Alibaba Speech Translation Systems for IWSLT 2018

Nguyen Bach, Hongjie Chen*, Kai Fan*, Cheung-Chi Leung*,*
Bo Li, Chongjia Ni*, Rong Tong*, Pei Zhang*,*
Boxing Chen, Bin Ma, Fei Huang

Machine Intelligence Technology Lab, Alibaba Group
firstname.lastname@alibaba-inc.org

Abstract

This work describes the En→De Alibaba speech translation system developed for the evaluation campaign of the International Workshop on Spoken Language Translation (IWSLT) 2018. In order to improve ASR performance, multiple ASR models including conventional and end-to-end models are built, then we apply model fusion in the final step. ASR pre- and post- processing techniques such as speech segmentation, punctuation insertion, and sentence splitting are found to be very useful for MT. We also employed most techniques that have proven effective during the WMT 2018 evaluation, such as BPE, back translation, data selection, model ensembling and reranking. These ASR and MT techniques, combined, improve the speech translation quality significantly.

1. Introduction

In this paper we describe the Alibaba speech translation system that was built as part of the Speech Translation Task in IWSLT 2018. The task involved translating English audio to German text in which English audio are from lectures and TED talks. Our system employs a pipeline approach that includes an automatic speech recognition system (ASR) and a machine translation (MT) system.

The paper is organized as follows: Section 2 presents the ASR system used, along with a description of conventional, end-to-end, and fusion systems. Section 3 focuses on the MT system in which we describe preprocessing, data augmentation, noisy input translation, ensembling, and reranking components in detail. We present our concluding remarks in Section 4.

2. Automatic Speech Recognition

In a pipeline-based speech translation system, ASR is the most front-end module. In order to get reliable translation of quality, it is critical to obtain ASR transcription as accurate as possible. To start, we build several conventional pipeline-based ASR systems using deep neural network/hidden Markov model (DNN/HMM) framework. In the DNN-HMM framework, we employ several DNNs with different structures including fully-connected DNN (FDNN), time-delay deep neural network (TDNN) [1, 2], and latency-controlled bidirectional long short-term memory (BLSTM) [3].

Our final goal is to build end-to-end speech translation system, that is, we need to simplify the model-building process of conventional pipeline-based ASR system by constructing complicated modules with a single DNN architecture or in a data-driven learning method. Thus we also employ an end-to-end ASR system which is based on a hybrid connectionist temporal classification (CTC) [4, 5] and attention based encoder-decoder [6] architecture. Finally, we combine the output of different ASR systems to boost the final ASR performance.

During the development of our system, all acoustic models are trained on TED dataset together with the training datasets provided by the organizer. We noticed that the organizer's segmentation of the talks/lectures is not quite accurate, e.g. some sentences are not properly split. Therefore, we employ our own model-based voice activity detection (VAD) module to split the talks/lectures into utterances before ASR decoding. For the model-based VAD, the recurrent neural network (RNN) model is used to train and classify each frame into non-speech or speech. The RNN based VAD model was trained by using TED and other speech corpus, and by using Alibaba's VAD segmentation, it can get better performance when comparing with Organizer's segmentation. Table 2 listed the comparsion.

Table 1: Configuration of DNNs in DNN-HMM acoustic models.

System	Input feature	#Dim	Network context
FDNN	FBank+Pitch	80	{-5, 5}
TDNN	MFCC+ivector	40+100	{-13, 9}
BLSTM	FBank+Pitch	80	{-8, 8}

#Dim: number of feature dimension
{-L, R}: L frames in the left context and R frames in the right context

* Equal contribution

Table 2: WER (%) of different ASR systems on IWSLT 2013 and 2015 dataset using organizer's and Alibaba's segmentation.

