The Speech Recognition and Machine Translation System of IOIT for IWSLT 2013

Ngoc-Quan Pham, Hai-Son Le
Tat-Thang Vu, Chi-Mai Luong

Institute of Information and Technology (IOIT),
Vietnamese Academy of Science and Technology (VAST)
(quanpn,lehaison,vtthang,lcmai)@ioit.ac.vn

Abstract

This paper describes the Automatic Speech Recognition (ASR) and Machine Translation (MT) systems developed by IOIT for the evaluation campaign of IWSLT2013. For the ASR task, using Kaldi toolkit, we developed the system based on weighted finite state transducer. The system is constructed by applying several techniques, notably, subspace Gaussian mixture models, speaker adaptation, discriminative training, system combination and SOUL, a neural network language model. The techniques used for automatic segmentation are also clarified. Besides, we compared different types of SOUL models in order to study the impact of words of previous sentences in predicting words in language modeling. For the MT task, the baseline system was built based on the open source toolkit N-code, then being augmented by using SOUL on top, i.e., in N-best rescoring phase.

1. Introduction

This paper describes the two systems developed by IOIT, serving the two tasks in the IWSLT 2013 evaluation campaign, namely Automatically Speech Recognition (ASR) and Machine Translation (MT).

The English ASR task focuses on translating TED talks which are a collection of public lectures on a variety of topics, ranging from Technology, Entertainment to Design. Apparently, the hindrances in the track are the spontaneous and natural way of speech, interruption of invalid noises such as music or applauses or dealing with topic adaptation. This year, since the evaluation data is no longer provided with manual sentence segmentation, dividing the long audio files into short utterances properly becomes a new challenging obstacle. For this task, we use Kaldi [1] to construct the system based on state-of-the-art techniques, notably, subspace Gaussian mixture models, speaker adaptation, discriminative training, system combination and SOUL [2], a neural network language model (NNLM). Finally, the system is a combination of two systems differing in acoustic model, augmented by rescoring the output N-best list with SOUL language models. Besides, we study the impact when SOUL language models take into account words of previous sentences in the context.

On the English to French MT task, since it is our first participation, our aim is to build a whole system from scratch using open source toolkits for normalization, tokenization, tagging, data filtering, system construction... which will be served as a baseline system for future research. The system is based on N-code[1], a bilingual n-gram approach for MT and the use of SOUL in N-best rescoring.

The organization of the paper is as follows: Section 2 is the description of our ASR system. While acoustic model training procedure is presented in Section 2.1, the automatic segmentation process is described in Section 2.2. The language modeling with three types of SOUL models are described in Section 2.3. Then, in Section 2.4, the decoding procedure will be presented in detail. Section 2.5 is devoted to ASR experimental results and our analyses . Section 3 is concentrated on the MT task. It consists of three parts: Section 3.1 for data preprocessing, Section 3.2 for the description of our system and Section 3.3 for the experimental evaluation.

2. Automatic Speech Recognition Task

2.1. Acoustic Modeling

2.1.1. Training corpus

We decided to collect TED lectures as training materials, in order to guarantee the homogeneity of training and development data in terms of speaking environment and speaking style. Approximately 220 hours of audio, distributed among 920 talks, were crawled with their subtitles, which were deliberately used for making transcripts. However, the provided subtitles do not contain the correct time stamps corresponding with each phrase as well as the exact pronunciation for the words spoken, which lead to the necessity for long-speech alignment.

Proved to be effective for long-speech alignment task, SailAlign [3, 4] is applied to extract text-aligned speech segments, which helps us to not only acquire the transcript with exact timing, but also to filter non-spoken sounds such as music or applauses. A part of these noises are kept for noise training while most of them are abolished. After that, the re-

[1] http://ncode.limsi.fr

mained audio used for training consists of around 175 hours of speech, distributed among nearly $175K$ utterances.

The lexicon was built based on the Carnegie Mellon University (CMU) Pronouncing Dictionary v0.7a, in which the phoneme set contains 39 phonemes and the word set contains 131,137 words. The vowels may also vary in lexical stress, ranging from no stress, primary stress to secondary stress.

2.1.2. Front-end

The front-end of the system is based on the conventional Mel-frequency cepstral coefficients (MFCC) features. The initial feature vectors, which contain 39 coefficients including 12 cepstral coefficients, 1 energy coefficient added with delta and double-delta features were extracted after windowing with the window size of 25 milliseconds and frame shift of 10 milliseconds. After that, Ceptral Mean and Variance Normalization (CMVN) was applied for normalization.

