A finite-state approach to phrase-based statistical machine translation

Jorge González
Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València
València, Spain
jgonzalez@dsic.upv.es

Abstract

This paper presents a finite-state approach to phrase-based statistical machine translation where a log-linear modelling framework is implemented by means of an on-the-fly composition of weighted finite-state transducers. Moses, a well-known state-of-the-art system, is used as a machine translation reference in order to validate our results by comparison. Experiments on the TED corpus achieve a similar performance to that yielded by Moses.

1 Introduction

Statistical machine translation (SMT) is a pattern recognition approach to machine translation which was defined by Brown et al. (1993) as follows: given a sentence $\mathbf{s}$ from a certain source language, a corresponding sentence $\hat{\mathbf{t}}$ in a given target language that maximises the posterior probability $\Pr(\mathbf{t}|\mathbf{s})$ is to be found. State-of-the-art SMT systems model the translation distribution $\Pr(\mathbf{t}|\mathbf{s})$ via the log-linear approach (Och and Ney, 2002):

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \Pr(\mathbf{t}|\mathbf{s}) \quad (1)$$

$$\approx \arg\max_{\mathbf{t}} \sum_{m=1}^{M} \lambda_m h_m(\mathbf{s}, \mathbf{t}) \quad (2)$$

where $h_m(\mathbf{s}, \mathbf{t})$ is a logarithmic function representing an important feature for the translation of $\mathbf{s}$ into $\mathbf{t}$, M is the number of features (or models), and λ_m is the weight of h_m in the log-linear combination.

This feature set typically includes several *translation* models so that different relations between a source and a target sentence can be considered. Nowadays, these models are strongly based on phrases, i.e. variable-length n-grams, which means that they are built from some other lower-context models that, in this case, are defined at phrase level. *Phrase-based* (PB) models (Tomas and Casacuberta, 2001; Och and Ney, 2002; Marcu and Wong, 2002; Zens et al., 2002) constitute the core of the current state-of-the-art in SMT. The basic idea of PB-SMT systems is:

1. to segment the source sentence into phrases, then
2. to translate each source phrase into a target phrase, and finally
3. to reorder them in order to compose the final translation in the target language.

In a monotone translation framework however, the third step is omitted as the final translation is just generated by concatenation of the target phrases.

Apart from translation functions, the log-linear approach is also usually composed by means of a target language model and some other additional elements such as word penalties or phrase penalties. The word and phrase penalties allow an SMT system to limit the number of words or target phrases, respectively, that constitute a translation hypothesis.

In this paper, a finite-state approach to a PB-SMT state-of-the-art system, Moses (Koehn et al., 2007), is presented. Experimental results validate our work because they are similar to those yielded by Moses. A related study can be found in Kumar et al. (2006) for the alignment template model (Och et al., 1999).

Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing, pages 90–98,
Donostia–San Sebastián, July 23–25, 2012.

2 Log-linear features for monotone SMT

As a first approach to Moses using finite-state models, a monotone PB-SMT framework is adopted. Under this constraint, Moses' log-linear model is usually taking into account the following 7 features:

Translation features

1. Direct PB translation probability
2. Inverse PB translation probability
3. Direct PB lexical weighting
4. Inverse PB lexical weighting

Penalty features

5. PB penalty
6. Word penalty

Language features

7. Target language model

2.1 Translation features

All 4 features related to translation are PB models, that is, their associated feature functions $h_m(\mathbf{s}, \mathbf{t})$, which are in any case defined for full sentences, are modelled from other PB distributions $\eta_m(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})$, which are based on phrases.

Direct PB translation probability

The first feature $h_1(\mathbf{s}, \mathbf{t}) = \log P(\mathbf{t}|\mathbf{s})$ is based on modelling the posterior probability by using the segmentation between $\mathbf{s}$ and $\mathbf{t}$ as a hidden variable β_1. In this manner, $\Pr(\mathbf{t}|\mathbf{s}) = \sum_{\beta_1} \Pr(\mathbf{t}|\mathbf{s}, \beta_1)$ is then approximated by $P(\mathbf{t}|\mathbf{s})$ by using maximization instead of summation: $P(\mathbf{t}|\mathbf{s}) = \max_{\beta_1} P(\mathbf{t}|\mathbf{s}, \beta_1)$.

