PEXACC: A Parallel Sentence Mining Algorithm from Comparable Corpora

Radu Ion
Research Institute for Artificial Intelligence, Romanian Academy
13, Calea 13 Septembrie, Bucharest 050711, Romania
radu@racai.ro

Abstract

Extracting parallel data from comparable corpora in order to enrich existing statistical translation models is an avenue that attracted a lot of research in recent years. There are experiments that convincingly show how parallel data extracted from comparable corpora is able to improve statistical machine translation. Yet, the existing body of research on parallel sentence mining from comparable corpora does not take into account the degree of comparability of the corpus being processed or the computation time it takes to extract parallel sentences from a corpus of a given size. We will show that the performance of a parallel sentence extractor crucially depends on the degree of comparability such that it is more difficult to process a weakly comparable corpus than a strongly comparable corpus. In this paper we describe PEXACC, a distributed (running on multiple CPUs), trainable parallel sentence/phrase extractor from comparable corpora. PEXACC is freely available for download with the ACCURAT Toolkit, a collection of MT-related tools developed in the ACCURAT project.

Keywords: parallel sentence extraction from comparable corpora, comparable corpora, machine translation, translation models

1. Introduction

When it comes to statistical machine translation (SMT) there is a growing need of training parallel data, especially for language pairs that are not well represented in the realm of Word Wide Web (WWW). Parallel corpora (PC) acquisition from WWW has been traditionally geared towards identifying similar structure (searching for anchors in titles, sections, images with identical descriptions, same inbound/outbound links, etc.) of the parallel documents and/or their referring URLs (Zhang et al., 2006) but recent research has been oriented towards scoring the candidate target document as to how well it (dictionary-) translates the source counterpart (Tsvetkov & Wintner, 2010).

In contrast, comparable corpora (CC) are easier to collect than PC basically because no parallel web sites are needed to be identified a priori and no (usually complicated) HTML parsing is required in order to identify the parallel parts at crawling time. CC is excepted from the tedious task of particular HTML parsing by assuming that documents are related in some (explicitly stated) way and thus, just collecting all the text from the HTML document is sufficient for the crawler. We accept the definition of CC from Munteanu & Marcu (2005): a pair of comparable documents "*... while not parallel in the strict sense, are somewhat related and convey overlapping information*". The relatedness of a pair of comparable documents has been defined and experimented with in many ways among which the membership to the same domain/topic/genre, the same publication date/period (especially for News documents), the appearance of the same named entities (names of persons, geographical entities, numeric entities, dates, times, etc.) and so on. The predominant paradigm for acquiring CC is the cross-language information retrieval method based on seed lists of source documents URLs (Talvensaari et al., 2008).

The direct consequence of the nature of CC and its collecting mechanism is the large size compared to (truly) PC collected from the WWW. The difference is several orders of magnitude and the size of the CC prompts for dealing with computational challenges that are not encountered when searching for parallel sentences in PC. Specifically, we are referring to the positional information of the translation units in parallel texts (paragraphs and sentences) which constitute a natural pruning technique when searching for parallel sentences in PC. That is, for the i-th sentence of the source document, the sought aligned sentence is to be found in a window of $\pm k$ sentences around the j-th sentence of the target document where i and j are proportional. This is not true in the case of CC. Furthermore, if we have M documents in the source language and N documents in the target language, a parallel sentence mining algorithm should look in every document pair from the set containing MN pairs in order to achieve the maximum recall. But when M and/or N are/is large, this is not feasible and one has to pre-align the documents in CC as to the likelihood to contain parallel data.

PEXACC is a computationally parallel algorithm that aims at finding parallel (translation equivalent) pieces of text in CC at two levels of granularity: sentence and phrase. We show that this algorithm is capable of top performances and that it is applicable to large amounts of CC provided that the required computational resources are available. Furthermore, its implementation is **freely available** within the ACCURAT Toolkit for extracting MT resources from CC (Ion et al., 2011a).

2. Related work

Extracting parallel data from comparable corpora in order to enrich existing statistical translation models is an avenue that attracted a fair amount of research in recent years. Generally speaking, we identified two approaches to parallel sentence mining from CC in the existing literature on the subject:

1. pair documents in CC (by using cross-language

information retrieval techniques or translation similarity measures); for each document pair generate the cartesian product of the source and target sentence sets; score (by classification or a type of translation similarity measure) each sentence pair from the cartesian product as to its parallelism degree;

2. use an initial SMT system (trained on existing parallel/comparable data) to translate every source sentence from CC into the target language; use standard information retrieval techniques to find the target sentences most similar to the translation and thus parallel to the initial source sentence.

The main challenges in these two approaches are **the document pairing in CC** and the parallelism degree scoring function of a sentence pair which we will call **the translation similarity measure**.

