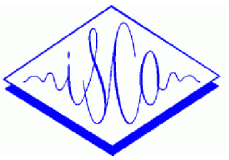
ISCA Archive
http://www.isca-speech.org/archive



International Workshop on
Spoken Language Translation 2011
San Francisco, CA, USA
December 8-9, 2011

Investigation on the Effects of ASR Tuning on Speech Translation Performance

Paul R. Dixon *Andrew Finch* *Chiori Hori* *Hideki Kashioka*

National Institute of Communication Technology
Kyoto, Japan
{paul.dixon,andrew.finch,chiori.hori,hideki.kashioka}@nict.go.jp

Abstract

In this paper we describe some of our recent investigations into ASR and SMT coupling issues from an ASR perspective. Our study was motivated by several areas: Firstly, to understand how standard ASR tuning procedures effect the SMT performance and whether it is safe to perform this tuning in isolation. Secondly, to investigate how vocabulary and segmentation mismatches between the ASR and SMT system effect the performance. Thirdly, to uncover any practical issues that arise when using a WFST based speech decoder for tight coupling as opposed to a more traditional tree-search decoding architecture.

On the IWSLT07 Japanese-English task we found that larger language model weights only helped the SMT performance when the ASR decoder was tuned in a sub-optimal manner. When we considered the performance with suitable wide beams that ensured the ASR accuracy had converged we observed the language model weight had little influence on the SMT BLEU scores.

After the construction of the phrase table the actual SMT vocabulary can be less than the training data vocabulary. By reducing the ASR lexicon to only cover the words the SMT system could accept, we found this lead to an increase in the ASR error rates, however the SMT BLEU scores were nearly unchanged. From a practical point of view this is a useful result as it means we can significantly reduce the memory footprint of the ASR system.

We also investigated coupling WFST based ASR to a simple WFST based translation decoder and found it was crucial to perform phrase table expansion to avoid OOV problems. For the WFST translation decoder we describe a semiring based approach for optimizing the log-linear weights.

1. Introduction

In this paper we describe our recent investigations into the coupling of a (Weighted Finite State Transducer) WFST based speech decoder with both standard Statistical Machine Translation (SMT) decoder and a simple WFST-based translation decoder.

Development of speech translation systems is a challenging problem. Not only is the construction of high quality models a challenging research problem, there is also the need to cascade the Automatic Speech Recognition (ASR) and Statistical Machine Translation (SMT) systems together in a complementary manner.

We investigated the relationship between the speech recognition performance in terms of both Word Error Rate (WER) and Real Time Factor (RTF) and the effects that various ASR tuning schemes had on the BLEU translation scores. One motivation of this work was to establish if certain ASR tuning practises could have a negative effect on the translation performance. Without meaningful and thoroughly tested baselines it can sometimes be difficult to analyse or interpret new techniques and results effectively.

1.1. Previous Work

Previous work on the coupling of ASR and SMT has looked at coupling issues and the effects of WER on SMT performance.

Some of the earliest work on coupling speech and translation used a joint source model [1] in the WFST framework. One advantage of this approach is the joint model can be use in place of the ASR language model. Other authors also investigated this approach and proposed extensions to allow for different lexical units or re-ordering either as second pass or via embedded information[2, 3, 4].

Later work has demonstrated that a phrase-based log-linear model offers better performance than the joint source approach. In addition it has been demonstrated that lattice coupling or confusion network coupling further improves performance [5, 6, 7]. To achieve optimal gains in this coupling scenario it is necessary to optimize the acoustic and source language model weights as part of the training process.

Recently in [8] the authors argued that WER was not the best criteria to optimize a speech translation system. They used a hierarchical SMT system that was connected to the n-best output of a speech decoder. They performed experiments where they varied the LM weight and observed large values of the LM weight gave better SMT performance but a reduced WER. Sarikaya [9] et al, observed that reducing the ASR WER correlated with an increased BLEU score, but the relations was not linear. Saon [10] wrote the better the WER the better the performance of the speech translation component.

2. WFST Based Speech Recognition

Recently the WFST based approach has become popular for constructing speech recognition decoders[11]. One of the advantages of the WFST framework is the manner in which we can represent each of the knowledge sources in a consistent manner. The knowledge sources can be composed together and optimized for speed and size ahead of decoding.

