org/~taku/software/yamcha/

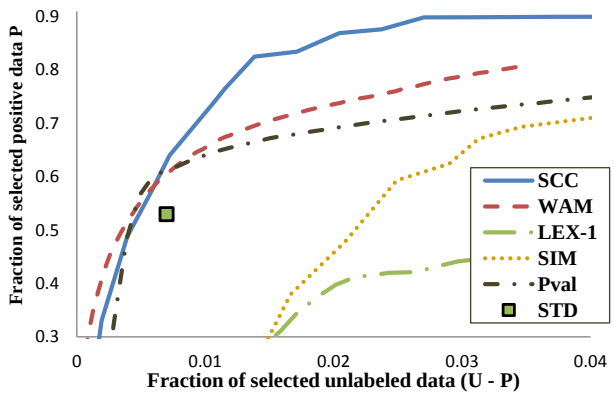

Figure 3: $\hat{P}P$ measure: SCC and some feature functions

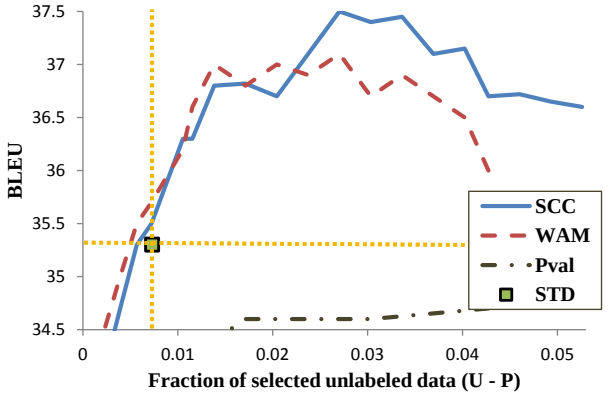

Figure 4: BLEU: SCC and some feature functions

the OC-SVM and the binary classifiers resulting from subsequent iterations of MC, are displayed in Figure 2. Each curve is obtained as described in Section 3.3, by quantizing the related classifier scores to divide the set $U - P$ into 300 quantiles of equal sizes and use the boundaries as different values for the threshold α. Each point on the plot reflects on the y-axis the percentage of selected positive phrase pairs from P_{test}, against the selected percentage of $U - P$ on the x-axis. The OC-SVM, depicted by the solid curve, achieves already a reasonable performance, identifying about 82% of positive examples while discarding about 90% of the rest. Better percentages are successfully achieved by subsequent iterations of MC, identifying 91% of positive examples and discarding 94% of the rest. The solid line connecting different points across curves plots the performance of the standard SVM classifiers at the threshold $\alpha = 0$.

Similarly to classifier scores, the different feature scores are quantized to obtain the curves depicted in Figure 3, where the curve corresponding to the best classifier is reproduced for comparison purpose. [9]

We note that the SCC classifier, which combines several features, achieves the best $\hat{P}P$ performance, improving on any feature acting solely.

6.3. Translation performance: BLEU

We study in this section the translation performance in BLEU for each phrase pair selection score. Similarly to the previous section, scores produced by the best classifier and different feature functions, are quantized into several quantiles per scoring method[10]. Each corresponding threshold α is used to construct a phrase table by retaining all phrase pairs having a higher score and estimating standard models described in [3]. After tuning the parameters of the translation systems[11] on the development corpus, BLEU is computed for each phrase table as the translation performance on the test corpus. Figure 4 [9] plots BLEU on the y-axis as a function of the percentage of retained phrase pairs, which corresponds to the size of the phrase table, from $U - P$, as on Figure 1.

Figure 4 shows that for any given threshold, extraction based on the weighted alignment matrix (WAM) feature achieves the best performance in BLEU, with improvements over the standard baseline that ranges from slight to significant with different sizes of the phrase table. The SCC classifier improves over WAM scores only for smaller values of α corresponding to larger phrase tables, while attaining comparable results for higher values of α (smaller phrase tables). Extraction based on scores by any other feature function, results in deterioration of performance. We note also that for SCC classifier and WAM perform better than the standard extraction for the same size of phrase table, and require fewer phrase pairs in order to obtain comparable performance, thus can be used for pruning large phrase tables.