Munteanu & Marcu (2005) (followed in spirit by Tillman (2009) and Quirk et al. (2007)) solved the document pairing problem by generating target language queries from a source document by using the first 5 most probable translations of each word, and then interrogating a standard IR engine to find the most similar 20 target documents for the source document. Ion (2011b) devised an EM algorithm to reveal the hidden document alignments in a CC on the assumption that there are certain word translation pairs that are very good indicators for these alignments. Finally, Fung & Cheung (2004) use a word translation similarity measure to discover similar documents.

Given a pair of sentences, the first in the source language and the last in the target language, the job of the translation similarity measure is to assess "how parallel" the two sentences are. Munteanu & Marcu (2005) (followed by Tillman (2009) with an improved variation) devised a maximum entropy classifier that will assign the label of "parallel" or "not parallel" to the pair of sentences. Among the features involved we can mention the following: fertility (Brown et al., 1998), contiguous connected spans of words, sentence lengths (with lengths difference and lengths ratio), etc. Rauf & Schwenk (2011) (and independently Thi Ngoc Diep et al. (2010)) use a previously trained SMT system to translate the source sentence into the target language and then apply MT assessment measures such as WER (the Levenshtein distance), TER (Snover et al. 2006) and TERp (Snover et al. 2009) in order to monolingually see how similar is the translation of the source sentence with the target sentence. A radical different approach to determine if the source sentence is similar to the target sentence is given by the generative models of Quirk et al. (2007). They assume that the target sentence (or phrase) is conditionally generated by the source sentence and proceed to model this generation probability. The parameters of the model are the source and target words and the positions of the source words that generated the translated target words.

3. PEXACC

PEXACC is a "Parallel phrase EXtractor from Comparable Corpora" that belongs to the category of extractors of type 1 defined in the previous section. That is, it requires that documents in CC are aligned and in order to assign a parallelism score to a pair of sentences, it implements a trainable, language independent translation similarity measure which is a weighted sum of translation features. The next section describes the PEXACC translation similarity measure at length including the weighted sum equation, the features descriptions and the training of the weights.

The general workflow of PEXACC is as follows (given a pair of source and target documents):

1. split the input source and target documents into sentences and then, if desired, into smaller parts (loosely called 'phrases' throughout this presentation) according to a list of language dependent markers. By a "marker" we understand a specific functional word that usually indicates the beginning of a syntactic constituent or a clause. For English these markers include: prepositions, particles and negations (the infinitive 'to', 'not'), auxiliary and modal verbs ('have', 'be', 'can', 'must'), interrogative and relative pronouns, determiners and adverbs ('which', 'what', 'who', 'that', 'how', 'when', 'where', etc.) and subordinating conjunctions ('that', 'as', 'after', 'although', 'because', 'before', etc.). An important design decision is choosing a set of markers such that, for the source and the target languages, the phrases we obtain by splitting are in a 1:1 correspondence as much as possible. Thus, for Romanian, the same types of markers can be considered and, in most of the cases, the phrases would align 1:1. In the next example pair of parallel sentences (the markers are underlined, square brackets indicate the phrases)

   en: **[**A simple example**]** **[**will demonstrate the splitting**]** **[**of this sentence**]** **[**into smaller parts**]**.

   ro: [Un exemplu elementar] [va demonstra împărţirea acestei propoziţii] [în părţi mai mici].

   we have the following correspondences: "**[**A simple example**]** ⇔ **[**Un exemplu elementar**]**" (1:1 correspondence), "**[**will demonstrate the splitting**]** **[**of this sentence**]** ⇔ **[**va demonstra împărţirea acestei propoziţii**]**" (2:1) and "**[**into smaller parts**]** ⇔ **[**în părţi mai mici**]**" (1:1).

2. score each possible pair of sentences/phrases as to their parallelism degree by using equation 1 (see the next section);
3. output all pairs of sentences/phrases for which equation 1 gives a score larger than a predefined threshold (default to 0.2 but the real parallelism threshold is dependent on the type of the corpus: parallel, strongly comparable and weakly comparable and on the values of the weights).

The computations in the second step of PEXACC are independent one of the other and as such, may be executed in parallel. In its current implementation, PEXACC is able to spread the computation of the translation similarity measure for different sentence pairs over multiple CPUs in order to considerably shorten the overall computation time.