2.1.3. Training Procedure

The acoustic models were based on Hidden Markov Model (HMM), using Gaussian Mixture Models (GMM) for emission probabilities. In order to model context dependency, we used the tri-phone setup, with three states per phoneme and the topology was left-to-right. The model was trained with the expectation-maximization (EM) algorithm with a splitting procedure according to the maximum likelihood criterion. After splitting, the total number of gaussians, which are initiated at 2000, reached 200000. Furthermore, Maximum Likelihood Linear Regression (MLLR) technique was used to adapt the acoustic models with speaker information, for which we assumed that each TED talk in the training data is corresponding to one speaker.

Figure 1 reveals that we developed the systems in two directions after the baseline. On one hand, the acoustic model was strengthened throughout further training with subspace GMM, which was proved to significantly increase the system performance [5]. The SGMM model was then enhanced with discriminative training, producing the SGMM-MMI system. On the other hand, feature space discriminative training was implemented on top of the baseline system, to create the fMMI system. In order to display the progressive result, the error rates on dev2010 and tst2010 data are illustrated in Table 1. It is notable that the language model used in the experiments is the 3-gram LM described in Section 2.3. The SGMM was able to improve the performance of our system by 8% relatively, while discriminative training on top of the SGMM system showed its effectiveness by reducing the error rates by 13%. Feature space MMI training over the baseline system was efficient enough to reduce 18% of errors relatively. In brief, the SGMM+MMI training on top of the baseline system was slightly better than the counterpart trained with fMMI.

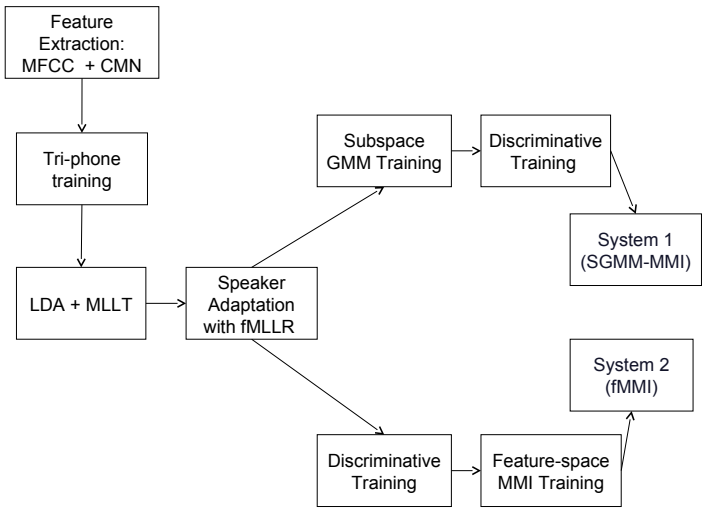

Figure 1: *Training Procedure diagram*

Table 1: *Progressive results shown by consecutively trained systems*

System	WER	
	dev2010	tst2010
MFCC+LDA+SAT (baseline)	26.6	26.4
baseline+SGMM	24.9	24.2
baseline+SGMM+MMI	21.8	21.1
baseline+fMMI	21.9	21.6

2.2. Auto-segmentation

Since the evaluation data in 2013 is no longer provided with timing information for segmentation, we utilize the LIUM Diarization toolkit [6] in order to divide the talk into small sentence-like segments,

Figure 2 provides a general description on the diarization process. First, 13 MFCC features are extracted from the long audio file. Subsequently, the long talk is segmented based on Viterbi Decoding, producing shorter segments which are at least 20 seconds long. After that, 8 one-state HMMs are used to remove music and jingle regions, leaving only speech segments. Detection of gender and bandwidth is then done using a GMM for each of the 4 combinations of gender (male / female) and bandwidth (narrow / wide band). Finally, GMM-based speaker clustering is carried out to map each speech segment to the corresponding speaker. Apparently, one TED talk can be given by only one or several speakers.

The disparity in word error rates is disclosed in Table 4, in Section 2.4. It is notable that the automatic speech detection caused approximately 2 percent loss of the spoken audio, resulted in inevitably decreasing the error rates, presented by deletions. Experiments conducted with tst2010 and dev2010 data illustrated that the WER increased 10% relatively, compared with the same data sets which are manually segmented.