Often the recognition cascade is constructed from the following components; the Language Model(LM) G which represents the recognition grammar, the lexicon L which is built from the pronunciation dictionary and maps phoneme sequences to words, a transducer C that converts context-dependent phonemes to context-independent phonemes, and optionally the acoustic models H. In our decoder the recognition cascade is constructed according to:

$$(C \circ det(L)) \circ G$$

where det is the determinization operation, and $\circ$ is the composition operation. In our construction process C and $det(L)$ are composed together with a standard static composition. The composition with G makes use of lookahead-composition[12], and is either performed online or offline depending on the task complexity and performance constraints. In our decoder the expansion of context-dependant arcs to HMM state sequence is performed online inside the decoder.

3. Task and Model Training

For the evaluations we used the IWSLT07 Japanese/English training, development and test data [13]. The Japanese/English task is based around The Basic Travel Expression Corpus (BTEC) task. This is a multi-lingual corpus of tourism style phrases.

3.1. ASR Training

We used the original acoustic models, lexicon and training text that were used to generate the IWSLT 2007 speech recognition output. By using the same ASR models it allows us to compare our results with other system results submitted to the same IWSLT 2007 task.

The acoustic models were a pair of gender dependent models each with 5700 states and 10 diagonal Gaussians per state. The acoustic models were trained on feature vectors comprised of 12 MFCCs with their deltas, augmented with a delta power term.

The ASR LM training text consisted of 85k sentences and had a vocabulary of 59k words. We built a modified Kneser-Ney smoothed tri-gram using the MITLM toolkit[14]. Prior to training the LM the Japanese text was segmented using the same propriety tool as used to segment the IWSLT SMT training text. This was done to make coupling easier and remove the need for a possible re-segmentation after recognition.

Our in-house decoder *SprinTra* is a general one-pass Viterbi decoder. To parallel decode the gender dependent acoustic models we first constructed separate male and female lexicon and context-dependency transducers. These transducers were combined and optimized according to:

$$(C_F \circ det(L_F)) \cup (C_M \circ det(L_M)))$$

Where $\cup$ is the union operation and the M and F subscripts indicate the gender, the individual acoustic models were merged to form a single acoustic model and appropriate relabeling was applied. The final recognition cascade we used was:

$$((C_F \circ det(L_F)) \cup (C_M \circ det(L_M))) \circ G$$

The composition of the language model G was performed on-the-fly using lookahead composition[12]. This construction allows us to drive several search networks in parallel with the general decoder in a memory efficient manner. This choice of construction is the same technique we use in our production systems. Although, it is possible to construct individual male and female CLG cascades, decode them separately in parallel and then select the best result. In practise we find the union system gives a less than 20% increase in CPU cost in comparison to a system with a single acoustic model. Furthermore, we can make additional memory saving by only having to load one copy of the language model.

This highlights one of the advantages of the WFST framework for speech recognition. The search network was modified to allow parallel decoding, whilst requiring no code changes to the decoder core itself. This parallel decoding scheme could easily be extended to various models for different tasks or environments, or even multiple languages as described in [15].

3.2. SMT Training

The SMT system was based around the Moses training and decoding components [16]. The SMT systems were all trained on the standard IWSLT 2007 training data which consisted of 20k training sentence pairs. The phrase tables were all built using the standard Moses training scripts. In all cases the English text was lower-cased and tokenized using the standard Moses scripts prior to training. After SMT decoding the casing was recovered prior to evaluation.

To build the target LM we used the mono-lingual component of the IWSLT training text. Again a modified Kneser-Ney trigram was built using the MITLM toolkit. We observed no improvement in translation performance when using higher order n-gram language models on this task.

The log-linear weights in the baseline Moses system were optimized using the standard Moses MERT script `mert-moses.pl`. The development data was 2k sentence pairs each with a single target translations from the IWSLT 2007 data.

4. Experiments

4.1. Matching the ASR and SMT systems

Our baseline system consists of a speech recognition decoder that generates a single best recognition hypothesis which is then used as the input to the translation decoder.