System	Organizer's segmentation			Alibaba's segmentation		
	tst2013	tst2015	Avg.	tst2013	tst2015	Avg.
CTC (baseline)	26.1	37.6	31.9	-	-	-
1. FDNN/HMM	22.9	31.6	27.3	19.4	28	23.7
2. TDNN/HMM	13.6	25.5	19.6	11	23.1	17.1
3. BLSTM/HMM	13.6	26.3	20.0	10.6	22.3	16.5
4. CTC/Attention	26	33.1	29.6	23	29.8	26.4
1+2+3+4	-	-	-	10.3	22.3	16.3
2+3+4	-	-	-	**8.6**	**21.7**	**15.2**

2.1. Conventional ASR system

In the conventional DNN-HMM framework, DNNs are designated to predict the alignments derived from a GMM-HMM based acoustic model, given different input features. As shown in Table 1, FDNN and BLTSM take 80-dimensional filter bank feature vectors (FBank) and pitch feature as input feature while TDNN take 40-dimensional Mel-frequency cepstral coefficients (MFCCs) appended with 100-dimensional iVector as input feature. These DNNs take different lengths of context. FDNN takes one frame together with 5 frames from its left and right context as the input window, i.e. context span is {-5, 5}. TDNN follows the setting in [2] which takes {-13, 9} as the context span. BLSTM takes context span {-8, 8}.

We train a 4-gram LM using all the allowed text. The 4-gram LM obtained by linearly interpolating a number of LMs that are trained using all the TED transcripts provided by the organizer, all the text in the WMT18 CommonCrawl corpus, some sentences selected from the WMT18 news, WMT18 news discussion and OpenSubtitles2018 corpora. We use cross-entropy based data selection to select sentences from the corpora that are close to the TED transcripts. The LM interpolation weights are optimized on all development data.

2.2. End-to-end ASR system

We employ a hybrid CTC/Attention architecture [6] in our end-to-end ASR system. This system consists of a BLSTM-based encoder, a CTC output layer and an attention decoder. The CTC output layer and the attention decoder takes the output of encoder and predicts the corresponding letters. That is, the end-to-end system is charactor level system. Please refer to [6] for more system details. And in Table 2, there are large differences between CTC/attention based ASR system and TDNN or BLSTM based ASR system. The reasons lie in that (1) the scale of speech training data. The CTC/attention based ASR need more large scale of traning data to get better performance comparing with TDNN or BLSTM based approach; and (2) the weak language model for CTC/attention based ASR system.

2.3. Fusion of ASR output

With all the aforementioned ASR systems, we have a set of ASR output of each test set. The WER of the ASR systems are summarized in Table. 2. We utilize ROVER [7] to fuse the output from different ASR systems. By enumerating all combinations of each ASR system, we select the output from the combination of the TDNN/HMM, BLSTM/HMM and hybrid CTC/attention based end-to-end systems, since this combined output gives the lowest WER. There are so big differences between them in model structure and used features for TDNN, BLSTM and CTC/attention systems, and therefore, after fusing between them, it can get better performance.

3. Machine Translation

3.1. MT baseline

In this section, we describe how our MT system has been developed. All our models are based on the transformer architecture in [8]. We start with the TED corpus, speech-translation TED corpus, and WMT18 data that are relevant to the speech translation domain. The total size of the bilingual corpus is 6.3 million sentence pairs. We use Marian toolkit for all experiments [9] and our development set includes dev2010 and tst2010-2015. The baseline architecture of Marian mainly follows the default setting for transformer NMT except for a 6-layer transformer encode-decoder, a 0.1 label smoothing, and 0.1 dropout between transformer layer. For parameter optimization, we use synchronized ADAM [10] with learning rate 0.0003, and set up the number of noam warm-up steps as 16,000.

3.2. Preprocessing

The training data is preprocessed following standard procedure. We first use the scripts in moses toolkit[1] for punctuation normalization, tokenization and lowercasing. Afterwards, we jointly learn and apply the byte pair encoding[2] for English and German together. Figure 1 shows a detailed

[1] https://github.com/moses-smt/mosesdecoder
[2] https://github.com/rsennrich/subword-nmt

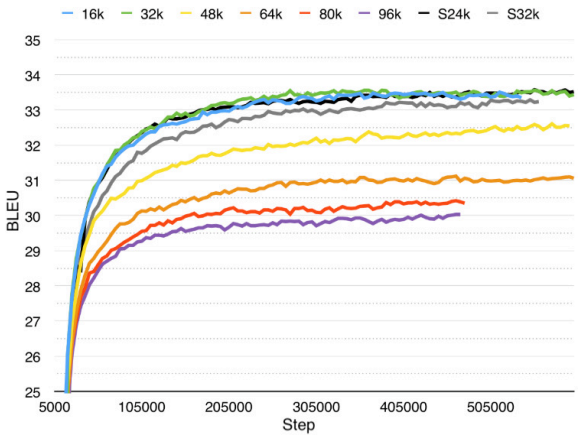

Figure 1: Performance of BPE operations including joint-bpe 16k, 32k, 48k, 64k, 80k, 96k, and shared-bpe 24K and 32K on our development set.

comparison of different BPE operations on our development set. We observe that joint bpe with 32K vocabulary performs best in this case and it is our final BPE code size. In the end, sentences longer than 100 tokens are removed.