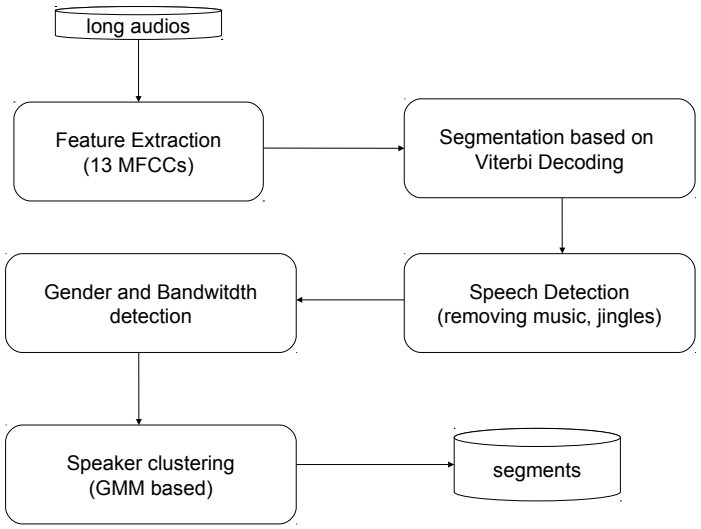

Figure 2: *Diarization Process.*

Due to the fact that the segmentation cannot be guaranteed to be precise at the beginning (end) of the sentence, the output segments are almost incomplete sentence, or incomplete phrases, which affects recognition results. The influence of language models on this problem will be analyzed later in Section 2.3.2.

2.3. Language Modeling

2.3.1. Overview

We used the in-domain data provided by organizer. In addition, we utilize $\frac{1}{8}$ of Giga corpus by filtering it according to the Moore-Lewis approach [7]. Both two datasets were normalized using the normalization toolkit from CMU[2]. The statistics of training data is summarized in Table 2. The vocabulary used to train language models is the same as in the lexicon. It consists of 131,137 words.

Table 2: *Training data for language modeling for English ASR Task*

Data	Number of sentences	Number of tokens
TED	156,460	2,708,816
$\frac{1}{8}$ Giga	2,565,687	56,488,064

The final model is the combination of two models trained on these datasets using SRILM toolkit with the modified interpolated Knesey-Ney smoothing technique [8].

Simultaneously, we trained SOUL language models on the same training data following exactly the procedure described in [9]. We use 300 as the dimension projection, $600; 300$ as the size of 2 hidden layers and $1000; 1000$ as the size of the shortlist and the number of classes for the out-of-shortlist words. For each type of SOUL models presented below, only one model is trained and used while decoding.

[2]http://www.festvox.org/nsw/

2.3.2. Auto-segmentation and sentence boundary problem

As auto-segmentation presented in Section 2.2 is based solely on acoustic features, each resulting segmentations does not correspond to a "normal" sentence but rather a phrase. For example, in dev2010, the audio for this sentence:

Now there are many of us who sort of forget that when I say...

is segmented into three parts corresponding to:

Now
there are many of us who sort of
forget that when I say...

If we train language models on data containing normal sentences, there will be a mismatch between test data and training data. The question is to what extent this mismatch affects the final performance. To partially answer this question, we proposed to use three types of SOUL models that differ in the way of treating sentence boundary. The detailed explanation of each model will be presented as follows:

Standard model Supposing that we use 4-gram language models and have a couple of sentences in a document:

Music can be the food of love
Let's do this

In the traditional way, the probability of the second sentence is:

$$p(\text{Let's}|\text{<s> <s> <s>}).p(\text{do}|\text{ <s> <s> Let's}).$$
$$p(\text{this}|\text{<s> Let's do}).p(\text{</s>}|\text{Let's do this}), \quad (1)$$

where <s>, </s> stand for the start (end) of the sentence. <s> is repeated at the beginning of the sentence to better represent the context in SOUL structure because the number of input tokens of SOUL is fixed to 3. So, sentence boundary is introduced by using these two special tokens. Each sentence in the document is independent which means that there is no information between consecutive sentences that is taken into account. This type of SOUL model is call "standard".

Cross model If we assume that there does not exist any negligible information between sentences, we can still follow an n-gram approach by considering the whole document as one long sentence and using </s> to mark sentence boundary. The probability of the second sentence turns out to be as follows:

$$p(\text{Let's}|\text{of love </s>}).p(\text{do}|\text{love </s> Let's}).$$
$$p(\text{this}|\text{</s> Let's do}).p(\text{</s>}|\text{Let's do this}), \quad (2)$$

By doing this, we obtained the "cross" SOUL model. Theoretically, by increasing the order n, the model could take almost all words of the previous sentences into the context to predict words in the current sentence. Note that, there exists other ways to take all previous words into account, such as a "cache" maximum entropy language model [10], a recurrent neural network language model (RNNLM) [11].