Given a monotone segmentation between $\mathbf{s}$ and $\mathbf{t}$, $P(\mathbf{t}|\mathbf{s}, \beta_1)$ is generatively computed as the product of the translation probabilities for each segment pair according to some PB probability distributions:

$$P(\mathbf{t}|\mathbf{s}, \beta_1) = \prod_{k=1}^{|\beta_1|} P(\tilde{\mathbf{t}}_k|\tilde{\mathbf{s}}_k)$$

where $|\beta_1|$ is the number of phrases that $\mathbf{s}$ and $\mathbf{t}$ are segmented into, i.e. every $\tilde{\mathbf{s}}_k$ and $\tilde{\mathbf{t}}_k$, respectively, whose dependence on β_1 is omitted for the sake of an easier reading.

Feature 1 is finally formulated as follows:

$$h_1(\mathbf{s}, \mathbf{t}) = \log \max_{\beta_1} \prod_{k=1}^{|\beta_1|} P(\tilde{\mathbf{t}}_k|\tilde{\mathbf{s}}_k) \qquad (3)$$

where $\eta_1(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}) = P(\tilde{\mathbf{t}}|\tilde{\mathbf{s}})$ is a set of PB probability distributions estimated from bilingual training data, once statistically word-aligned (Brown et al., 1993) by means of GIZA++ (Och and Ney, 2003), which Moses relies on as far as training is concerned. This information is organized as a *translation table* where a pool of phrase pairs is previously collected.

Inverse PB translation probability

Similar to what happens with Feature 1, Feature 2 is formulated as follows:

$$h_2(\mathbf{s}, \mathbf{t}) = \log \max_{\beta_2} \prod_{k=1}^{|\beta_2|} P(\tilde{\mathbf{s}}_k|\tilde{\mathbf{t}}_k) \qquad (4)$$

where $\eta_2(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}) = P(\tilde{\mathbf{s}}|\tilde{\mathbf{t}})$ is another set of PB probability distributions, which are simultaneously trained together with the ones for Feature 1, $P(\tilde{\mathbf{t}}|\tilde{\mathbf{s}})$, over the same pool of phrase pairs already extracted.

Direct PB lexical weighting

Given the word-alignments obtained by GIZA++, it is quite straight-forward to estimate a maximum likelihood stochastic dictionary $P(\mathbf{t}_i|\mathbf{s}_j)$, which is used to score a weight $D(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})$ to each phrase pair in the pool. Details about the computation of $D(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})$ are given in Koehn et al. (2007). However, as far as this work is concerned, these details are not relevant.

Feature 3 is then similarly formulated as follows:

$$h_3(\mathbf{s}, \mathbf{t}) = \log \max_{\beta_3} \prod_{k=1}^{|\beta_3|} D(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k) \qquad (5)$$

where $\eta_3(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}) = D(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})$ is yet another score to use with the pool of phrase pairs aligned during training.

Inverse PB lexical weighting

Similar to what happens with Feature 3, Feature 4 is formulated as follows:

$$h_4(\mathbf{s}, \mathbf{t}) = \log \max_{\beta_4} \prod_{k=1}^{|\beta_4|} I(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k) \qquad (6)$$

where $\eta_4(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}) = I(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})$ is another weight vector, which is computed by using a dictionary $P(\mathbf{s}_j|\mathbf{t}_i)$, with which the translation table is expanded again, thus scoring a new weight per phrase pair in the pool.

2.2 Penalty features

The penalties are not modelled in the same way. The PB penalty is similar to a translation feature, i.e. it is based on a monotone sentence segmentation. The word penalty however is formulated as a whole, being taken into account by Moses at decoding time.

PB penalty

The PB penalty scores $e = 2.718$ per phrase pair, thus modelling somehow the segmentation length. Therefore, Feature 5 is defined as follows:

$$h_5(\mathbf{s}, \mathbf{t}) = \log \max_{\beta_5} \prod_{k=1}^{|\beta_5|} e \qquad (7)$$

where $\eta_5(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}) = e$ extends the PB table once again.

Word penalty

Word penalties are not modelled as PB penalties. In fact, this feature is not defined from PB scores, but it is formulated at sentence level just as follows:

$$h_6(\mathbf{s}, \mathbf{t}) = \log e^{|\mathbf{t}|} \qquad (8)$$

where the exponent of e is the number of words in $\mathbf{t}$.

2.3 Language features

Language models approach the a priori probability that a given sentence belongs to a certain language. In SMT, they are usually employed to guarantee that translation hypotheses are built according to the peculiarities of the target language.