Generally, the ASR raw output is a long stream of words with no punctuation, capitalization or segmentation markers. [11] shown that using various types of text segmenters between ASR and MT modules can improve speech translation quality. Our ASR system performed audio segmentation, punctuation and capitalization prediction. We use spaCy[3] to segment processed ASR transcription into shorter chunks. By using text segmentation we observed BLEU score gains between 0.2 and 1.0, depending on different ASR and MT systems.

For sentence boundary detection and punctuation insertion model, we experiment with a 3 layers LSTM for sequence tagging, and 3-gram KenLM[12] for additional scoring. In prediction phrase, we store some token in buffer as the foregoing context which is useful in real time prediction. Silence time in ASR is also used for sentence boundary detection. We also trained single layer self attention sequence tagging model and a bi-directional self attention LM. And we found it got much higher comma F1 score, but a litter lower period F1 score. This approach was not used in the final result and we will do more experiments in our future work.

3.3. Data Augmentation

In order to obtain a high quality domain related training corpus, we exploit the algorithm described in [13, 14], aiming at selecting sentence pairs from large out-domain corpus that are similar to the target domain. In our experiments, the 200K TED talks data is considered as in-domain corpus, and all the other parallel corpora provided are combined as a large out-domain corpus. Two 3-gram language models are trained over the source and target side of the in-domain corpus, respectively. Then, another set of two 3-gram language models are trained, whose training data is randomly selected from the out-domain corpus, with size being similar. Thus, each sentence pair from out-domain corpus is scored by the bilingual cross-entropy difference model. Finally, we sort all sentence pairs and select top ranked sentences pairs. Our experiments show that with different amount of additional data, we obtain BLEU gain on the development set between 0.4 to 1.8.

[3] https://spacy.io/

3.4. Noisy input translation

Though our transformer translation system is applied to the post-processed speech recognition outputs, the insertion, deletion and substitution errors still cannot be removed. Following the idea in [15, 16], we use the corrupted inputs to train a robust neural machine translation model. Since we observe that the insertion error is rare in our ASR system, only the deletion and substitution noises are considered for the source sentences of the regular parallel training data. In this way, the gap between training data and testing ASR output will become potentially smaller.

Basically, we first randomly delete the token of the source sentence with a small probability (0.01 and 0.02 are selected in our experiments by cross-validation). Specifically, we also pre-define a functional-words list including 120 tokens with number of letters less than 5, and assign a doubled deletion rate for them. Notice that we train several deletion-noisy models alone without any other corrupted strategy for further ensemble. In our experiments, the single model trained with deletion noise can increase the case insensitive (CI) BLEU for at least 0.5 BLEU point over the baseline on the ASR output.

Additionally, we attempt to introduce the substitution noise by randomly replacing the token with its pronunciation-like candidates with a small probability. By trying different substitution rate, we empirically found that the substitution noise model achieved no significant improvement over deletion noise model. One possible solution is to use the adaptive substitution rate of each token, estimated with the maximum likelihood in the ASR model. We will leave this as the future work.

The third strategy for noisy training is punctuation simulation. We randomly spare 30% of the regular parallel corpus and remove all punctuations from the source side, then annotate/re-generate the same data with the tool used in the ASR post-processing. We add the punctuation noisy corpus back and train our model, and empirically observe an improvement of at least 0.5 BLEU point gain compared with baseline as well. The punctuation simulation and deletion/substitution noise are typically not combined, since the underlying true noise comes from the ASR system and our two manually designed noising systems may have a large bias. Therefore, we decide to apply them separately to increase the diversity of our models in ensemble.

3.5. Refinements

3.5.1. Ensemble decoding

Model ensemble is a widely used technique to boost the performance of a MT system, which is to combine the prediction of multiple models at each decode step. We adopt the ensemble method GMSE (Greedy Model Selection based Ensemble) detailed in [17].