Equation 1 makes use of several feature functions that are designed to indicate the parallelism of two sentences/phrases s and t. These functions are designed to return 1 when s and t are perfectly parallel (i.e. t has been obtained from s by translation if s and t were to be presented together as a pair to a human judge). The functions should return a value close to 0 when s and t are not related at all but this behavior is critically influenced by the quality and the completeness of the dictionary that is used (see the next section for details). Thus, s and t may still be parallel but if individual words in s do not have the relevant t translations in the dictionary and/or the translations probabilities are small, the resulting (low) score could be misleading. This is the main reason for which we have incorporated a "**relevance feedback loop**" (idea from Fung & Cheung (2004)). Thus, steps 2—4 of the algorithm are executed for a fixed number of times and

4. takes the output of step 3 and trains a supplementary GIZA++ (Gao & Vogel, 2008) dictionary on all sentence/phrase pairs with a certain parallelism score (to minimize noise) and adds it to the main initial dictionary. The combination method between the main dictionary *D* and the learnt one *T* is as follows:
   - if the pair of the word translation equivalents e is found in both dictionaries, its new translation probability $p(e)$ becomes $p(e) = 0.7p_D(e) + 0.3p_T(e)$ where $p_D(e)$ is the probability of e in the *D* dictionary and $p_T(e)$ is the probability of e in the *T* dictionary;
   - if the pair of translation equivalents e is found in either *D* or *T* but not both, leave its probability unchanged.

4. PEXACC translation similarity measure

We have modeled the translation similarity measure as a weighted sum of feature functions that indicate if the source sentence/phrase is translated by the target sentence/phrase. Given two sentences s in the source language and t in the target language, then the translation similarity measure $P(s,t)$ is

$$P(s,t) = \sum_i \theta_i f_i(s,t) \qquad (1)$$

such that $\sum_i \theta_i = 1$. Each feature function $f_i(s,t)$ will return a real value between 0 (s and t are not related at all) and 1 (t is a translation of s) and contributes to the overall parallelism score with a specific fraction θ_i that is language-pair dependent and that will be automatically determined by training a logistic regression classifier on existing parallel data (to be described in subsection 4.2).

We consider that this approach is better suited to discovery of parallel and, especially, quasi-parallel sentences than the binary classification approach of Munteanu & Marcu (2005) because of the fact that pairs of sentences extracted from comparable corpora may fall into the "strongly comparable" intermediate class which does not exist with the "parallel"/"not parallel" binary distinction. Furthermore, we postulate that one cannot simply extend the number of classes of the classifier because the levels of comparability of two sentences correspond to a continuous, real-valued function with values ranging from e.g. "not parallel and not related at all" (0) to "parallel" (1).

Each of the feature functions $f_i(s,t)$ has been designed to return a value close to 1 on parallel s and t by manually inspecting a fair amount of parallel examples in the English-Romanian pair of languages. By negation, we assume that the same feature functions will return a value close to 0 for non-parallel, not-related s and t but this behavior is critically influenced by the quality and completeness of the linguistic computational resources that we use: bilingual translation lexicons, lists of inflectional suffixes used for stemming and lists of stop-words. Thus, generally, a feature function that uses one (or more) of the resources mentioned above can falsely return a value close to 0 for parallel s and t due to the fact that this decision was made in the absence of the relevant entries in that resource. The prototypical example here is that the translation lexicon does not contain the relevant translations for the words in s.

Equation 1 will return a score based on the translation similarity of the sentences but, in practice, we also apply a length ratio filter (as Munteanu & Marcu (2005)) in order to reject pairs of s and t whose lengths are disproportionate.

Thus, if $\frac{\max(|s|,|t|)}{\min(|s|,|t|)} > T$, where $|s|$ is the length of s in words and T is a language pair dependent ratio (e.g. 1.5 for English-Romanian), we assign the score of 0 to the pair (not related at all).

4.1. Features

Before being passed to the feature functions, sentences s and t are tokenized, functional words are identified and content word are stemmed using language-dependent inflectional suffixes. Given these transformations of s and t, all features $f_i(s,t)$ are language-independent. We use 5 features.

$f_1(s,t)$ is the "**content words translation strength**" feature. Given a statistical translation lexicon obtained by e.g. applying GIZA++ on a parallel corpus[1], we find the best 1:1 alignment A between content words in s and t such that the translation probability[2] is maximized. If $\langle cw_i^s, cw_j^t\rangle$ is a word pair from A, $p(\langle cw_i^s, cw_j^t\rangle)$ is the translation probability of the word pair from the dictionary and $|s|$ is the length (in content words) of sentence s, then

$$f_1(s,t) = \frac{\sum_{\langle cw_i^s, cw_j^t\rangle \in A} p(\langle cw_i^s, cw_j^t\rangle)}{|s|} \qquad (2)$$

This feature has a maximum value of 1 if all content words from s are translated in t with the maximum probability of 1.