4.1.1. Punctuation

We first used the clean test-set as SMT input and compared the performance when training the SMT system with punctuation, without punctuation, and with punctuation only on the target side. After MERT optimizing the weights of each system, we found that punctuation on both sides gave the best performance, followed by target side only punctuation. Removing punctuation from both sides gave the worst performance. Based on these results we trained with punctuation marks present on both the source and target sides and later considered techniques to add punctuation to the ASR output. The average length of the English and Japanese training sentences was 7.6 and 9.15 words per sentence respectively. Japanese sentences were slightly longer because the text segmenter outputs morpheme like units which are smaller than words. The average length of the Japanese test reference transcriptions was 7.2 words per sentence.

4.1.2. Segmentation

We investigated how matching the segmentation and vocabulary units of the ASR and SMT components changed the translation performance. Japanese is normally written without whitespaces separating the words and this gives an even greater possibility of mismatch between the ASR and SMT components if the systems are trained on different segmentations. Furthermore, Japanese uses four scripts simultaneously and permits these scripts to be interchanged when writing words or even parts of words.

Often when constructing a Japanese ASR system the training text is segmented with a Part-of-Speech (POS) tagger, this will output the base form of each word along with the most likely pronunciation and POS tags. In Japanese a base form can have multiple pronunciations and possibly meanings based on the context. In Japanese ASR it is very common to construct a tuple from all of these fields and consider the tuple as the *word*.

Data sparseness issue may occur when including the POS and pronunciation as part of the words, however, under the WFST framework there is a slight practical advantage. The tuple form of a vocabulary entries will reduce or remove the amount of non-determinism on the output side of the lexicon transducer. This is because for each lexical entry there is only one possible pronunciation, and therefore in the lexicon transducer at any state there is at most one transition with a given output symbol. In practise we find for Japanese this helps to reduce the amount of memory required to perform the determinzation of $L \circ G$. In general we have found that adding the POS and pronunciations tags to the final LM and lexicon entries often leads to smaller search networks that are faster to decode. Although recent progress in composition and on-the-fly algorithms [12] have helped to remove the expensive $det(L \circ G)$ step.

For the IWSLT task we did not observe any substantial difference in ASR performance with or without the tuple form of the vocabulary entries. Although, for a large vocabulary Japanese spontaneous speech task we have observed significantly larger search networks and slower decoding speeds by removing the POS tags. A problem with stripping POS tags from the lattice and applying optimizations is it may increase the chances that the lattice cannot be determinized.

For future work in this area we plan to investigate the use of POS tags and pronunciation data in a factored SMT based system, or to extend the Lexicographic semiring idea as proposed by Roark et al [17].

4.1.3. Punctuation Recovery

In the first experiments we looked a several simple heuristics to add punctuation. The silences in the recognition output are either removed or mapped to periods or commas. We found that removing all silences and simply adding a final period to the recognition output performed best. On inspection of the Japanese test-set we found that the only types of punctuation present were final periods.

4.2. Decoder Tuning and Translation Performance

In this section we considered how the SMT performed as we changed the speech decoder parameters. We were not only interested in the relationship between WER and BLEU scores, but if certain sub-optimal WER values could give comparable BLEU scores at faster decoding speeds.

The WFST decoder only has a few search parameters and this makes tuning simple. There is a main search beam that prunes hypotheses with scores worse than the best score plus a threshold, this parameter controls the balance between speed and errors. The other parameter we changed was the LM weight which balances the score contributions from the acoustic and language models. In speech recognition these parameters are normally tuned to minimize the WER under some possible speed constraints.

In these experiments we fixed the band at 10000 hypotheses and looked at the change in SMT and ASR performance for a set of LM weights across various beam widths. The LM weight was varied between 12 to 20 in increments of 2, this choice was based on the following: In ASR we often find weight values of 12-15 give the best balance of RTF and WER, however in [8] the authors reported more favourable BLEU score for larger LM weights.

Figure 1 shows the RTF vs WER as we changed the speech recognition decoder parameters. We observed smaller language model weights values give more favourably ASR performance curves.