Target language model

An n-gram is used as target language model $P(\mathbf{t})$, where a word-based approach is usually considered. Then, $h_7(\mathbf{s}, \mathbf{t}) = \log P(\mathbf{t})$ is based on a model where sentences are generatively built word by word under the influence of the last $n-1$ previous words, with the cutoff derived from the start of the sentence:

$$h_7(\mathbf{s}, \mathbf{t}) = \log \prod_{i=1}^{|\mathbf{t}|} P(\mathbf{t}_i|\mathbf{t}_{i-n+1} \dots \mathbf{t}_{i-1}) \qquad (9)$$

where $P(\mathbf{t}_i|\mathbf{t}_{i-n+1} \dots \mathbf{t}_{i-1})$ are word-based probability distributions learnt from monolingual corpora.

3 Data structures

This section shows how the features from Section 2 are actually organized into different data structures in order to be efficiently used by the Moses decoder, which implements the search defined by Equation 2 to find out the most likely translation hypothesis $\hat{\mathbf{t}}$ for a given source sentence $\mathbf{s}$.

3.1 PB models

The PB distributions associated to Features 1 to 5 are organized in table form as a translation table for the collection of phrase pairs previously extracted. That builds a PB database similar to that in Table 1

Source	Target	η_1	η_2	η_3	η_4	η_5
barato	low cost	1	0.3	1	0.6	2.718
me gusta	I like	0.6	1	0.9	1	2.718
es decir	that is	0.8	0.5	0.7	0.9	2.718
por favor	please	0.4	0.2	0.1	0.4	2.718
...			...			2.718

Table 1: A Spanish-into-English PB translation table. Each source-target phrase pair is scored by all η models.

where each phrase pair is scored by all five models.

3.2 Word-based models

Whereas PB models are an interesting approach to deal with translation relations between languages, language modelling itself is usually based on words. Feature 6 is a length model of the target sentence, and Feature 7 is a target language model.

Word penalty

Penalties are not models that need to be trained. However, while PB penalties are provided to Moses to take them into account during the search process (see for example the last column of Table 1, η_5), word penalties are internally implemented in Moses as part of the log-linear maximization in Equation 2, and are automatically computed on-the-fly at search.

Target n-gram model

Language models, and n-grams in particular, suffer from a sparseness problem (Rosenfeld, 1996). The n-gram probability distributions are smoothed to be able to deal with the unseen events out of training data, thus aiming for a larger language coverage.

This smoothing is based on the *backoff* method, which introduces some penalties for level downgrading within hierarchical language models. For example, let $\mathcal{M}$ be a trigram language model, which, as regards smoothing, needs both a bigram and a unigram model trained on the same data. Any trigram probability, $P(c|ab)$, is then computed as follows:

$$\begin{array}{lrl} \text{if} & abc \in \mathcal{M}: & P_{\mathcal{M}}(c|ab) \\ \text{elseif} & bc \in \mathcal{M}: & BO_{\mathcal{M}}(ab) P_{\mathcal{M}}(c|b) \\ \text{elseif} & c \in \mathcal{M}: & BO_{\mathcal{M}}(ab)\, BO_{\mathcal{M}}(b) P_{\mathcal{M}}(c) \\ \text{else} & : & BO_{\mathcal{M}}(ab)\, BO_{\mathcal{M}}(b) P_{\mathcal{M}}(unk) \end{array} \quad (10)$$

where $P_{\mathcal{M}}$ is the probability estimated by $\mathcal{M}$ for the corresponding n-gram, $BO_{\mathcal{M}}$ is the backoff weight to deal with the unseen events out of training data, and finally, $P_{\mathcal{M}}(unk)$ is the probability mass reserved for unknown words.

The $P(\mathbf{t}_i|\mathbf{t}_{i-n+1}\dots\mathbf{t}_{i-1})$ term from Equation 9 is then computed according to that algorithm above, given the model data organized again in table form as a collection of probabilities and backoff weights for the n-grams appearing in the training corpus. This model displays similarly to that in Table 2.

n-gram	P	BO
please	0.02	0.2
low cost	0.05	0.3
I like	0.1	0.7
that is	0.08	0.5
...	...	

Table 2: An English word-based backoff n-gram model. The likelihood and the backoff model score for each n-gram.