Intuitively, it is evident that the information between sentences in the document is helpful. However, in practice, it is often difficult to take this type of information into account to improve the system performance, especially on large scale tasks. Conclusions for the literature for this problem are mixed at best. In [12], RNNLM was shown to work better than any other methods including n-gram NNLM. However, it is unclear that RNNLM is more efficient due to the difference in structure of the two models, or the capacity of RNNLM to take into account a long-range dependency between words (possible to be in different sentences), or both. Measuring the influence between words was once implemented in [13]. In this article, a recurrent SOUL model is shown to work only on par with a standard 10-gram SOUL models on a large scale WMT English to French translation task. The problem for this comparison is that two types of models don't have the same architecture.

For this reason, we used the same n-gram SOUL structure with large n (10) to investigate whether the words in previous sentences which have the distance to the predicted word not further than 9 is helpful in prediction.

Cross-wo-boundary model Both standard and cross SOUL models could not deal with the mismatch between training and test data. To clarify, supposing that after employing auto-segmentation, we have two phrases:

Music can be the food
of love Let's do this

The probability of the new second sentence estimated by a cross SOUL model becomes:

$$\begin{aligned} &p(\text{of}|\text{the food } \text{</s>}).p(\text{love}|\text{food } \text{</s>} \text{ of}).\\ &p(\text{Let's}|\text{<s>} \text{ of love}).p(\text{do}|\text{of love Let's}).\\ &p(\text{this}|\text{love Let's do}).p(\text{</s>}|\text{Let's do this}) \end{aligned} \quad (3)$$

Compared to Equation (2), the sentence boundary is moved two positions to the left. It leads to poor probability estimation because typically the training data do not have any sentence boundary placed in similar position.

One solution is to carry out the same auto-segmentation procedure with the acoustic training data, then using the corresponding transcriptions of the resulting audio segmentation as the training data. In this case, the training data and test data are guaranteed to be drawn from the same distribution, i.e., no mismatch exists. But now the training data does not contain "real" sentences but rather phrases. The main problem is that since the audio is required, the size of the training data for language modeling is restricted. Moreover, this solution hinders the use of out-of-domain data because there is now the mismatch between in-domain data having the associated audio from the same source as the test data and out-of-domain data often composed of "real" sentences.

Another solution is to completely ignore the sentence boundary, so the probability of the second sentence becomes:

$$\begin{aligned} &p(\text{of}|\text{be the food }).p(\text{love}|\text{the food of}).\\ &p(\text{Let's}|\text{food of love}).p(\text{do}|\text{of love Let's}).\\ &p(\text{this}|\text{love Let's do}) \end{aligned} \quad (4)$$

The underlying idea is simple: Since there is no trivial solution for detecting sentence boundary when testing, we completely ignore it in the training phase to guarantee the homogeneity between the training and test data. In equations, sentence boundary is not in the context neither in the predicted position. So we have a "cross-wo-boundary" model. It is worth noting that three types of SOUL models presented above have the same architecture. They differ only in the way of constructing the context, see Table 3 for an example about the probability of the word "of".

Table 3: *Example for three types of SOUL models*

SOUL	probability
standard	$p(\text{of}\|\text{<s> <s> <s>})$
cross	$p(\text{of}\|\text{the food </s>})$
cross-wo-boundary	$p(\text{of}\|\text{be the food})$

In Section 2.5, these three types of SOUL models will be compared experimentally in both cases where long audio signals are automatically segmented into phrases or where they are segmented manually into sentences.

2.4. Decoding Procedure

As can be seen from Figure 3, there are three main phrases constituting the decoding process. The first phase begins with the feature extraction step, followed by decoding with the baseline system (MFCC+LDA+SAT) in order to estimate the transformations for speaker adaptation (fMLLR algorithm). In the second phase, Viterbi decoding is conducted with the SGMM-bMMI model and the fMMI model separately, with fMLLR adaption using the pre-estimated transformations, resulted in one set of lattice for each system. These two lattice sets are then re-scored with the 4-gram language model. Afterward, system combination is carried out to reduce the error rates from both above systems, by exploiting lattice interpolation. In our experiment, the two systems are equally treated, by setting their lattice weights to 0.5.

The last phase is where our NNLM is applied for rescoring N-best results from the lattices. Specifically, each lattice is decoded for 1000 best outputs, in which the best output is chosen based on NNLM rescoring. To do N-best rescoring with SOUL, we follow the same scheme for RNNLM provided by Kaldi [1], i.e., we adapt related scripts for SOUL.