$f_2(s,t)$ is the "**functional words translation strength**" feature. The intuition is that functional words around content words aligned as in feature $f_1(s,t)$, will also align for parallel s and t because of the fact that, from a dependency-syntactic point of view, functional words (prepositions, determiners, articles, particles, etc.) are usually governed by or govern nearby content words. Mathematically, if $\langle fw_k^s, fw_l^t\rangle$ is the highest scored pair of aligned functional words near (in a window of ±3 words) the aligned pair of content words $\langle cw_i^s, cw_j^t\rangle$ from A, $|A|$ is the cardinal of the best alignment as found by $f_1(s,t)$ and $p(\langle fw_k^s, fw_l^t\rangle)$ is the probability of the functional word pair from the dictionary, then

$$f_2(s,t) = \frac{\sum_{\langle cw_i^s, cw_j^t\rangle \in A} p(\langle fw_k^s, fw_l^t\rangle)}{|A|} \qquad (3)$$

The maximal value of $f_2(s,t)$ is 1 and it is reached when for each aligned pair of content words from A, there is a pair of functional words that align with the maximum probability of 1.

$f_3(s,t)$ is the "**alignment obliqueness**" feature (Tufiş et al., 2006). Here we have redefined it to be a discounted correlation measure because there are pairs of languages for which the natural word order implies crossing word alignment links. $f_3(s,t)$ also uses the alignment set A of content words described for feature $f_1(s,t)$ from which we derive two source and target vectors x^s and x^t of the same length containing the indices i in the ascending order $(1 \le i \le |s|)$ and j respectively $(1 \le j \le |t|)$ of content words cw_i^s and cw_j^t that form an alignment pair in A. Alignment obliqueness is computed as

$$f_3(s,t) = \mathrm{abs}(\rho_{x^s,x^t}) \frac{1}{1 + e^{-10\frac{|A|}{\min(|s|,|t|)}+5}} \qquad (4)$$

where ρ_{x^s,x^t} is the Pearson correlation coefficient of the x^s and x^t vectors and $\mathrm{abs}(x)$ is the absolute value function. The second term is a modified sigmoid function

$$f(x) = \frac{1}{1 + e^{-10x+5}}$$

designed to be a discount factor with values between 0 and 1 when x takes on values between 0 and 1. The idea here is that if A contains a few alignments relative to $\min(|s|,|t|)$ (the size of A is at most $\min(|s|,|t|)$), then even if ρ_{x^s,x^t} is high, $f_3(s,t)$ should be small because a few alignments usually do not indicate parallelism.

$f_4(s,t)$ is the "**strong translation sentinels**" feature. Intuitively, if sentences s and t are parallel then, frequently, one can find content words that align near the beginning and end of the considered sentences. $f_4(s,t)$ is a binary-valued feature which is 1 if we can find "strong" translation pairs (probability greater than 0.2; set experimentally) between the first 2 content words at the beginning of s and t and between the last 2 content words at the end of s and t. $f_4(s,t)$ is 0 in the opposite case.

Finally, $f_5(s,t)$ is the "**end with the same punctuation**" feature. This is also a binary-valued feature which is 1 if both s and t end with the same type of punctuation: period, exclamation mark, etc. It is also 1 if both s and t lack final punctuation. $f_5(s,t)$ is 0 in the opposite case.

The observant reader has noticed by now that all the features with the exception of $f_5(s,t)$ are not symmetrical (because of the fact that they are all based on the word alignment computed in $f_1(s,t)$ which is itself not symmetrical due to the translation lexicon) and as such, the measure from equation 1 is not symmetrical as well. In order to have evidence from both directions, we will use the arithmetic mean to get the final measure:

$$M(s,t) = M(t,s) = \frac{P(s,t) + P(t,s)}{2} \qquad (5)$$

4.2. Learning the optimal weights

The weights θ_2 and θ_3 corresponding to the features "functional words translation strength" and "alignment obliqueness" are language-pair dependent because of the

[1] To obtain the dictionaries mentioned throughout this section, we have applied GIZA++ on the JRC Acquis corpus (http://langtech.jrc.it/JRC-Acquis.html).

[2] For two source and target words, if the pair is not in the dictionary, we use a 0 to 1 normalized version of the Levenshtein distance in order to assign a "translation probability" based on string similarity alone. If the source and target words are similar above a certain threshold (experimentally set to 0.7), we consider them to be translations.

specific word ordering of the source and target languages. At the same time, θ_1 through θ_4 have to be optimized with respect to the translation lexicon in use because of the fact that the construction of the word alignments is based on this dictionary. Last but not least, since $P(s,t)$ is not symmetrical, we will have to learn different θ_i weights from source to target and vice versa.

In order to derive a set of optimal weights for each language pair and translation lexicon, we have trained a **standard logistic regression classifier**. Briefly, the logistic regression classifier learns the θ_i weights that define the hyperplane (whose equation is the same as equation 1) that best separates the positive training examples from the negative ones. In our case, the examples are the multidimensional points whose coordinates are given by the feature functions $f_i(s,t)$.