4 Weighted finite-state transducers

Weighted finite-state transducers (Mohri et al., 2002) (WFSTs) are defined by means of a tuple $(\Sigma, \Delta, Q, q_0, f, P)$, where Σ is the alphabet of input symbols, Δ is the alphabet of output symbols, Q is a finite set of states, $q_0 \in Q$ is the initial state, $f : Q \to \mathbb{R}$ is a state-based weight distribution to quantify that states may be final states, and finally, the partial function $P : Q \times \Sigma^\star \times \Delta^\star \times Q \to \mathbb{R}$ defines a set of edges between pairs of states in such a way that every edge is labelled with an input string in $\Sigma^\star$, with an output string in $\Delta^\star$, and is assigned a transition weight.

When weights are probabilities, i.e. the range of functions f and P is constrained between 0 and 1, and under certain conditions, a weighted finite-state transducer may define probability distributions. Then, it is called a *stochastic finite-state transducer*.

4.1 WFSTs for SMT models

Here, we show how the SMT models described in Section 3 (that is, the five η scores in the PB translation table, the word penalty, and the n-gram language model) are represented by means of WFSTs.

First of all, the word penalty feature in Equation 8 is equivalently reformulated as another PB score, as in Equations 3 to 7:

$$h_6(\mathbf{s}, \mathbf{t}) = \log e^{|\mathbf{t}|} = \log \max_{\beta_6} \prod_{k=1}^{|\beta_6|} e^{|\tilde{\mathbf{t}}_k|} \quad (11)$$

where the length of $\mathbf{t}$ is split up by summation using the length of each phrase in a segmentation β_6. Actually, this feature is independent of β_6, that is, any segmentation produces the expected value $e^{|\mathbf{t}|}$, and therefore the maximization by β_6 is not needed. However, the main goal is to introduce this feature as another PB score similar to those in Features 1 to 5, and so it is redefined following the same framework. The PB table can be now extended by means of $\eta_6(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}) = e^{|\tilde{\mathbf{t}}|}$, just as Table 3 shows.

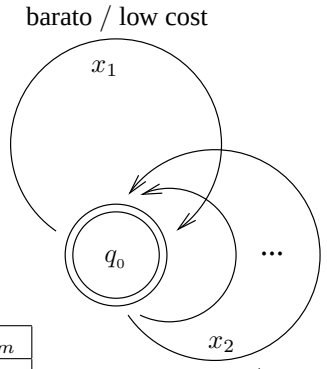
Source	Target	η_1	η_2	η_3	η_4	η_5	η_6
barato	low cost			...		e	e^2
me gusta	I like			...		e	e^2
es decir	that is			...		e	e^2
por favor	please			...		e	e
...				...			

Table 3: A word-penalty-extended PB translation table. The exponent of e in η_6 is the number of words in Target.

Now, the translation table including 6 PB scores and the target-language backoff n-gram model can be expressed by means of (some stochastic) WFSTs.

Translation table

Each PB model included in the translation table, i.e. any PB distribution in $\{\eta_1(\tilde{\mathbf{s}}, \tilde{\mathbf{t}}), \ldots, \eta_6(\tilde{\mathbf{s}}, \tilde{\mathbf{t}})\}$, can be represented as a particular case of a WFST. Figure 1 shows a PB score encoded as a WFST, using a different looping transition per table row within a WFST of only one state.



Source	Target	η_m
barato	low cost	x_1
me gusta	I like	x_2
...		...

Figure 1: Equivalent WFST representation of PB scores. Table rows are embedded within as many looping transitions of a WFST which has no topology at all; η-scores are correspondingly stored as transition weights.

It is straight-forward to see that the application of the Viterbi method (Viterbi, 1967) on these WFSTs provides the corresponding feature value $h_m(\mathbf{s}, \mathbf{t})$ for all Features 1 to 6 as defined in Equations 3 to 8.

Language model

It is well known that n-gram models are a subclass of stochastic finite-state automata where backoff can also be adequately incorporated (Llorens, 2000).

Then, they can be equivalently turned into transducers by means of the concept of identity, that is, transducers which map every input label to itself. Figure 2 shows a WFST for a backoff bigram model.

It is also quite straight-forward to see that $h_7(\mathbf{s}, \mathbf{t})$ (as defined in Equation 9 for a target n-gram model where backoff is adopted according to Equation 10) is also computed by means of a parsing algorithm, which is actually a process that is simple to carry out given that these backoff n-gram WFSTs are deterministic.