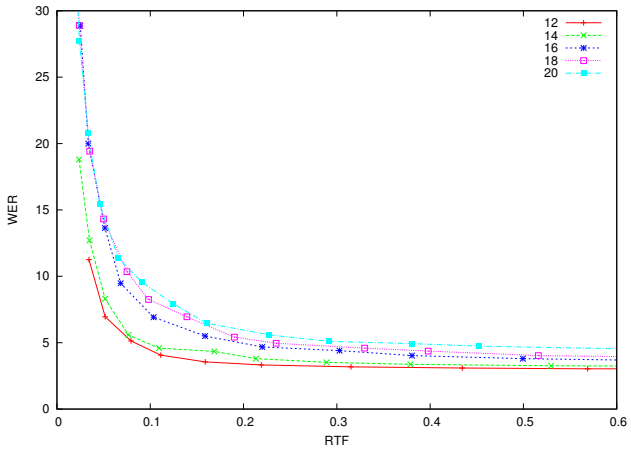


Figure 1: ASR WER vs RTF across a set of different language model weights.

Figure 2 shows the SMT BLEU vs WER for each LM weight. For any given beam there is almost a linear relationship between the WER and BLEU score. Reducing the error rate improves the translation. The results seem to agree with findings from [8], where larger LM weights appear to give better SMT performance but worse ASR performance. However, we also observed the SMT system achieved the same asymptotic accuracy regardless of LM weight width as long as the beam used in the ASR decoder was wide enough.

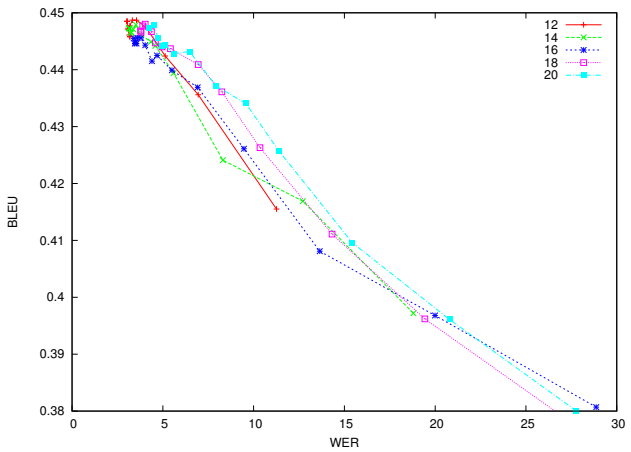


Figure 2: SMT BLEU score vs ASR WER across a set of different language model weights.

From figure 1 we see a smaller LM weights lead to faster convergence of the ASR performance. In figure 3 we plotted the RTF of the ASR decoder and the SMT BLEU scores. What is interesting is how close the curves are and for a given RTF the BLEU scores become closer regardless of the LM weight. In fact the smaller LM weights give better WER and BLEU scores and faster ASR decoding speeds. If we were tuning for tight computational constraints it makes a lot more sense to pick a smaller value for the LM weight. Reducing the value below 12 or above 20 gave poorer performance characteristics. Further investigations are needed to ascertain whether this result is specific to this task.

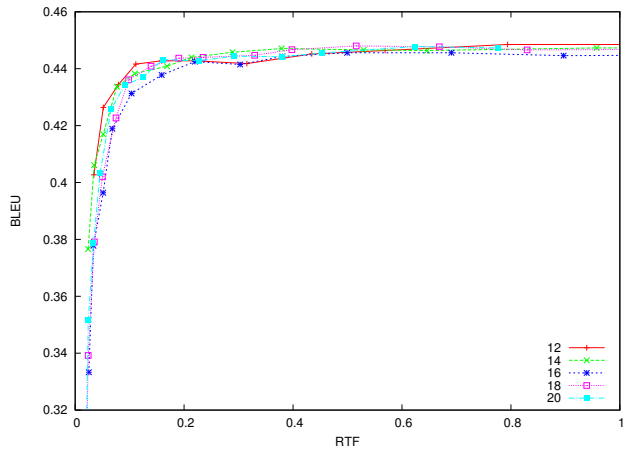


Figure 3: MT BLEU vs ASR RTF for a set of different language model weights

4.3. ASR and SMT Vocabulary Matching

There was a mismatch between the vocabulary used for ASR and SMT training. The presence of the Out-of-Vocabulary (OOV) words in the SMT output has a negative impact on the readability and naturalness of the translation quality. Here, we investigated whether matching the vocabularies of both systems would improve the SMT BLEU scores.