We conducted an additional experiment where we incorporate the classifier score as an additional accuracy-based feature to the translation log-linear model and let MERT tune its weight. Figure 5 shows, for different phrase table sizes, slight improvements in BLEU scores for systems that use this feature over the baselines that do not. Nevertheless, this feature is effective only for larger, noisier phrase tables. Similar behaviors are observed when adding all the other feature functions described in Section 5 are incorporated simultaneously[12].

6.4. Discussion

We would like to further analyze the dynamics of the scores computed with different methods. We consider three methods: the single-class classifier, the weighted alignment matrix and the standard extraction heuristic. Figure 6 shows for each method and for a given source phrase, the score of all corresponding phrase pairs on the y-axis. The x-axis enumerate the target phrases in the order of the descendant score of the standard extraction heuristic. Figure 6 reveals that substituting the standard heuristic scores with the weighted alignment matrix scores and further with the classifier score, has two effects: (1) it modifies the score and the rank of some phrase pairs causing the extraction of previously missed ones; (2) it smooths the scores and allows increased control over the selection process using the threshold α.

We note that while all of these three scoring methods identify well most of the best phrase pairs and rank them high in the list, they differ in their ability to rank phrase pairs of worst quality. While scores based on the weighted alignment matrix may be sufficient to

[9] Missing feature curves from figures 3 and 4 show similar or worse performance than displayed features, and are omitted for clarity.

[10] Note that different dynamics of feature scores result in different number of quantiles per feature

[11] In total we have 19 systems for the SCC classifier, 16 for the WAM feature, and 5 for each of the remaining features

[12] We had to run MERT 3 times for each point and take the maximum BLEU score in this experiment since it was less stable when optimizing all the new features.

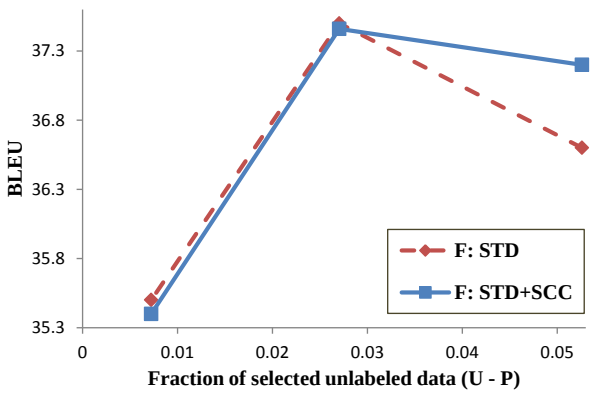

Figure 5: BLEU: SCC scores as a new feature

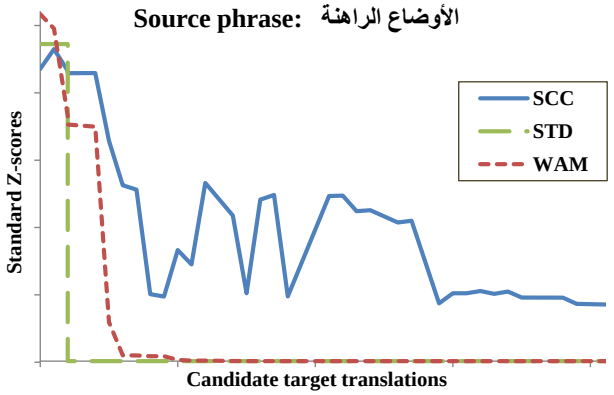

Figure 6: Z-scores: comparison of different scoring methods

construct high precision phrase tables with the best phrase pairs, recall oriented phrase tables require more sophisticated decision procedures to retrieve good translations in the large set of candidates that are difficult to distinguish and ignored by standard methods.