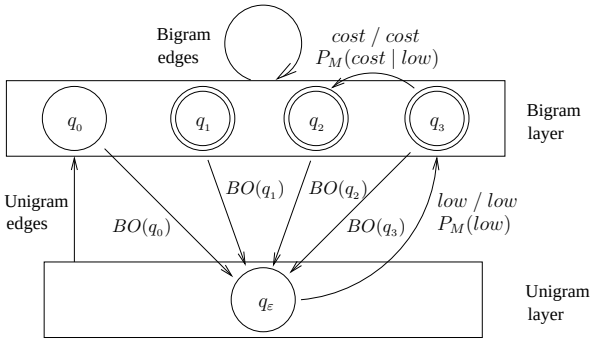


Figure 2: A WFST example for a backoff bigram model. Backoff (BO) is dealt with failure transitions from the bigram layer to the unigram layer. Unigrams go in the other direction and bigrams link states within the bigram layer.

To sum up, our log-linear combination scenario considers 7 (some stochastic) WFSTs, 1 per feature: 6 of them are PB models related to a translation table while the 7th one is a target-language n-gram model.

Next in Section 4.2, we show how these WFSTs are used in conjunction in a homogeneous framework.

4.2 Search

Equation 2 is a general framework for log-linear approaches to SMT. This framework is adopted here in order to combine several features based on WFSTs, which are modelled as their respective Viterbi score.

As already mentioned, the computation of $h_m(\mathbf{s}, \mathbf{t})$ for each PB-WFST, let us say $\mathcal{T}_m$ (with $1 \leq m \leq 6$), provides the most likely segmentation β_m for $\mathbf{s}$ and $\mathbf{t}$ according to $\mathcal{T}_m$. However, a constraint is used here so that all $\mathcal{T}_m$ models define the same segmentation β:

$$|\beta| > 0$$
$$\mathbf{s} = \tilde{\mathbf{s}}_1 \ldots \tilde{\mathbf{s}}_{|\beta|}$$
$$\mathbf{t} = \tilde{\mathbf{t}}_1 \ldots \tilde{\mathbf{t}}_{|\beta|}$$

where the PB scores corresponding to Features 1 to 6 are directly applied on that particular segmentation for each phrase pair $(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k)$ monotonically aligned. Equations 3 to 7 and 11 can be simplified as follows:

$$\forall m = 1, \ldots, 6$$
$$h_m(\mathbf{s}, \mathbf{t}) = \log \max_{\beta} \prod_{k=1}^{|\beta|} \eta_m(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k) \qquad (12)$$

Then, Equation 2 can be instanced as follows:

$$\hat{\mathbf{t}} = \arg\max_{\mathbf{t}} \sum_{m=1}^{7} \lambda_m \, h_m(\mathbf{s}, \mathbf{t}) \quad (13)$$

$$= \arg\max_{\mathbf{t}} \left[\sum_{m=1}^{6} \lambda_m \max_{\beta} \sum_{k=1}^{|\beta|} \log \eta_m(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k) \right] + \lambda_7 \sum_{i=1}^{|\mathbf{t}|} \log P(\mathbf{t}_i | \mathbf{t}_{i-n+1} \ldots \mathbf{t}_{i-1})$$

$$= \arg\max_{\mathbf{t}} \left[\max_{\beta} \sum_{k=1}^{|\beta|} \sum_{m=1}^{6} \lambda_m \log \eta_m(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k) \right] + \sum_{i=1}^{|\mathbf{t}|} \lambda_7 \log P(\mathbf{t}_i | \mathbf{t}_{i-n+1} \ldots \mathbf{t}_{i-1})$$

as logarithm rules are applied to Equations 9 and 12.

The square-bracketed expression of Equation 13 is a Viterbi-like score which can be incrementally built through the contribution of all the PB-WFSTs (along with their respective λ_m-weights) over some phrase pair $(\tilde{\mathbf{s}}_k, \tilde{\mathbf{t}}_k)$ that extends a partial hypothesis. As these models share their topology, we implement them jointly including as many scores per transition as needed (González and Casacuberta, 2008). These models can also be merged by means of union once their λ_m-weights are transferred into them. That allows us to model the whole translation table (see Table 3) by means of just 1 WFST structure $\mathcal{T}$. Therefore, the search framework for single models can also be used for their log-linear combination.

As regards the remaining term from Equation 13, i.e. the target n-gram language model for Feature 7, it is seen as a rescoring function (Och et al., 2004) which is applied once the PB-WFST $\mathcal{T}$ is explored. The translation model returns the best hypotheses that are later input to the n-gram language model $\mathcal{L}$, where they are reranked, to finally choose the best $\hat{\mathbf{t}}$.