The ASR system training vocabulary was comprised of approximately 59k words, whilst the SMT system had only 12.5k unique source words. The ASR vocabulary was essentially the superset of SMT vocabulary. Only a small number of words (approx. 50) were only present in the SMT source vocabulary.

In a phrase-based decoder even when matching the ASR and SMT vocabularies there is no guarantee certain source words won't appear in the output like an OOV. In the training process the phrase table construction is not guaranteed to generate a phrase for every single source word and therefore certain source words will only appear in larger phrases. If these words are encountered in a different context and the decoder is unable to handle them and they will be propagated through to the output in the same way as a true OOV. To counter this we added one-to-one word pairs to allow every individual word to be translated. Our approach is simpler than the scheme described in[18], we just look for the best possible word alignments from the training data and add these to the phrase table using a very small flooring probability.

Moreover, in practise the SMT vocabulary can become even smaller. After phrase table construction we found the SMT vocabulary only had 8.5k unique words.

Any ASR vocabulary item that did not appear in the SMT

training text was mapped to an unknown word token before training the ASR language model.

Figure 4 shows the SMT BLEU vs WER for each LM weight under matched vocabulary conditions. When matching the vocabularies we first noticed a large increase in the error rate the speech recognition system. When restricting the ASR vocabulary the best WER was 5.5 versus 3.03 for the unrestricted case. However, there was a smaller reduction in SMT performance, reducing from 0.4490 in the unmatched conditions to 0.4452 for matched vocabularies.

From a practical point of view limiting the ASR vocabulary gave no speed gains for the speech decoding phase. However, we compared the sizes of the full composed static search network and observed an approximately 40% reduction in ASR memory requirements. These size reductions would be beneficial if we need to deploy the system to a smaller scale device such as a tablet or smartphone.

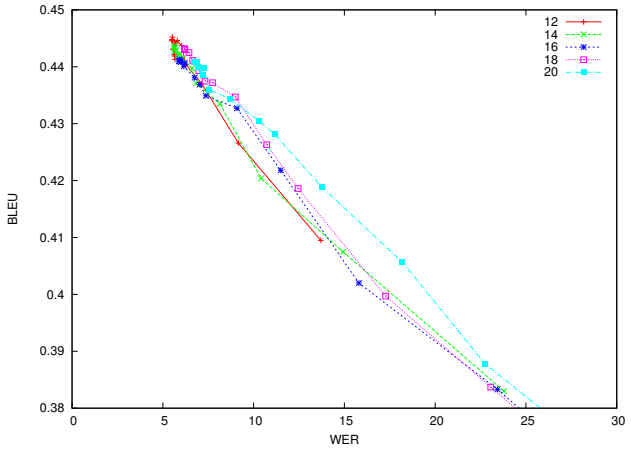


Figure 4: MT BLEU vs ASR WER for a set of different language model weights using a matched vocabulary.

5. WFST Lattice Based Interface

This section describes our initial investigations into the tighter coupling of speech recognition and translation systems. In particular we investigate the practical issues involved in coupling the lattice output of a WFST ASR system to a WFST based SMT. Our final goal is to have a complete end-to-end speech translation system implemented using WFSTs.

Lattice based coupling of ASR and SMT has been extensively studied and observed to improve performance. One of the most successful approaches is to use confusion networks [19] to interface the ASR output to the SMT component or lattice coupling based on confusion network information [7].

Using the lattices from a WFST based speech recognition decoder introduces a few problems when interpreting the costs and labels of the lattice arcs. This is due to a combination of the transducer optimizations and the structure of the component WFSTs used to construct the ASR cascade. In particular there are three specific issues which make the computation of the confidence scores problematic.

- The determinization operation will push the output labels forward towards the initial state of the WFST. Furthermore, the context-dependency transducer will introduce a delay between the input and output labels [20]. This means that we cannot interpret the output labels as corresponding to the actual word endings.
- The determinization and weight pushing algorithms will move the weights closer to the initial state of the WFST.
- Repeated lower order language model paths are introduced by the back-off approximation used in the WFST representation of the language model[21]. The lattices generated from the WFST decoder will have paths replicated that correspond to each of the lower order n-grams sequences.