7. Related Work

In [16] phrase pairs extraction is formulated as an information retrieval problem, aiming to achieve a good precision/recall balance for the training corpus. They incorporate several feature functions into a log-linear model parametrized with $\{\lambda_k\}$, used to score phrase pairs and then apply a threshold τ for extraction. The set of parameters $\{\lambda_k, \tau\}$ are tuned to maximize a translation quality measure using the downhill simplex method. However, to obtain optimal solution, each optimization iteration that involves training a standard phrase table with parameters $\{\lambda_k, \tau\}$, should tune its weights with MERT, which is expensive and hence omitted in their experiments. A similar model is used in [22] to add features to extraction, without any parameter tuning. Our work is similar in respect of incorporating additional features to extraction, whereas our formulation of the problem in the supervised classification framework, unlike [16], allows much less expensive incorporation of the translation quality measure, which is ignored in [22].

In [27] the standard features in the log-linear translation model tuned with MERT is used to score phrase pairs already existing in the phrase table and employ a competitive linking algorithm to keep the best one-to-one phrase matching while discarding the rest. Contrarily tou our approach, feature weights selected by MERT, although directly optimizing translation quality, they are learned for a given phrase table and do not generalize to unseen phrase pairs.

In [28], the whole set of phrasal translation rules that should be extracted from a sentence-pair is predicted at once instead of predicting one phrase pair at a time. Word and phrase level features are incorporated into a discriminative model for extraction. Manually annotated word alignments are used to automatically obtain training extraction sets, whereas we use the oracle decoder.

Another related line of research is phrase table pruning, which is carried out by first assigning scores to phrase pairs, by ways of statistical significance tests [4, 17], or by computing the decoder usage statistics [29]. Our method takes advantage of different pruning criteria and integrates them into the filtering procedure.

The introduction of the translation process into the definition of useful phrase pairs has also been investigated in the literature. In [30] an oracle decoder is used to compute forced phrasal alignment, that are then used in a leaving-one-out smoothing technique, which results in a better estimation of translation probabilities. In [11] an oracle decoder is used to identify the best hypothesis in the n-best list output of the decoder and use an average edit-distance between phrase pairs occurring in the oracle hypothesis and other phrase pairs in other hypothesis in the n-best list, to compute a translation quality-based feature that is added to the phrase table. Unlike these methods, our procedure applies the oracle decoder to the problem of phrase pairs extraction and not only to estimating features of already extracted phrase pairs. Additionally, the proposed method estimates a similar quality-based feature but differs from this work as it uses the complete search space instead of a limited n-best list as input to the oracle decoder, and involves supervised learning of the additional scoring feature and hence generalizes better to unseen phrase pairs.

8. Conclusions and Future Work

In this paper we presented a novel translation quality informed procedure for both extraction and scoring of phrase pairs. The model at the center of our procedure combines arbitrary features to assess phrase quality. It is parametrized with a threshold that allows improved control over the size of the resulting phrase table, which is useful for fine tuning of the precision/recall balance. The proposed method helps exploring regions in the space of possible phrase pairs that are ignored by the standard extraction approach, which leads to improvements in BLEU scores for recall-oriented translation models. Additionally, we experimentally studied the effect on BLEU of adding new features to the phrase table and learning their weights with MERT, including a feature trained as a by-product of our procedure. This procedure can be viewed as a method to combine several criteria for filtering large phrase tables. We leave the verification of its effectiveness to future work.

9. Acknowledgments

This work was partly realized as part of the Quaero Program, funded by OSEO, the French agency for innovation.

10. References

[1] F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. A. Smith, K. Eng, V. Jain,

Z. Jin, and D. Radev, "A smorgasbord of features for statistical machine translation," in *Proc. of NAACL-HLT*, 2004.

[2] D. Chiang, K. Knight, and W. Wang, "11,001 new features for statistical machine translation," in *Proc. of NAACL-HLT*, 2009, pp. 218–226.

[3] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proc. NAACL-HLT*, 2003, pp. 48–54.

[4] H. Johnson, J. Martin, G. Foster, and R. Kuhn, "Improving translation quality by discarding most of the phrasetable," in *Proc. of EMNLP-CoNLL*, 2007, pp. 967–975.

[5] Y. Liu, T. Xia, X. Xiao, and Q. Liu, "Weighted alignment matrices for statistical machine translation," in *Proc. of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, 2009, pp. 1017–1026.