For each language pair, the training set consists of 9500 parallel sentences[3] for the positive examples and 9500 of non-parallel sentences (obtained from the parallel pairs by random shuffling) for the negative examples. For the training set in question, we also have 500 additional parallel sentences together with 500 non-parallel sentences (obtained by random shuffling as well) as the test set. An example[4] is obtained by computing all the feature functions $f_i(s,t)$ for the given positive (parallel) or negative (non-parallel) s and t.

Table 1 summarizes the derived optimal weights for considered language-pairs (both directions).

Lang.	$\boldsymbol{\theta_1}$	$\boldsymbol{\theta_2}$	$\boldsymbol{\theta_3}$	$\boldsymbol{\theta_4}$	$\boldsymbol{\theta_5}$	**F1/BL**
en-ro	0.31	0.02	0.37	0.21	0.09	0.93/0.88
ro-en	0.31	0.01	0.37	0.20	0.11	0.93/0.91
en-de	0.31	0.02	0.3	0.17	0.2	0.94/0.89
de-en	0.35	0.02	0.28	0.16	0.19	0.96/0.92
en-sl	0.23	0.01	0.38	0.2	0.18	0.96/0.89
sl-en	0.2	0.03	0.38	0.19	0.2	0.94/0.89
en-el	0.61	0.08	0.21	0	0.1	0.99/0.98
el-en	0.47	0.08	0.28	0.07	0.1	0.98/0.98
en-lv	0.27	0.05	0.41	0.16	0.1	0.98/0.96
lv-en	0.49	0.03	0.41	0	0.07	0.99/0.96
en-lt	0.33	0.01	0.41	0.15	0.1	0.96/0.91
lt-en	0.28	0.01	0.41	0.15	0.15	0.94/0.90
en-et	0.28	0.08	0.36	0.17	0.11	0.98/0.96
et-en	0.27	0.07	0.38	0.18	0.1	0.96/0.93
en-hr	0.29	0.01	0.41	0.16	0.13	0.98/0.95
hr-en	0.25	0.02	0.44	0.17	0.12	0.98/0.97

Table 1: Optimal weights for the translation similarity measure

The language pairs for which we trained the optimal weighs (for both directions) are: English-Romanian (en-ro), English-German (en-de), English-Slovene (en-sl), English-Greek (en-el), English-Latvian (en-lv), English-Lithuanian (en-lt), English-Estonian (en-et) and English-Croatian (en-hr). The column named "F1/BL" in Table 1 indicates the gain in F1 measure when testing the translation similarity measure with the optimal weights on the test set as compared to a baseline (BL) consisting of applying the measure using fixed values of the weights corresponding to our intuition of their importance[5]: $\theta_1 = 0.45, \theta_2 = 0.2, \theta_3 = 0.15, \theta_4 = 0.15, \theta_5 = 0.05$.

5. Experiments and results

We tested PEXACC in several scenarios:

1. in order to measure the precision, recall and F1 measure on different types of comparable corpora, we artificially inserted noise (unrelated sentences) into a parallel corpus in specified proportions and checked the ability of PEXACC to re-discover the parallel corpus in the presence of noise; subsection 5.1 gives the details;
2. to compare PEXACC to current state of the art parallel sentence mining from CC, we implemented Munteanu & Marcu (2005) maximum entropy classifier (MaxEntClass) for English-German (Ion, 2011c) and ran the two algorithms on the artificially-created CC described above; we show that the task of extracting parallel sentences from CC is progressively more difficult as the CC type varies from strongly comparable to weakly comparable; subsection 5.2 describes the experiment;
3. in order to see how PEXACC behaves on real-world data, we have run it on a real English-Romanian CC News corpus collected in the ACCURAT project; subsection 5.3 presents the results.

5.1. Computing P, R and F1

To be able to compute recall, we needed to know exactly how many parallel sentence pairs are present in the test CC. Since the collected CC are usually very large and cannot be evaluated by hand, we decided to "pollute" an existing parallel corpus with "noisy sentences" – sentences that are drawn from the same domain but are unrelated. The proportion of noisy sentences vs. parallel sentences was controlled and was set to 2:1, 5:1 and 10:1. That is, for a noise proportion of 2:1, for each pair of parallel sentences, two pairs of noisy sentences were added.

We performed the tests on English-German (en-de), English-Romanian (en-ro), English-Greek (en-el) and English-Latvian (en-lv) to have a diverse language representation. The parallel test corpus for each language pair is a News corpus containing **100 parallel sentence pairs**. There is a source file containing 100 sentences (one sentence per line) and a target file containing the parallel

[3] Mostly from the News domain for all language pairs.

[4] It may be the case that an example occurs multiple times with both labels. In this case we retain all the occurrences of the example with the most frequent label and remove all the conflicting occurrences of that example.

[5] We used these values of the weights when we developed the features.

counterparts on the same line numbers. After source and target noise sentences are added to each file (in the specified proportion), the ordering of the sentences is destroyed by random shuffling.