However, these two steps can be processed at once if both the WFST $\mathcal{T}$ and the WFST $\mathcal{L}$ are merged by means of their composition $\mathcal{T} \circ \mathcal{L}$ (Mohri, 2004). The product of such an operation is another WFST as WFSTs are closed under a composition operation. In practice though, the size of $\mathcal{T} \circ \mathcal{L}$ can be very large so composition is done on-the-fly (Caseiro, 2003), which actually does not build the WFST for $\mathcal{T} \circ \mathcal{L}$ but explores both $\mathcal{T}$ and $\mathcal{L}$ as if they were composed, using the n-gram scores in $\mathcal{L}$ on the target hypotheses from $\mathcal{T}$ as soon as they are partially produced.

Equation 13 represents a Viterbi-based composition framework where all the (weighted) models contribute to the overall score to be maximized, provided that the set of λ_m-weights is instantiated. Using a development corpus, the set of λ_m-weights can be empirically determined by means of running several iterations of this framework, where different values for the λ_m-weights are tried in each iteration.

5 Experiments

Experiments were carried out on the TED corpus, which is described in depth throughout Section 5.1. Automatic evaluation for SMT is often considered and we use the measures enumerated in Section 5.2. Results are shown and also discussed, in Section 5.3.

5.1 Corpora data

The TED corpus is composed of a collection of English-French sentences from audiovisual content whose main statistics are displayed in Table 4.

Subset		English	French
Train	Sentences	47.5K	
	Running words	747.2K	792.9K
	Vocabulary	24.6K	31.7K
Develop	Sentences	571	
	Running words	9.2K	10.3K
	Vocabulary	1.9K	2.2K
Test	Sentences	641	
	Running words	12.6K	12.8K
	Vocabulary	2.4K	2.7K

Table 4: Main statistics from the TED corpus and its split.

As shown in Table 4, develop and test partitions are statistically comparable. The former is used to train the λ_m-weights in the log-linear approach, in the hope that they can also work well for the latter.

5.2 Evaluation measures

Since its appearance as a translation quality measure, the BLEU metric (Papineni et al., 2002), which stands for *bilingual evaluation understudy*, has become consolidated in the area of automatic evaluation as the most widely used SMT measure. Nevertheless, it was later found that its correlation factor

with subjective evaluations (the original reason for its success) is actually not so high as first thought (Callison-Burch et al., 2006). Anyway, it is still the most popular SMT measure in the literature.

However, the *word error rate* (WER) is a very common measure in the area of speech recognition which is also quite usually applied in SMT (Och et al., 1999). Although it is not so widely employed as BLEU, there exists some work that shows a better correlation of WER with human assessments (Paul et al., 2007). Of course, the WER measure has some bad reviews as well (Chen and Goodman, 1996; Wang et al., 2003) and one of the main criticisms that it receives in SMT areas is about the fact that there is only one translation reference to compare with. The MWER measure (Nießen et al., 2000) is an attempt to relax this dependence by means of an average error rate with respect to a set of multiple references of equivalent meaning, provided that they are available.

Another measure also based on the edit distance concept has recently arisen as an evolution of WER towards SMT. It is the *translation edit rate* (TER), and it has become popular because it takes into account the basic post-process operations that professional translators usually do during their daily work. Statistically, it is considered as a measure highly correlated with the result of one or more subjective evaluations (Snover et al., 2006).

The definition of these evaluation measures is as follows:

BLEU: It computes the precision of the unigrams, bigrams, trigrams, and fourgrams that appear in the hypotheses with respect to the n-grams of the same order that occur in the translation reference, with a penalty for too short sentences. Unlike the WER measure, BLEU is not an error rate but an accuracy measure.

WER: This measure computes the minimum number of editions (replacements, insertions or deletions) that are needed to turn the system hypothesis into the corresponding reference.

TER: It is computed similarly to WER, using an additional edit operation. TER allows the movement of phrases, besides replacements, insertions, and deletions.

5.3 Results

The goal of this section is to assess experimentally the finite-state approach to PB-SMT presented here. First, an English-to-French translation is considered, then a French-to-English direction is later evaluated.