We initially looked at converting the WFST phone lattices to word lattices or confusion networks and using them as the input to the Moses decoder. However, we found that the system did not perform well, possibly due to the previously mentioned issues. Given these issues, we decided to investigate the use of a WFST based translation decoder. When direct coupling WFST implementations the total path scores not individual label scores are important, and this will circumvent the previously mentioned problems.

5.1. WFST Based Machine Translation

The WFST framework has been extensively studied for machine and direct speech translation tasks. The approach we adopted is based on the method proposed in [22]. In the WFST decoder we used the following cascade of transducers $\lambda_1 M \circ \lambda_2 T \circ \lambda_3 N \lambda_4 \circ P \circ \lambda_5 G$. Where M is the source phrase segmentation transducer, T is the phrase table, N is the target phrase segmentation transducer, P is the insertion penalty and G is the target language model. All the transducers are assumed to be in the tropical semiring and λ_n parameters are the log-linear weights. In the following section we describe our approach to optimizing the log-linear weights λ_n in the WFST framework.

5.2. Optimization of WFST Feature Weights

The area we address in this section is the problem of tuning the log-linear weights within the WFST framework. The optimization of the feature weights requires that during decoding we maintain the individual feature contributions. However in the WFST framework after composition and optimization the contributions from each of the underlying knowledge sources is lost. One way to obtain the feature contributions is perform a re-alignment phase after decoding as described in [23], the drawback with this approach is the

need for a second decoding pass. An alternative approach is to access the internals of the composition process and try to recover the component state sequences from the state pairs the composition algorithm maintains internally . This approach would fail if during or after the decoding process we apply operations such as determinization or epsilon removal.

The method we describe uses a tuple semiring which can track individual score contributions from a cascade of WFSTs. This allows for the preservation of scores after any composition or even optimization algorithms.

A semiring is formally defined as the quintuple $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$, where $\mathbb{K}$ is a non-empty weight set, $\oplus$is a commutative operator with identity $\bar{0}$, and $\otimes$ is an addition operator with identity $\bar{1}$ [24].

The tropical semiring $(\mathbb{R} \cup \{-\infty, +\infty\}, min, +, \infty, 0)$ is example of a semiring that is often used in speech and language processing applications [24]. Here, the $\otimes$ operation corresponds to extending a path or hypothesis by adding a cost, and the $\oplus$ operation is used to compute the cost of paths passing through a state, in the tropical semiring this corresponds to taking a minimum under the Viterbi approximation.

For the purpose of tracking score contributions we used a semiring that is tuple of tropical weights. The length of the tuple n is the number of FSTs in the full cascade, including the input sequence or lattice that is represented as an acceptor. Each slot in the tuple corresponds to the score contribution of a component WFST.

Given a pair of weights: $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$. The $\otimes$ operator performs a component wise multiplication:

$$A \otimes B = (a_1 \otimes b_1, a_2 \otimes b_2, \ldots, a_n \otimes b_n)$$

This is simply applying the standard $\otimes$ operation to each of the elements of the tuple. The $\oplus$ is operator defined as:

$$A \oplus B = \begin{cases} A & \text{if} \sum_i a_i \leq \sum_i b_i, \\ B & \text{otherwise} \end{cases}$$

In the tuple version of the tropical semiring, the component wise sum of weights will correspond to the same weight in the scalar tropical semiring. Therefore we can use tuple version for training, or scalar version for decoding.

For the realization we make use of OpenFst's [25] elegant templating mechanism. The new semiring can be easily integrated into the existing tools and libraries. To optimize the log-linear factors we make use of the ZMERT toolkit[26]. After decoding an n-best list is generated from the lattices output from the WFST based translation decoder and input to ZMERT. The new weights are then applied to the component FST and decoding/optimization process is run until ZMERT converges.

5.3. OOV in WFST translation

After the SMT training process we frequently observed cases where individual words only occurred in the context of larger phrases. During testing if these words occurred in different contexts it would be impossible to construct a valid path through the WFST translation machines and the translation would fail. Adding epsilon transitions to model deletions from the source sentence did not work well in practise and substantially increased the search time. The phrase-table expansion discussed in section 4.3 was essential in the WFST decoder to achieve BLEU scores comparable to Moses.

Table 1: Comparison of the Moses and WFST decoding approaches.