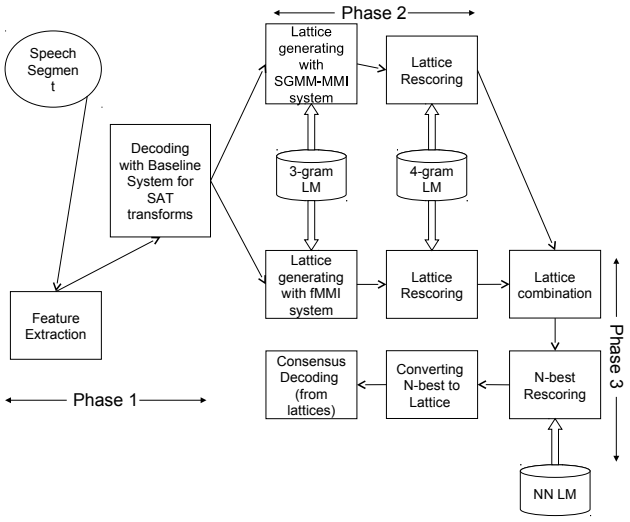

Figure 3: *Decoding diagram.*

Basically, it is done as follows. First, N-best is extracted from lattices. Then the probability estimated by SOUL models for each sentence is computed. Language model scores are updated as the interpolation of the scores provided by the back-off language model and the SOUL model. The coefficient is optimized on the development data. After that, N-best is converted back into lattices. Finally, any standard decoding method can be employed on the output lattices to have final results. In our case, consensus decoding is used at this step.

So the scripts we need to modify is for using SOUL models to compute probability for each line of a text file and combining scores of language models. For the first task, it is one of basic inference functions of standard SOUL models which can be done efficiently by using several speed-up techniques such as multi-threading, context grouping...[3]. Therefore, the computational time of N-best rescoring phase is dominated by the other steps concerning N-best extraction and lattice construction. In case of cross or cross-wo-boundary models, the computational time is similar. The only difference is that we need to use words from previous sentences while we don't have true previous sentences but their best lists. For simplification, we decide to use the best hypotheses of previous sentences provided by original lattices to predict words in a current sentence.

For the second task, it is in fact straightforward to use the script provided by Kaldi where for each sentence, a final score is the weighted average of its scores estimated by two language models. However, the interpolation in this way is only at sentence level while a (more) traditional way is to interpolate models at word level, i.e., for each word, its probability is computed as a combination of scores provided by language models. Therefore, we add scripts in order to compare these two interpolation fashions.

[3]On lattices tst2013 of $\approx$ 33 million n-grams, it costs around 4 minutes on Intel(R) Core(TM) i7-3770K CPU with Intel(R) Math Kernel Library.

2.5. Experimental results

Table 4 shows the experimental results with the three final systems. The combination technique allows us to reduce slightly the WER, by around 3% which is identical in the case of rescoring the lattice with the 4-gram language model. Besides, it is clear that auto-segmentation and speech detection exacerbated the systems' performance, by increasing the WER by 10% relatively. As mentioned above, the speech detection inevitably ignores 2% of spoken data, leading to uncompensated deletions in recognition.

Table 4: *ASR results for various acoustic models and segmentation types (manual, auto)*

System	WER			
	dev2010		tst2010	
	manual	auto	manual	auto
SGMM+MMI+4gram(1)	21.6	23.6	20.9	23.4
fMMI+4gram(2)	21.4	23.0	21.3	23.8
combine(1+2)	20.8	22.2	20.0	22.5

Table 5: *ASR results for different types of SOUL models*

System	WER			
	dev2010		tst2010	
	manual	auto	manual	auto
combine(1+2)	20.8	22.2	20.0	22.5
+ standard SOUL (inter)	18.8	20.5	18.1	20.9
+ standard SOUL	18.9	20.4	18.1	20.6
+ cross-wo-boundary SOUL	18.9	20.1	18.6	20.6
+ cross SOUL	19.0	20.4	18.4	20.8

Table 6: *Official results for English ASR task. Note that, results in tst2013 column is with auto-segmentation*

System	WER		
	tst2011	tst2012	tst2013
combine(1+2)	16.8	18.5	30.0
+ standard SOUL	14.6	16.2	27.4

In Table 5, we summarize WER results for different types of SOUL models which are used in N-best rescoring. There are some remarks drawn from these results. First, interpolation at sentence level is slightly better than at n-gram level. It supports the idea that two types of language models (back-off, SOUL) have different characteristics, so it is better to combine them at sentence level.