A first interesting experiment is to judge the performance of PEXACC with the optimized weights vs. the default values of the weights (see subsection 4.2). We ran PEXACC with the default values and with the optimized values for the weights on the English-German CC, 2:1 noise ratio. Figures 1 and 2 plot the precision (P), recall (R), F1 measure (F1) and F0.2 measure for the two runs, in percents, over the values of the translation similarity measure (step 0.01) as computed by equation 1.

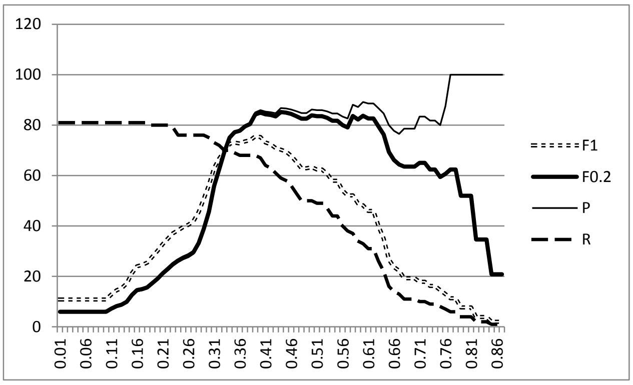

Figure 1: P, R, F1 and F0.2 of PEXACC running with default weights

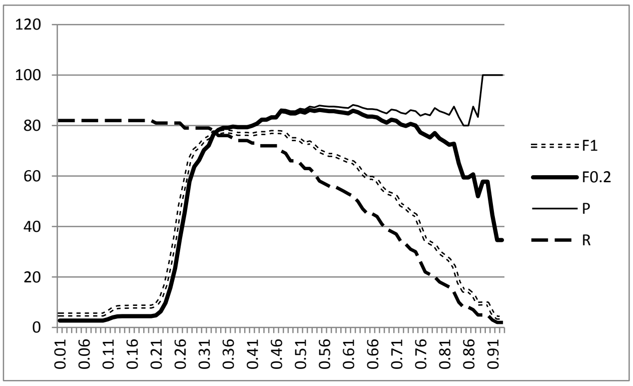

Figure 2: P, R, F1 and F0.2 of PEXACC running with the optimized weights

The F0.2 measure is computed as

$$F_\beta = (1+\beta^2)\frac{PR}{\beta^2 P + R}$$

where $\beta = 0.2$. F0.2 weighs precision more than recall. We have plotted F0.2 because we are more interested in precision than recall, PEXACC main usage being for SMT training.

When running with the default weights, the best F1 measure is 75.55% and the best F0.2 measure is 85.22%. With the optimized weights, the best F1 is 77.55% (+2%) and the best F0.2 is 86.21% (+1%). Comparatively studying Figures 1 and 2, the area of the graphic delimited by the F1 and F0.2 curves is significantly larger in the case of the optimized weights run. This translates directly into a better behavior of P (rapid increase) and R (slower decrease) across the range of the translation similarity measure values.

Tables 2 to 4 present the performance of PEXACC exhibiting the best F1/F0.2 measures as a function of the equation 1 translation similarity values. That is, all the sentence pairs that are discovered by PEXACC are sorted in the decreasing order of the translation similarity measure and, we compute P, R, F1 and F0.2 for the top produced results up to the considered threshold (varied from 0 to 1 in steps of 0.01; see Figure 2 for a graphical representation). We present the best values for F1 and F0.2

	P	**R**	**F1**	**P**	**R**	**F0.2**
en-de	0.791	0.76	0.775	0.878	0.58	0.861
en-ro	0.684	0.78	0.728	1	0.38	0.94
en-el	0.864	0.83	0.846	0.971	0.67	0.954
en-lv	0.916	0.77	0.836	0.985	0.68	0.968

Table 2: PEXACC run with optimized weights on the 2:1 noise ratio test CC

	P	**R**	**F1**	**P**	**R**	**F0.2**
en-de	0.76	0.7	0.729	0.862	0.5	0.838
en-ro	0.819	0.59	0.686	1	0.35	0.933
en-el	0.896	0.78	0.834	0.971	0.67	0.954
en-lv	0.88	0.74	0.804	0.947	0.54	0.92

Table 3: PEXACC run with optimized weights on the 5:1 noise ratio test CC

	P	**R**	**F1**	**P**	**R**	**F0.2**
en-de	0.724	0.63	0.673	0.838	0.52	0.819
en-ro	0.814	0.44	0.571	0.916	0.33	0.858
en-el	0.789	0.75	0.769	0.944	0.51	0.914
en-lv	0.774	0.72	0.746	0.973	0.37	0.916

Table 4: PEXACC run with optimized weights on the 10:1 noise ratio test CC

Looking at the tables above, we can see that that the task of finding parallel sentences becomes harder with the increasing noise level in the CC. For instance, the English-Romanian F1 measure drops with 15.7% when processing CC with noise ratio 10:1 compared to CC with noise ratio of 2:1. Comparatively, the English-Greek F1 measure drops with only 7.7% but the lower difference is explained by the fact that the noise was automatically introduced and thus, the length ratio of the noisy sentence pairs has not been checked. As a consequence, in the case of English-Greek, some of the wrong pairs were filtered more easily.