Configuration	Moses	WFST
Baseline	32.29	33.16
+ phrase table expansions	32.38	34.29
+ MERT optimization	35.44	35.77

5.4. Experiments

In these experiments we compared Moses to the WFST based decoder, the goal was to evaluate the effectiveness of the phrase table expansion and the weight tuning procedure.

We used the same phrase table and testset from the previous experiments. Because the WFST system did not use a re-ordering component we disabled re-ordering in Moses.

To optimize the weights in both system we used 1000 one-to-one sentence pairs. For Moses we used the standard `mert-moses.pl` script to optimize the weights. For the WFST based system we used ZMERT. Once optimized the WFST can be switched into the more efficient tropical semiring.

In the case of WFST decoder the final search network was $M \circ T \circ N \circ P \circ G$. Where M is the source phrase segmentation transducer, T is the phrase table, N is the target phrase segmentation transducer, P is the insertion penalty and G is the target language model.

The results in table 1 show the comparison between the phrase-based decoder and the WFST. The results show the phrase-table expansion give a large improvement in the case of the WFST decoder, because it allows all test sentences to be successfully translated. The expansion method also gives a very small improvement in the Moses decoder.

The tuning semiring is effective for the WFST decoder and gives slightly better performance when compared to the Moses decoder. In the un-tuned systems the WFST performs better, possibly due to differences in the default weight selections.

Finally we considered the performance of the WFST decoder using lattice input. The lattices had both acoustic and language model weights, silence words were mapped to epsilon transitions. On this testset we only observed a 0.2 improvement in BLEU scores. It is possible that optimizing the WFST decoder using the lattice input and taking into account the language model weighting could improve performance.

5.5. Future Work in WFST Coupling

In future work we would like to optimize the speech recognition weighting parameters as part of the MERT training of the lattice coupled system.

We also plan to investigate techniques to extract better confidence scores from the lattices generated from a WFST decoder. One possibility is to is encode word end markers into the context-dependency transducer and use a specialized epsilon removal algorithm to convert the phone-lattices to word-level lattices with the correct alignment of the acoustic scores. To optimize the lattices it should be possible to use some of the recent work on semirings and determinization algorithms [17, 27, 28]. Finally, the LM scores can be re-applied using failure translation to ensure correct weight synchronization whilst avoiding the expansion of un-needed lower order paths. With such lattices available it should be possible to perform deeper comparison between the WFST decoder and Moses decoder.

Finally, we need to add re-ordering into SMT WFST decoder, another WFST alternative would be to to investigate the hierarchical WFST approach as described in [23].

6. Conclusions

In the first part of the paper we looked at basic coupling between a WFST ASR system and Moses phrase decoder. We found we can substantially reduce the memory consumption of the ASR system by reducing the vocabulary to exactly match the SMT, whilst having little impact on the BLEU score.

In the second part of the paper, we reported our initial experiments on coupling WFST ASR and SMT decoders. We have shown it is more important to deal with OOV and perform phrase table expansion when using a WFST based SMT decoder. Our proposed approach is extremely simple and yields goods results. We have also described a semiring that allows the optimization of the log-linear weights in the WFST framework and shown its effectiveness.

7. Acknowledgements

Thanks to Michael Paul, Shigeki Matsuda and Teruaki Hayashi for helping to obtain the original acoustic models and data used in for IWSLT 07 Japanese speech recognition system. We thank the anonymous reviewers for helpful comments on this paper.

8. References

[1] E. Vidal, "Finite-state speech-to-speech translation," in *Proc. ICASSP*, 1997, pp. 111–114.

[2] F. Casacuberta, "Finite-state transducers for speech-input translation," in *Proc. ASRU*, 2001.

[3] S. Bangalore and G. Riccardi, "Stochastic finite-state models for spoken language machine translation," *Machine Translation*, vol. 17, pp. 165 – 184, 2002.

[4] D. Caseiro and I. Trancoso, "Weighted finite-state transducer inference for limited-domain speech-to-speech translation," in *Computational Processing of the Portuguese Language*, 2006, vol. 3960, pp. 60–68.

[5] W. Shen, B. W. Delaney, T. Anderson, and R. Slyh, "The MIT-LL/AFRL IWSLT-2007 mt system," in *Proc. IWSLT*, 2007.