[6] N. Tomeh, A. Allauzen, and F. Yvon, "Discriminative weighted alignment matrices for statistical machine translation," in *Proc. of the EAMT*, 2011, pp. 305–312.

[7] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: parameter estimation," *Comput. Linguist.*, vol. 19, pp. 263–311, 1993.

[8] D. Marcu and W. Wong, "A phrase-based, joint probability model for statistical machine translation," in *Proc. of EMNLP*, 2002, pp. 133–139.

[9] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proc. of ACL*, 2003, pp. 160–167.

[10] D. Tax, "One-class classification," phd, Delft University of Technology, 2001.

[11] S. Penkale, Y. Ma, D. Galron, and A. Way, "Accuracy-based scoring for phrase-based statistical machine translation," in *Proc. of AMTA*, 2010, pp. 257–266.

[12] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, pp. 1443–1471, 2001.

[13] H. Yu, "Single-class classification with mapping convergence," *Mach. Learn.*, vol. 61, pp. 49–69, 2005.

[14] B. Raskutti and A. Kowalczyk, "Extreme re-balancing for SVMs: a case study," *SIGKDD Explor. Newsl.*, vol. 6, pp. 60–69, 2004.

[15] D. Galron, S. Penkale, A. Way, and I. D. Melamed, "Accuracy-based scoring for DOT: towards direct error minimization for data-oriented translation," in *Proc. of EMNLP-HLT*, 2009, pp. 371–380.

[16] Y. Deng, J. Xu, and Y. Gao, "Phrase table training for precision and recall: What makes a good phrase and a good phrase pair?" in *Proc. of ACL-HLT*, 2008, pp. 81–88.

[17] N. Tomeh, N. Cancedda, and M. Dymetman, "Complexity-based phrase-table filtering for statistical machine translation," in *MT Summit XII*, 2009, pp. 144–151.

[18] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural Comput.*, vol. 12, pp. 1207–1245, 2000.

[19] B. Zhang and W. Zuo, "Learning from positive and unlabeled examples: A survey," in *Proc. of International Symposiums on Information Processing*, 2008, pp. 650–654.

[20] M. Hovelynck and B. Chidlovskii, "Multi-modality in one-class classification," in *Proc. of WWW*, 2010, pp. 441–450.

[21] M. Dreyer, K. Hall, and S. Khudanpur, "Comparing reordering constraints for SMT using efficient BLEU oracle computation," in *Proc. of NAACL-HLT/AMTA Workshop on Syntax and Structure in Statistical Translation*, 2007, pp. 103–110.

[22] A. Venugopal, S. Vogel, and A. Waibel, "Effective phrase translation extraction from alignment models," in *Proc. of ACL*, 2003, pp. 319–326.

[23] M. Turchi and M. Ehrmann, "Knowledge expansion of a statistical machine translation system using morphological resources," in *Proc. of CICLing*, 2011, pp. 37–43.

[24] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large-Margin Classifiers*, 1999, pp. 61–74.

[25] N. Habash, "Arabic morphological representations for machine translation," in *Arabic Computational Morphology*, 2007, vol. 38, pp. 263–285.

[26] A. Ittycheriah, Y. Al-Onaizan, and S. Roukos, "The IBM Arabic-English Word Alignment Corpus," Tech. Rep., 2006.

[27] L. S. Zettlemoyer and R. C. Moore, "Selective phrase pair extraction for improved statistical machine translation," in *NAACL-HLT Short*, 2007, pp. 209–212.

[28] J. DeNero and D. Klein, "Discriminative modeling of extraction sets for machine translation," in *Proc. of ACL-HLT*, 2010, pp. 1453–1463.

[29] M. Eck, S. Vogel, and A. Waibel, "Translation model pruning via usage statistics for statistical machine translation," in *Proc. of NAACL-HLT Short*, 2007, pp. 21–24.

[30] J. Wuebker, A. Mauser, and H. Ney, "Training phrase translation models with leaving-one-out," in *Proc. of ACL*, 2010, pp. 475–484.