The current implementation of PEXACC is in C# on .NET Framework 4.0. The processing time is dependent on the language pair because the translation similarity measure from equation 1 computes, for each word, a type of transliteration in order to be able to compare and detect similar words in the source and target languages:

- for all languages all diacritics are replaced with their diacritical mark free form;
- for Greek, a full transliteration is applied to get to the Latin alphabet.

As a consequence, the number of processed sentence pairs varies with the language pair (as measured on a single core of an Intel i7 980 @ 3.33GHz, 16 GB DDR3 @ 800MHz):

- English-Romanian: 450 sentence pairs per second;
- English-German: 500 sentence pairs per second;
- English-Greek: 200 sentence pairs per second;
- English-Latvian: 540 sentence pairs per second.

As we already stated, PEXACC is able to distribute the computation on available CPU or CPU cores and as such, it is able to sustain the same sentence pair processing rate on each of the 12 cores of the Intel i7 980 @ 3.33GHz CPU on which we ran it. Thus, for instance, it finished (minus the time it needed to load the resources) the English-German processing of the 2:1 noise ratio text CC in approximately 15 seconds: 90,000 sentence pairs to process, 500 sentence pairs per second running on a single core, 12 CPU cores. This means it is able to process 500 * 12 = 6,000 sentence pairs per second on 12 cores which gives 90,000 / 6,000 = 15 seconds of processing time in total.

5.2. Comparison with the state of the art

In order to compare PEXACC with MaxEntClass, we ran both of them on the English-German CC constructed with the methodology described in the previously. For PEXACC, we give precision (P), recall (R) and F1 measure (F1) corresponding to the best F1 measure computed as a function of the translation similarity score from equation 1. In the case of MaxEntClass, we considered all the output that was produced.

	2:1		5:1		10:1	
	MEC	PXC	MEC	PXC	MEC	PXC
P	0.800	0.791	0.789	0.760	0.523	0.724
R	0.560	0.760	0.450	0.700	0.450	0.630
F1	0.658	0.775	0.573	0.729	0.483	0.673

Table 5: Comparison between the MaxEntClass (MEC) and PEXACC (PXC) when applied on different types of English-German CC

We can see that the performance of both algorithms decreases significantly when asked to extract parallel sentences from weakly comparable corpora (noise ratio 10:1) when compared to strongly comparable corpora (noise ratio 2:1).

PEXACC has the advantage that its output can be trimmed by imposing a certain threshold on the value of the translation similarity measure as computed by equation 1. Thus, above a certain threshold, the precision of PEXACC can even be as high as 1 at a significant cost of the recall (see Figure 2). We consider this to be an asset of PEXACC since, as already stated, its main use is to generate parallel data for SMT training. Furthermore, by carefully choosing a value of this threshold, the desired tradeoff between precision and recall (e.g. measuring F0.2) may be achieved.

5.3. Running PEXACC on real-world data

PEXACC has been used to collect parallel sentence pairs for SMT training in the ACCURAT project[6]. We present its evaluated precision on an English-Romanian weakly comparable news corpus collected in the project (version 14-02-2012) by continuously harvesting news articles from selected URLs. The documents in the corpus are aligned based on their titles and publication dates. Table 6 presents the corpus statistics.

	Docs.	Sentences	Tokens	Size
en	17,845	464,961	9,309,338	53.7MB
ro	7,120	121,104	2,605,976	16.9MB

Table 6: 14-02-2012 News CC statistics

We ran PEXACC on this corpus and kept all the sentence pairs with a translation similarity measure of at least 0.3 which are 22,352. We then sorted these pairs in **descending order** and inspected them by hand from the pair with the largest score (0.99) until the last pair with score larger than or equal to 0.5. Table 7 presents the results.

Threshold	Precision	Sentence pairs
0.9	1	22
0.8	1	166
0.7	0.99	973
0.6	0.95	3,267
0.5	0.92	7,186

Table 7: Accuracy of PEXACC on the first 7,186 sentence pairs extracted from the 14-02-2012 News CC

In Table 7, the "Sentence pairs" column shows the number of sentence pairs that have a translation similarity score of at least the specified value in the "Threshold" column. We cannot compute the recall because we do not know how may parallel sentence pairs are in the corpus. We have inspected all produced pairs with a score of at least 0.7. Below this threshold, we did a random sampling of the results by selecting and evaluating 100 pairs from each threshold range (0.7-0.6 and 0.6-0.5).