On the one hand, our log-linear framework is tuned on the basis of BLEU as the only evaluation measure in order to select the best set of λ_m-weights. That is accomplished by means of development data, however, once the λ_m-weights are estimated, they are extrapolated to test data for the final evaluation. Table 5 shows: a) the BLEU translation results for the development data; and b) the BLEU, WER and TER results for the test data. In both a) and b), the λ_m-weights are trained on the development partition. These results are according to different feature combinations in our log-linear approach to PB-SMT.

As shown in Table 5, the first experimental scenario is not a log-linear framework since only one feature, (a direct PB translation probability model) is considered. The corresponding results are poor and, judging by the remaining results in Table 5, they reflect the need for a log-linear approach.

The following experiments in Table 5 represent a log-linear framework for Features 1 to 6, i.e. the PB translation table encoded as a WFST $\mathcal{T}$, where different PB models are the focus of attention. Only the log-linear combination of Features 1 and 2

Log-linear features	Develop BLEU	Test BLEU	Test WER	Test TER
1 (baseline)	8.5	7.1	102.9	101.5
1+2	4.0	3.0	116.6	115.6
1+2+3	22.7	18.4	66.6	64.4
1+2+3+4	22.8	18.5	66.3	64.2
1+2+3+4+5	22.7	18.8	65.2	63.2
1+2+3+4+5+6	23.1	19.1	65.9	63.8
1+7	24.6	20.5	65.1	62.9
1+2+7	25.5	21.3	63.7	61.6
1+2+3+7	25.9	**22.2**	**62.5**	**60.4**
1+2+3+4+7	26.3	22.0	63.4	61.3
1+2+3+4+5+7	26.4	22.1	63.1	61.0
1+2+3+4+5+6+7	27.0	21.8	64.4	62.2
Moses (1+...+7)	27.1	22.0	64.0	61.8

Table 5: English-to-French results for development and test data according to different log-linear scenarios. The set of λ_m-weights is learnt from development data for every feature combination log-linear scenario defined.

is worse than the baseline, which feeds us back on the fact that the λ_m-weights can be better trained, that is, the log-linear model for Features 1 and 2 can be upgraded until baseline's results with $\lambda_2 = 0$.

This battery of experiments on Features 1 to 6 allows us to see the benefits of a log-linear approach. The baseline results are clearly outperformed now, and we can say that the more features are included, the better are the results.

The next block of experiments in Table 5 always include Feature 7, i.e. the target language model $\mathcal{L}$. Features 1 to 6 are progressively introduced into $\mathcal{T}$. These results confirm that the target language model is still an important feature to take into account, even though PB models are already providing a surrounding context for their translation hypotheses because translation itself is modelled at phrase level. These results are significantly better than the ones where the target language model is not considered. Again, the more translation features are included, the better are the results on the development data. However, an overtraining is presumedly occurring with regard to the optimization of the λ_m-weights, as results on the test partition do not reach their top the same way the ones for the development data do, i.e. when using all 7 features, but when combining Features 1, 2, 3, and 7, instead. These differences are not statistically significant though.

Finally, our finite-state approach to PB-SMT is validated by comparison, as it allows us to achieve similar results to those yielded by Moses itself.

On the other hand, a translation direction where French is translated into English gets now the focus. Their corresponding results are presented in Table 6. A similar behaviour can be observed in Table 6 for the series of French-to-English empirical results.

Log-linear features	Develop BLEU	Test BLEU	Test WER	Test TER
1 (baseline)	7.1	7.4	101.6	100.0
1+2	4.1	3.5	117.5	116.0
1+2+3	24.2	21.1	58.9	56.5
1+2+3+4	24.4	20.8	58.0	55.7
1+2+3+4+5	24.9	21.2	56.9	54.8
1+2+3+4+5+6	25.2	21.2	57.1	55.0
1+7	24.7	22.5	60.0	57.7
1+2+7	26.0	23.2	58.8	56.5
1+2+3+7	28.5	23.0	56.1	54.0
1+2+3+4+7	28.4	23.1	56.0	53.8
1+2+3+4+5+7	28.8	23.4	56.0	53.9
1+2+3+4+5+6+7	28.7	**23.8**	**55.8**	**53.7**
Moses (1+...+7)	28.9	23.5	55.8	53.6

Table 6: French-to-English results for development and test data according to different log-linear scenarios.

6 Conclusions and future work

In this paper, a finite-state approach to Moses, which is a PB-SMT state-of-the-art system, is presented. A monotone framework is adopted, where 7 models in log-linear combination are considered: a direct and an inverse PB translation probability model, a direct and an inverse PB lexical weighting model, PB and word penalties, and a target language model.

