Automated Machine Translation for Regional Languages

Problem Statement

India is a highly multilingual country with eighteen constitutionally recognized languages and several hundred dialects & other living languages. Even though, English is understood by less than 3% of Indian population, it continues to be the de-facto link language for administration, education and business. Hindi, which is official language of the country, is used by more than 400 million people. Therefore, machine translation assumes a much greater significance in breaking the language barrier within the country's sociological structure. as English continues to be the link language, a machine translation system catering to English as the source language and the target language being all Indian languages, was considered to be a priority. Further, as the state of current technology is short of producing high quality automated translation and the human translators are unable to cope up with the volume, a machine-aided translation (MAT) system is an obvious answer.

ANGLABHARTI (Sinha et.al., 1995) is a rule-based MAT system with source language as English and uses a pseudo-interlingua to cater to all Indian languages. Although, the design methodology of Anglabharti, is geared to achieve an 'acceptable' translation at the first instance, it is recognized that the system will have inherent weaknesses of being short of producing 'quality' translation thus requiring post-editing. AnglaHindi is an English to Hindi version of the ANGLABHARTI translation methodology with a mixture of some example-based translation methodology. AnglaHindi system has been web enabled and is available at URL: http://anglahindi.iitk.ac.in for free translation. This is first such system designed to our knowledge.

Background

The n-gram approach presented in Mariño et al. (2006) has been derived from the work of Casacuberta and Vidal (2004), which used finite state transducers for statistical machine translation. In this approach, units of source and target words are used as basic translation units. Then the translation model is implemented as an n-gram model over the tuples. As it is also done in phrase-based translations, the different translations are scored by a log-linear combination of the translation model and additional models

A first approach of integrating the idea presented in the n-gram approach into phrase-based machine translation was described in Matusov et al. (2006). In contrast to our work, they used the bilingual units as defined in the original approach and they did not use additional word factors. Hasan et al. (2008) used lexicalized triplets to introduce bilingual context into the translation process. These triplets include source words from outside the phrase and form and additional probability p(f|e, e0) that modifies the conventional word probability of f given e depending on trigger words e 0 in the sentence enabling a context-based translation of ambiguous phrases.

Methodology

Bilingual Language Model: The bilingual language model is a standard n-grambased language model trained on bilingual tokens instead of simple words. These bilingual tokens are motivated by the tuples used in n-gram approaches to machine translation. We use different basic units for the n-gram model compared to the n-gram approach, in order to be able to integrate them into a phrase-based translation system. In this context, a bilingual token consists of a target word and all source words that it is aligned to. More formally, given a sentence pair e I 1 = e1...eI and f J 1 = f1...fJ and the corresponding word alignment A = {(i, j)} the following tokens are created: tj = {fj} ∪ {ei|(i, j) ∈ A} (1)

Therefore, the number of bilingual tokens in a sentence equals the number of target words. If a source word is aligned to two target words like the word aller in the example sentence, two bilingual tokens are created: all_aller and the_aller. If, in contrast, a target word is aligned to two

source words, only one bilingual token is created consisting of the target word and both source words. The existence of unaligned words is handled in the following way. If a target word is not aligned to any source word, the corresponding bilingual token consists only of the target word. In contrast, if a source word is not aligned to any word in the target language sentence, this word is ignored in the bilingual language model.

This probability is then used in the log-linear combination of a phrase-based translation system as an additional feature. It is worth mentioning that although it is modeled like a conventional language model, the bilingual language model is an extension to the translation model, since the translation for the source words is modeled and not the fluency of the target text.

To train the model a corpus of bilingual tokens can be created in a straightforward way. In the generation of this corpus the order of the target words de- fines the order of the bilingual tokens. Then we can use the common language modeling tools to train the bilingual language model. As it was done for the normal language model, we used Kneser-Ney smoothing.

Experimental Design

Vector matching alignment Translation equivalence of the bilingual embeddings is evaluated by naive word alignment to match word embeddings by cosine distance.5 The Alignment Error Rates (AER) reported and suggest that bilingual training using given Equation produces embeddings with better translation equivalence compared to those produced by monolingual training.

Phrase-based machine translation Our experiments are performed using the Stanford Phrasal phrase-based machine translation system (Cer et al., 2010). In addition to NIST08 training data, we perform phrase extraction, filtering and phrase table learning with additional data from GALE MT evaluations in the past 5 years. In turn, our baseline is established at 30.01 BLEU and reasonably competitive relative to NIST08 results. We use Minimum Error Rate Training (MERT) (Och, 2003) to tune the decoder. In the phrase-based MT system, we add one feature to bilingual phrase-pairs. For each phrase, the word embeddings are averaged to obtain a feature vector. If a word is not found in the vocabulary, we disregard and assume it is not in the phrase; if no word is found in a phrase, a zero vector is assigned