Table 7 shows that PEXACC precision is consistent with the results reported in Tables 2 to 4.

6. Conclusion

Parallel sentence mining from comparable corpora is a proven alternative to producing parallel data for SMT training for under-resourced languages. We have

[6] http://www.accurat-project.eu/

presented PEXACC, a trainable, extensible and language independent approach to parallel sentence mining from comparable corpora and showed that it is capable of state of the art performances.

Currently, PEXACC (together with a derived application based on its translation similarity measure) is used in the ACCURAT project to mine for parallel sentence pairs in the CC collected by the ACCURAT partners. SMT improvement based on the extracted data is, at the time of writing, under testing.

PEXACC is available for download in the "ACCURAT Toolkit", a collection of tools to extract MT-useful data from CC. The ACCURAT project website[7] gives all the details.

7. Acknowledgements

This work has been supported by the ACCURAT project funded by the European Community's Seventh Framework Program (FP7/2007-2013) under the Grant Agreement n° 248347.

8. References

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

Thi Ngoc Diep, D., Besacier, L., Castelli, E. (2010). A Fully Unsupervised Approach for Mining Parallel Data from Comparable Corpora. In Proceedings of the 14th Annual Conference of the European Association for Machine Translation (EAMT 2010), Saint-Raphaël, France, 27-28 May 2010.

Fung, P. and Cheung, P. (2004). Mining Very-Non-Parallel Corpora: Parallel Sentence and Lexicon Extraction via Bootstrapping and EM. In *Proceedings of EMNLP 2004*, Barcelona, Spain. July 2004.

Gao, Q. and Vogel, S. (2008). Parallel Implementations of a Word Alignment Tool. In *Proceedings of ACL-08 HLT: Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, June 20, 2008. The Ohio State University, Columbus, Ohio, USA, pp. 49—57.

Ion, R., Pinnis, M., Ştefănescu, D., Aker, A., Paramita, M., Su, F., Irimia, E., Zhang, X. and Ljubešić, N. (2011a). Toolkit for multi-level alignment and information extraction from comparable corpora. Technical report no. D2.6 of the ACCURAT Project (http://www.accurat-project.eu/).

Ion, R. Ceauşu, A. and Irimia, E. (2011b). An Expectation Maximization Algorithm for Textual Unit Alignment. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora (BUCC 2011)* held at the 49th Annual Meeting of the Association for Computational Linguistics, Portland, Oregon, USA, June 24th, 2011, pp. 128—135.

Ion, R., Zhang, X., Su, F., Paramita, M. and Ştefănescu, D. (2011c). Report on multi-level alignment of comparable corpora Technical report no. D2.2 of the ACCURAT Project (http://www.accurat-project.eu/).

Munteanu, D.S., and Marcu, D. (2005). Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4), pp. 477–504.

Rauf, S.A. and Schwenk, H. (2011). Parallel sentence generation from comparable corpora for improved SMT. *Machine Translation*, 25, pp. 341—375.

Quirk, C., Udupa, R., and Menezes, A. (2007). Generative Models of Noisy Translations with Applications to Parallel Fragment Extraction. In *Proceedings of the MT Summit XI*, Copenhagen, Demark, September, 2007, pp. 321–327.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th conference of the Association for Machine Translation in the Americas (AMTA 2006): visions for the future of machine translation*, Cambridge, MA, USA, pp. 223–231.

Snover, M., Madnani, N., Dorr, B. and Schwartz, R. (2009). Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, Association for Computational Linguistics, Athens, Greece, pp. 259–268.

Talvensaari, T., Pirkola, A., Järvelin, K., Juhola, M. and Laurikkala, J. (2008). Focused web crawling in the acquisition of comparable corpora. *Information Retrieval*, 11(5), pp. 427—445.

Tao, T., and Zhai, C.X. (2005). Mining Comparable Bilingual Text Corpora for Cross Language Information Integration. In *Proceedings of KDD'05*, August 21-24, 2005, Chicago, Illinois, USA.

Tillmann, C. (2009). A Beam-Search Extraction Algorithm for Comparable Data. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, Suntec, Singapore, 4 August 2009, pp. 225–228.

Tsvetkov, Y. and Wintner, S. (2010). Automatic Acquisition of Parallel Corpora from Websites with Dynamic Content. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010, pp. 3389—3392.

Tufiş, D., Ion, R., Ceauşu, A. and Ştefănescu, D. (2006). Improved Lexical Alignment by Combining Multiple Reified Alignments. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, Trento, Italy, April 3–7 2006, pp. 153–160.

Zhang, Y., Wu, K., Gao, J. and Vines, P. (2006). Automatic Acquisition of Chinese-English Parallel Corpus from the Web. In *Proceedings of 28th European Conference on Information Retrieval*.

[7] http://www.accurat-project.eu/