The candidate single models are first sorted as a list according to their performance on the development dataset. Another two model lists are maintained during the ensemble, named "keep", "redemption". For each iteration, a candidate model could be either drawn from the beginning or the end of the candidates list with probability p or p_{reserve}, or the "redemption" list with probability p_{redempt}, where $p + p_{\text{reserve}} + p_{\text{redempt}} = 1$. Then, the selected model is temporarily concatenated to the "keep" list. If the evaluation of the current model ensemble achieves a better BLEU score, the model is permanently added to the "keep" list. Otherwise, it will be put into the "redemption" list. Notice that one model from the "redemption" list can only be redeemed once, after which it is withdrawn permanently from the candidates.

In order to achieve better ensemble performance, we increase the diversity of our candidate models by introducing another 8 training schedules and further obtain about 200 single checkpoints. The greedy nature of the GMSE algorithm makes the search feasible in an acceptable time frame. In summary, we list the different training schedules as follows.

1. 10 million training corpus is selected by BPE level language models.
2. 8 million training corpus is selected by word level language models.
3. Adding another 3 million back-translation data to the original parallel corpus.
4. Training with deletion/substitution noise.
5. Training with punctuation simulation noise.
6. Fine-tuning with the 200K TED talks in-domain data.
7. Fine-tuning with the punctuation simulation data.
8. 7-layer transformer NMT model.

Due to the time limitation, we cannot do all the ablative experiments on listed strategies. However, we can still report the best single model and best ensemble result on our ASR output in Table 3. Note that our development set is the combination of tst2013 and tst2015, we did not test our model on these two dataset separately.

Table 3: Improvement with ensemble

	tst2013 + tst2015
best single model	22.96
best ensemble	23.84

3.5.2. Reranking

Besides building multiple ensemble systems with different random initialization and configurations, we also build and optimize the n-best list reranker. We follow the approach in [17] in which several neural machine translation models and language models have been experimented with. The n-best list reranker involves the following steps

- Build and optimize neural MT models including single models, ensemble models, and models with different configurations such as beam size. To improve diversity, we also use the right-to-left and target-to-source models.
- Build ngram language models from in-domain and out-of-domain data using the data selection method similar to [17, 18].
- Apply the the greedy feature selection based reranking method in [17] to train the reranker. To deal with overfiting, we use the tuning set that contains both manual transcription and our ASR transcription.

Our experiments show that depending on different settings the reranker typically obtains improvements between 0.1 to 0.4 BLEU point over the best system.

Additionally, we experiment with the multi ASR inputs for reranking. The main motivation is to directly exploit strengths of different ASR systems into the reranking system. Out initial results show that the multi ASR inputs does not outperform the input for the ASR fusion output.

3.5.3. Recaser

Since the lowercased corpus are applied to train our MT system, an additional post-processing recaser model is necessary to obtain the truecased (or capitalized) German translation output. In principle, we exploit the combination of the moses[4] SMT recaser model and the Char-RNN based neural recaser model [19]. The SMT recaser model essentially trains a word-to-word translation model and a cased language model without reordering. Unlike the word-level approach, the neural recaser model restores the case information at the character-level, reducing the recasing problem to a sequential binary classification task. No special treatment is required for mixed cased words.

The final combination rule is that for every single word, if the SMT recaser and the neural recaser reach a consensus

[4] http://www.statmt.org/moses/?n=Moses.SupportTools

on the final recasing output, we will accept it; if they have a disagreement, we will always capitalize that word. This strategy will result in a combined recaser model which is slightly better than any other one.

4. Conclusions

The IWSLT 2018 Speech Translation task provided the opportunity to compare different speech translation approaches using shared datasets and standardized evaluation metrics. Table 4 shows our submission results on the IWSLT 2018 official test set.

Table 4: IWSLT 2018 final evaluation results on contrastive and primary submissions

	con.1	con.2	primary	con.3
BLEU	22	22.16	22.36	22.5
TER	63.44	63.52	63.03	63.03
BEER	52.47	52.69	51.77	52.64
CharTER	59.56	57.54	69.26	57.76
BLEU(ci)	23.97	24.14	24.23	24.3
TER(ci)	60.44	60.41	60.22	60.15

Our participation in this task revealed three important aspects of speech translation that we regard as important for the future.

First, our experiments indicated that the speech segmentation and transcription post processing, by themselves, can make a big differences on transcription quality as well as translation quality. Furthermore, these components not only improve WER and BLEU scores, in live speech translation scenario they also greatly improve user experience.