Second, cross model under-performs significantly standard model in both cases (manual and auto). It means that within the SOUL structure, taking into account words of previous sentences seems to be harmful rather than useful.

Third, concerning manual segmentation, the cross-wo-boundary model performed worst than the standard model. It shows that to predict a word, while words in previous sentences seems unnecessary, the role of sentence boundary is undeniable. On the contrary, in the case of auto segmentation, as the sentence boundaries for test data are not reliable, cross-wo-boundary model can potentially bring benefit. The experiments with development data showed this improvement, but unfortunately the improvement is not carried over test data. There are several possible reasons behind this phenomenon. First, we used only the best original hypotheses of the previous sentences to predict the words in the context. Second, the automatic segmentation caused the high rate of word deletion so the continuity of segmentations is not guaranteed.

Finally, all types of SOUL models bring significant improvements over the baseline system. As seen in Table 6, on all test data (tst2011, tst2012, tst2013), the standard SOUL model achieves improvements of about 10% relatively. Note that, the achievements could be more considerable if we use more than one SOUL model for N-best rescoring.

3. Machine Translation Task

In this section, we present our system used for the English to French Machine Translation task. The baseline system is based on the bilingual n-gram approach for Statistical MT [14, 15, 16]. This system is then enhanced with a SOUL language model [2]. The experimental evaluation shows that the system achieves competitive results, therefore it can be served as a baseline system for our further research.

3.1. Data setup and preprocessing

We used the training TED data provided by the campaign [17] and several datasets from the evaluation campaign of Workshop for Machine Translation (WMT) 2013[4]. We don't use Common-Crawl or any data from LDC. Considering the TED data as the in-domain data, half of the parallel dataset Giga is filtered out by applying a technique described in [7] on the French side. Note that, in our configuration, we use tst2010 as the development set and dev2010 as the test set. The reason behind this substitution is simple: We want to have more sentences in the development set than in the test set. This development data is used in the optimization procedure for the log-linear framework as well as optimizing other hyper-parameters such as the interpolation weights for language modeling, data filtering... The (internal-)test data is used to choose the best system for evaluation. The final parallel data consist of TED, NewsCommentary, Europarl and $\frac{1}{2}$ Giga. The monolingual data contain TED, News2008-2012, Europarl, Giga, UN for a total of $58{,}793{,}286$ sentences and $1{,}744{,}768{,}777$ tokens.

The preprocessing step was done as follows. As data sets are obtained from several sources, notably Internet. In order to have a clean and homogeneous data in terms of format, we decided to delete unnecessary characters, especially malform unicode ones, then converted texts into standard precomposed unicode format. We treated cases as is.

For the English side, we followed Penn Treebank style and used the script provided by Penn[5]. As we need Part-Of-Speech (POS) tags on the source side, we use TreeTagger [18] toolkit applied on the tokenized data. For the French side, the tokenization process was done by using BonSai toolkit well adapted for French[6] [19]. It separates common French phrases such as "donnez-le-nous" into three words: "donnez -le -nous". Another point of this process is that it matches compounds in a text, then replacing the space that separates the components by a "_". Compounds were taken from a built-in list, which contains phrases such as "a fortiori", "au lieu de", " partir de" ... As there is not any available scripts in Bonsai toolkit to convert tokenized texts back to original texts, we implement that task ourselves by breaking out compound words and then applying detokenization.

3.2. System overview

We used N-code to build a baseline system, hence following exactly the bilingual n-gram approach described in [14, 15, 16]. Note that, the baseline system construction is very similar to the one used in [20]. To build a translation model, word alignments were first obtained by carrying out MGIZA++[21]. Based on the information from word alignment, words in each source sentence were reordered to match the word order in its target sentence. Tuples were defined as basic translation units containing source and target phrases. Each pair of sentences was considered as a sequence of tuples. For each pair, there were maybe more than one possible sequence. Therefore, some conditions are added [15] to guarantee that there is a unique sequence of tuples which can be associated to a pair of sentences. The most important condition is that each tuple in the sequence cannot be divided into small tuples. After that, translation models are n-gram models that estimate the probability of a sequence of tuples.

When inference, translation was broken into two steps: a source reordering step and a translation step. In the source reordering step, a source sentence was represented in the form of word lattices which contains the most likely reordering hypotheses. These hypotheses were obtained by applying rewrite rules learned from word alignments and Part-of-Speech (POS) taggers of the source side. It has been shown in [16] that learning rules from POS tagger has a better generalization. In the translation step, all hypotheses in the lattice were translated monotonically using the log-linear framework.