[6] N. Bertoldi, R. Zens, M. Federico, and W. Shen, "Efficient speech translation through confusion network decoding," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, pp. 1696 – 1705, 2008.

[7] E. Matusov and H. Ney, "Lattice-based ASR-MT interface for speech translation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, pp. 721–732, 2011.

[8] X. He, L. Deng, and A. Acero, "Why word error rate is not a good metric for speech recognizer training for the speech translation task?" in *Proc. ICASSP 2011*, 2011, pp. 5632–5635.

[9] R. Sarikaya, Z. Bowen, D. Povey, M. Afify, and Y. Gao, "The impact of ASR errors on speech-to-speech translation performance," in *Proc. ICASSP*, 2007, pp. 1289 – 1292.

[10] G. Saon and M. Picheny, "Lattice-based viterbi decoding techniques for speech translation," in *Proc. ASRU*, 2007, pp. 386 – 389.

[11] M. Mohri, F. C. N. Pereira, and M. Riley, "Speech recognition with weighted finite-state transducers," *Springer Handbook of Speech Processing*, pp. 1–31, 2008.

[12] C. Allauzen, M. Riley, and J. Schalkwyk, "A generalized composition algorithm for weighted finite-state transducers," in *Proc. Interspeech*, 2000, pp. 1203–1206.

[13] C. S. Fordyce, "Overview of the IWSLT 2007 Evaluation Campaign," in *Proc. IWSLT 2007*, 2007, pp. 1–12.

[14] B.-J. P. Hsu and J. Glass, "Iterative Language Model Estimation: Efficient Data Structure & Algorithms," in *Proc. Interspeech*, 2008, pp. 58–65.

[15] T. J. Hazen, I. L. Hetherington, and A. Park, "FST-Based Recognition Techniques for Multi-Lingual and Multi-Domain Spontaneous Speech," in *Proc. Eurospeech*, 2001, pp. 58–65.

[16] H. Hoang and P. Koehn, "Design of the Moses Decoder for Statistical Machine Translation," in *Proc. Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, 2008, pp. 58–65.

[17] B. Roark, R. Sproat, and I. Shafran., "Lexicographic semirings for exact automata encoding of sequence models," in *Proc. ACL*, 2011, pp. 1–5.

[18] K. Arora, M. Paul, and E. Sumita, "Translation of unknown words in phrase-based statistical machine translation for languages of rich morphology." in *Proc. of 1st International Workshop on Spoken Languages Technologies for Under-resourced languages*, 2008, pp. 70–75.

[19] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, 2000.

[20] M. Riley, F. Pereira, and M. Mohri, "Transducer composition for context-dependent network expansion," in *Proc. Eurospeech*, 1997, pp. 1427–1430.

[21] C. Allauzen, M. Mohri, and B. Roark, "Generalized algorithms for constructing statistical language models," in *Proc. of 41st Annual Meeting of the Assocication for Computational Linguistics*, 2003, pp. 40–47.

[22] S. Kumar, Y. Deng, and W. Byrne, "A weighted finite state transducer translation template model for statistical machine translation," *Natural Language Engineering*, vol. 12, pp. 35 – 75, 2006.

[23] A. de Gispert, G. Iglesias, G. Blackwood, E. R. Banga, and W. Byrne, "Hierarchical phrase-based translation with weighted finite-state transducers and shallow-n grammars," *Computational Linguistics*, vol. 36, pp. 505–533, 2010.

[24] M. Mohri, "Weighted automata algorithms," *Springer Handbook of weighted automata*, (to appear) 2009.

[25] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFst: A general and efficient weighted finite-state transducer library," in *Proc. of CIAA 2007*, 2007, pp. 11–23.

[26] O. Zaidan, "Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems," *The Prague Bulletin of Mathematical Linguistics*, no. 91, pp. 79–88, 2009.

[27] M. Y. Izhak Shafran, Richard Sproat and B. Roark, "Efficient determinization of tagged word lattices using categorial and lexicographic semirings," in *Proc. ASRU*, 2011, to appear.

[28] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiát, S. Kombrink, P. Motlíček, Y. Qian, N. T. Vu, K. Riedhammer, and K. Veselý, "Generating exact lattices in the WFST framework," 2011, submitted to ICASSP 2012.