The second aspect relates to the importance of noisy inputs. There are many types of noises in speech translation scenario. For example noise come from speech audio, transcription errors, and the nature of spoken language. Our experiments show that by modeling transcription errors directly in the neural MT (NMT) training, we obtained consistent improvement. It indicates that the NMT model becomes more robust against errors of our ASR system. Also, if we compare statistical machine translation (SMT) technique, we find SMT is generally more robust to noises than NMT. It is probably because SMT models are built on probability distributions estimated from many occurrences of words and phrases, therefore any unsystematic noise in the training only affects the tail end of the distribution.

The third aspect is the importance of model engineering. In the statistical machine learning, the best model is typically from through several rounds of feature engineering. In the NMT context, we see that our best model is also from many steps of model engineering and refinements. Given the availability and good scalability of ASR and MT toolkits today, it is tempting to throw as much model configurations as possible and let the built-in mechanisms of these learning algorithms figure out which one is the best. However, the strategy has its own limitations, and, in conjunction with the limited availability of the labeled data, can easily produce models that are under-performing on blind test sets.

5. Acknowledgments

We thank the support from Shanbo Cheng, Jun Lu for their help in reranking and recasing.

6. References

[1] A. H. Waibel, T. Hanazawa, G. E. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, 1989.

[2] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Interspeech*, 2015.

[3] S. Xue and Z. Yan, "Improving latency-controlled BLSTM acoustic models for online speech recognition," in *Proc. ICASSP*, 2017, pp. 5340–5344. [Online]. Available: https://doi.org/10.1109/ICASSP.2017.7953176

[4] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: first results," in *Proc. NIPS*, vol. abs/1412.1602, 2014.

[5] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, 2014, pp. 1764–1772.

[6] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE J. Sel. Topics Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[7] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997, pp. 347–352.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, 2017, pp. 5998–6008.

[9] M. Junczys-Dowmunt, R. Grundkiewicz, T. Dwojak, H. Hoang, K. Heafield, T. Neckermann, F. Seide, U. Germann, A. Fikri Aji, N. Bogoychev, A. F. T. Martins, and A. Birch, "Marian: Fast neural machine translation in C++," in *Proceedings of ACL 2018, System Demonstrations*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 116–121.

[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[11] V. K. R. Sridhar, J. Chen, S. Bangalore, A. Ljolje, and R. Chengalvarayan, "Segmentation strategies for streaming speech translation," in *Proc. of the Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics*, 2013, pp. 230–238.

[12] K. Heafield, I. Pouzyrevsky, J. H. Clark, and P. Koehn, "Scalable modified Kneser-Ney language model estimation," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, August 2013, pp. 690–696.

[13] A. Axelrod, X. He, and J. Gao, "Domain adaptation via pseudo in-domain data selection," in *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2011, pp. 355–362.

[14] K. Duh, G. Neubig, K. Sudoh, and H. Tsukada, "Adaptation data selection using neural language models: Experiments in machine translation," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2013, pp. 678–683.

[15] M. Sperber, J. Niehues, and A. Waibel, "Toward robust neural machine translation for noisy input sequences," in *Proc. of the International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan, December, 14-15 2017.

[16] N.-Q. Pham, M. Sperber, E. Salesky, T.-L. Ha, J. Niehues, and A. Waibel, "Kit's multilingual neural machine translation systems for iwslt 2017," in *Proc. of the International Workshop on Spoken Language Translation (IWSLT)*, Tokyo, Japan, December, 14-15 2017.

[17] Y. Deng, S. Cheng, J. Lu, K. Song, J. Wang, S. Wu, L. Yao, G. Zhang, H. Zhang, P. Zhang, C. Zhu, and B. Chen, "Alibaba's neural machine translation systems for wmt18," in *Proc. of the Conference on Machine Translation*. Brussels, Belgium: Association for Computational Linguistics, November 2018.

[18] B. Chen and F. Huang, "Semi-supervised convolutional networks for translation adaptation with tiny amount of in-domain data," in *Proc. of the Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 2016, pp. 314–323.

[19] R. H. Susanto, H. L. Chieu, and W. Lu, "Learning to capitalize with character-level recurrent neural networks: An empirical study," in *Proc. of the Empirical Methods in Natural Language Processing)*. Austin, Texas: Association for Computational Linguistics, November 2016, pp. 2090–2095.