The baseline system is the combination of four translation models based on lexicalized weighting and relative frequency (4 features), a monotone-swap-forward-backward (MSFB) lexicalized reordering model [22, 23] (8 features), a

[4] http://www.statmt.org/wmt13/translation-task.html

[5] http://www.cis.upenn.edu/ treebank/tokenizer.sed

[6] http://alpage.inria.fr/statgram/frdep/fr_stat_dep_malt.html

word bonus (1 feature), a tuple bonus (1 feature), a "weak" distance-based distortion model (1 feature), four 3-gram translation models trained on TED, NewsCommentary, Europarl and $\frac{1}{2}$ Giga (4 features) and a 4-gram language model (1 feature). It results in 20 feature functions combined in the log-linear framework. Their optimization weights are obtained by employing the MERT procedure [24]. N-gram translation models and language models are trained using SRILM toolkit with the modified interpolated Knesey-Ney smoothing technique [8].

For language modeling, the vocabulary contains 500,156 most frequent words, which occur more than 15 times in the training data. The back-off language model is the interpolation of nine 4-gram sub-models trained on each training dataset with weights optimized on the development data. The perplexity of the final model computed on the test data is 74.

We trained a 10-gram SOUL model on the same training data following exactly the procedure described in [9]. We used 500 as the dimension projection, $1000; 500$ as the size of 2 hidden layers and $2000; 2000$ as the size of the shortlist and the number of classes for out-of-shortlist words. It achieves 59 as the perplexity on the test data (20% better than the back-off model).

Proved to be helpful [25, 26], SOUL language model was used on top of this system in the 300-best rescoring phase. For each hypothesis in the list, a score of the SOUL language model is computed, then being added as a new score. Weights of models are re-optimized following the MERT procedure on the development data.

3.3. Experimental results

MT systems are evaluated according to BLEU, NIST metrices computed by the script provided by NIST[7]. The results are summarized in Tables 7 and 8. Note that the results in Table 7 are computed on tokenized texts while the results in Table 8 are the official results provided by organizer. We see that 300-best rescoring with SOUL improves significantly the performance (≈ 1.4 BLEU points improvement).

Table 7: *Results for English to French MT task. Scores are case-sensitive with tokenized texts*

Systems	Scores			
	dev data		test data	
	BLEU	NIST	BLEU	NIST
baseline	34.9	7.55	28.6	6.60
+ rescoring with SOUL	35.8	7.69	29.7	6.73

4. Conclusion

In this paper, our systems served for the English ASR and English to French MT tasks of IWSLT2013 were presented in detail. On the ASR task, by comparing three types of SOUL language models distinguished in the way of treating sentence boundary, we found that in the SOUL structure, taking into account words of previous sentences were not effective, even in the case of auto-segmentation. In both tasks, the SOUL models were used on top in N-best rescoring phase. They were proved to improve significantly the system performance with approximately 10% relative WER reduction for ASR task and an addition of about 1.4 BLEU points for MT task.

Table 8: *Official results for English to French MT task. BLEU is case-sensitive*

Systems	BLEU		
	tst2011	tst2012	tst2013
baseline	37.1	38.6	36.2
+ rescoring with SOUL	38.8	39.9	37.6

This work was partially supported by National ICT Project KC.01.03/11-15

5. References

[1] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[2] H.-S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Structured Output Layer neural network language model," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, may 2011, pp. 5524 –5527.

[3] A. Katsamanis, M. Black, P. G. Georgiou, L. Goldstein, and S. S. Narayanan, "SailAlign: Robust long speech-text alignment," in *Proc. of Workshop on New Tools and Methods for Very-Large Scale Phonetics Research*, Jan. 2011.

[4] H. Yamamoto, Y. Wu, C.-L. Huang, X. Lu, P. R. Dixon, S. Matsuda, C. Hori, and H. Kashioka, "The NICT ASR system for IWSLT 2012," in *International Workshop on Spoken Language Translation (IWSLT)*, Hong Kong, 12 2012.

[5] D. Povey, L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafit, A. Rastrow, R. C. Rose, P. Schwarz, and S. Thomas, "The subspace Gaussian mixture model - A structured model for speech recognition," *Computer Speech & Language*, vol. 25, no. 2, pp. 404 – 439, 2011.

[6] S. Meignier and T. Merlin, "LIUM SpkDiarization: an

[7] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v13a.pl

open source toolkit for diarization," in *CMU SPUD Workshop*, Dallas (Texas, USA), mars 2010.