Five out of these models are based on PB scores which are organized under a PB translation table. These models can also be implemented by means of WFSTs on the basis of the Viterbi algorithm. The word penalty can also be equivalently redefined as another PB model, similar to the five others, which allows us to constitute a translation model $\mathcal{T}$ composed of six parallel WFSTs that are constrained to share the same monotonic bilingual segmentation.

A backoff n-gram model for the target language $\mathcal{L}$ can be represented as an identity WFST where $P(\mathbf{t})$ is modelled on the basis of the Viterbi algorithm. The whole log-linear approach to Moses is attained by means of the on-the-fly WFST composition $\mathcal{T} \circ \mathcal{L}$.

Our finite-state log-linear approach to PB-SMT is validated by comparison, as it has allowed us to achieve similar results to those yielded by Moses.

Monotonicity is an evident limitation of this work, as Moses can also feature some limited reordering. However, future work on that line is straight-forward since the framework described in this paper can be easily extended to include a PB reordering model $\mathcal{R}$, by means of the on-the-fly composition $\mathcal{T} \circ \mathcal{R} \circ \mathcal{L}$.

Acknowledgments

The research leading to these results has received funding from the European Union 7th Framework Programme (FP7/2007-2013) under grant agreement no. 287576. Work also supported by the EC (FEDER, FSE), the Spanish government (MICINN, MITyC, "Plan E", grants MIPRCV "Consolider Ingenio 2010" and iTrans2 TIN2009-14511), and the Generalitat Valenciana (grant Prometeo/2009/014).

References

P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, and R.L. Mercer. 1993. The mathematics of machine translation. In *Computational Linguistics*, volume 19, pages 263–311, June.

C. Callison-Burch, M. Osborne, and P. Koehn. 2006. Re-evaluating the Role of Bleu in Machine Translation Research. In *Proceedings of the 11th conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256.

D. Caseiro. 2003. *Finite-State Methods in Automatic Speech Recognition.* PhD Thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

S. F. Chen and J. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting of the Association for Computational Linguistics*, pages 310–318.

J. González and F. Casacuberta. 2008. A finite-state framework for log-linear models in machine translation. In *Proc. of European Association for Machine Translation*, pages 41–46.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proc. of Association for Computational Linguistics*, pages 177–180.

S. Kumar, Y. Deng, and W. Byrne. 2006. A weighted finite state transducer translation template model for statistical machine translation. *Natural Language Engineering*, 12(1):35–75.

David Llorens. 2000. *Suavizado de autómatas y traductores finitos estocásticos.* PhD Thesis, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia.

D. Marcu and W. Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proc. of Empirical methods in natural language processing*, pages 133–139.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88.

Mehryar Mohri. 2004. Weighted finite-state transducer algorithms: An overview. *Formal Languages and Applications*, 148:551–564.

S. Nießen, F. J. Och, G. Leusch, and H. Ney. 2000. An evaluation tool for machine translation: Fast evaluation for MT research. In *Proceedings of the 2nd international Conference on Language Resources and Evaluation*, pages 39–45.

F.J. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of Association for Computational Linguistics*, pages 295–302.

F.J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51, March.

F. J. Och, C. Tillmann, and H. Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the joint conference on Empirical Methods in Natural Language Processing and the 37th annual meeting of the Association for Computational Linguistics*, pages 20–28.

F.J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. 2004. A smorgasbord of features for statistical machine translation. In D. Marcu S. Dumais and S. Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 161–168, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

M. Paul, A. Finch, and E. Sumita. 2007. Reducing human assessment of machine translation quality to binary classifiers. In *Proceedings of the conference on Theoretical and Methodological Issues in machine translation*, pages 154–162.

R. Rosenfeld. 1996. A maximum entropy approach to adaptative statistical language modeling. *Computer Speech and Language*, 10:187–228.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th biennial conference of the Association for Machine Translation in the Americas*, pages 223–231.

J. Tomas and F. Casacuberta. 2001. Monotone statistical translation using word groups. In *Proc. of the Machine Translation Summit*, pages 357–361.

A. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

Y. Wang, A. Acero, and C. Chelba. 2003. Is word error rate a good indicator for spoken language understanding accuracy. In *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding*, pages 577–582.

Richard Zens, Franz Josef Och, and Hermann Ney. 2002. Phrase-based statistical machine translation. In *Proc. of Advances in Artificial Intelligence*, pages 18–32.