[7] R. C. Moore and W. Lewis, "Intelligent Selection of Language Model Training Data," in *Proceedings of the ACL 2010 Conference Short Papers*. Uppsala, Sweden: Association for Computational Linguistics, July 2010, pp. 220–224.

[8] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Technical Report TR-10-98*. Cambridge, Massachusetts, USA: Computer Science Group, Harvard University, 1998.

[9] H.-S. Le, I. Oparin, A. Messaoudi, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Large Vocabulary SOUL Neural Network Language Models," in *Proceedings of the 12th Annual Conference of the INTERSPEECH 2011*, Florence, Italy, 2011.

[10] R. Rosenfeld, "Adaptive Statistical Language Modeling: A Maximum Entropy Approach," Ph.D. dissertation, Carnegie Mellon University, 1994.

[11] T. Mikolov, M. Karafit, L. Burget, J. ernock, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of the 11th Annual Conference of the INTERSPEECH 2010*, vol. 2010, no. 9, 2010, pp. 1045–1048.

[12] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocký, "Empirical Evaluation and Combination of Advanced Language Modeling Techniques," in *Proceedings of the 12th Annual Conference of the INTERSPEECH 2011*, 2011, pp. 605–608.

[13] H.-S. Le, A. Allauzen, and F. Yvon, "Measuring the Influence of Long Range Dependencies with Neural Network Language Models," in *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*. Montréal, Canada: Association for Computational Linguistics, June 2012, pp. 1–10.

[14] F. Casacuberta and E. Vidal, "Machine Translation with Inferred Stochastic Finite-State Transducers," *Comput. Linguist.*, vol. 30, no. 2, pp. 205–225, June 2004.

[15] J. B. Mariño, R. E. Banchs, J. M. Crego, A. de Gispert, P. Lambert, J. A. R. Fonollosa, and M. R. Costa-jussà, "N-gram-based Machine Translation," *Comput. Linguist.*, vol. 32, no. 4, pp. 527–549, Dec. 2006.

[16] J. M. Crego and J. B. Mariño, "Improving statistical MT by coupling reordering and decoding," *Machine Translation*, vol. 20, no. 3, pp. 199–215, Sept. 2006.

[17] M. Cettolo, C. Girardi, and M. Federico, "WIT^3: Web Inventory of Transcribed and Translated Talks," in *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, Trento, Italy, May 2012, pp. 261–268.

[18] H. Schmid, "Probabilistic Part-of-Speech Tagging Using Decision Trees," in *Proceedings of International Conference on New Methods in Language Processing*, 1994.

[19] M. Candito, J. Nivre, P. Denis, and E. Henestroza Anguiano, "Benchmarking of Statistical Dependency Parsers for French," in *Coling 2010: Posters*. Beijing, China: Coling 2010 Organizing Committee, August 2010, pp. 108–116.

[20] A. Allauzen, N. Pcheux, Q. K. Do, M. Dinarelli, T. Lavergne, A. Max, H.-S. Le, and F. Yvon, "LIMSI @ WMT13," in *Proceedings of the Eighth Workshop on Statistical Machine Translation*, Sofia, Bulgaria, August 2013, pp. 62–69.

[21] Q. Gao and S. Vogel, "Parallel implementations of word alignment tool," in *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, ser. SETQA-NLP '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 49–57.

[22] C. Tillmann, "A unigram orientation model for statistical machine translation," in *Proceedings of HLT-NAACL 2004: Short Papers*, ser. HLT-NAACL-Short '04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004, pp. 101–104.

[23] J. M. Crego, F. Yvon, and J. B. Mario, "N-code: an open-source Bilingual N-gram SMT Toolkit," *Prague Bulletin of Mathematical Linguistics*, vol. 96, pp. 49–58, 2011.

[24] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ser. ACL '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 160–167.

[25] T. Lavergne, A. Allauzen, H.-S. Le, and F. Yvon, "LIMSIs experiments in domain adaptation for IWSLT11," in *Proceedings of the 8th International Workshop on Spoken Language Translation (IWSLT)*, San Francisco, CA, 2011.

[26] A. Allauzen, H. Bonneau-Maynard, H.-S. Le, A. Max, G. Wisniewski, F. Yvon, G. Adda, J. M. Crego, A. Lardilleux, T. Lavergne, and A. Sokolov, "LIMSI @ WMT11," in *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Edinburgh, Scotland: Association for Computational Linguistics, July 2011, pp. 309–315.