The Xiaomi Text-to-Text Simultaneous Speech Translation System for IWSLT 2022

Bao Guo[1*] **Mengge Liu**[2*†] **Wen Zhang**[1] **Hexuan Chen**[1] **Chang Mu**[1]
Xiang Li[1] **Jianwei Cui**[1] **Bin Wang**[1] **Yuhang Guo**[2]
[1]Xiaomi AI Lab, Beijing, China
[2]Beijing Institute of Technology, Beijing, China
guobao@xiaomi.com 3120201046@bit.edu.cn
{zhangwen17,chenhexuan,muchang1,lixiang21}@xiaomi.com
{cuijianwei,wangbin11}@xiaomi.com guoyuhang@bit.edu.cn

Abstract

This system paper describes the Xiaomi Translation System for the IWSLT 2022 Simultaneous Speech Translation (noted as SST) shared task. We participate in the English-to-Mandarin Chinese Text-to-Text (noted as T2T) track. Our system is built based on the Transformer model with novel techniques borrowed from our recent research work. For the data filtering, language-model-based and rule-based methods are conducted to filter the data to obtain high-quality bilingual parallel corpora. We also strengthen our system with some dominating techniques related to data augmentation, such as knowledge distillation, tagged back-translation, and iterative back-translation. We also incorporate novel training techniques such as R-drop, deep model, and large batch training which have been shown to be beneficial to the naive Transformer model. In the SST scenario, several variations of `wait-k` strategies are explored. Furthermore, in terms of robustness, both data-based and model-based ways are used to reduce the sensitivity of our system to Automatic Speech Recognition (ASR) outputs. We finally design some inference algorithms and use the adaptive-ensemble method based on multiple model variants to further improve the performance of the system. Compared with strong baselines, fusing all techniques can improve our system by 2~3 BLEU scores under different latency regimes.

1 Introduction

In the IWSLT 2022 Evaluation Campaign, our team at Xiaomi AI Lab participates in one Simultaneous Speech Translation task (Anastasopoulos et al., 2022), which is the Text-to-Text track in English to Mandarin Chinese translation direction. We first introduce the techniques used in our final submitted system from four aspects: data, model, inference, and robustness.

Data-related techniques are introduced from two perspectives: data augmentation and domain-related data selection. For data augmentation, we employ technologies such as back-translation (BT) (Sennrich et al., 2016a), knowledge distillation (KD) (Kim and Rush, 2016), and iterative back-translation (Hoang et al., 2018) etc. to generate large-scale synthetic bilingual datasets, which have been proved to be very effective in the field of machine translation. We also use the technology of Tagged Back-Translation (TaggedBT) (Caswell et al., 2019), that is, prepending a reserved token <BT> to the beginning of the synthetic source sentence in the training set, so that the model could distinguish the originality of the source sentence. Meanwhile, the effects of different combinations of multiple training sets on the model performance are explored. For domain-related data selection, differences in the domains of the training and test sets can have a large negative impact on the results on the test sets. To make the model obtain domain-related knowledge as much as possible, we apply the LM-based data selection technique (Axelrod et al., 2011) to select high-quality and domain-related data from bilingual corpora.

In terms of model, since the submitted systems will be ranked by the translation quality with three latency regimes (low, medium, and high), participants are encouraged to submit multiple systems for each regime to provide more data points for latency-quality tradeoff analyses. Besides, we empirically believe that different models have different translation performance and inference latency on T2T tasks, and they can complement each other, so we build various advanced SST models (i.e. BASEDEEP and BIGDEEP), which are all based on deep Transformer model (Vaswani et al., 2017), but have been empirically proven to outperform the Transformer-Big model on the SST model. For

[*]Equal contribution.
[†] The work was done during the author's internship at Xiaomi.

the T2T track, the output of a streaming ASR system (usually prefix of the entire source sentence) will be fed into the SST system as input instead of the gold transcript. So we adopt the `wait-k` training strategy (Ma et al., 2019; Elbayad et al., 2020) to meet the scenario of simulating simultaneous translation. In addition, we also employ the R-Drop (Liang et al., 2021) and adaptive-ensemble techniques (Zheng et al., 2020) which have also been proven beneficial for translation models.

For inference, we empirically analyze the problems of our system in translation under low latency (e.g. when k is equal to 3) and propose a constrained decoding strategy to wait for some specific words or phrases to appear before translation, which can alleviate some translation issues of the `wait-k` model in low-latency situations as much as possible.

The input fed into the SST model is the output of the ASR system, and according to the statistics of previous researchers, the two error types homophones and words with a similar pronunciation account for a large proportion in the output of the ASR system. Therefore, in order to weaken the model's sensitivity to ASR output errors, we introduce methods to enhance the model's robustness to both error types: homophones or words with a similar pronunciation. Additionally, a char-to-subwords error correction model is further proposed to correct ASR errors before feeding into the translation model.

The remainder of this paper is organized as follows. We perform statistics on the data used and introduce pre-processing in Section 2. Section 3 and 4 elaborate our systems, the techniques employed, and evaluation, followed by the main experimental results and ablation studies reported in Section 5. Finally, we conclude this paper in Section 6.

2 Data

We introduce the data used in our system from the following three aspects: statistics, pre-processing and filtering.

Statistics. We use the allowed training sets, which include MuST-C v2.0 [1], CoVoST [2], TED corpus [3], OpenSubtitles2018 [4], and the bilingual corpus provided by WMT2021 [5]. We find that the four datasets MuST-C v2.0, CoVoST, TED corpus, and OpenSubtitles2018 are all datasets that are biased towards the oral domain, so we combined these four datasets as the training set in **Oral** domain. We also empirically treat WMT21 as the training set in the **News** domain. The statistical results of the original datasets are shown in Table 1. Among them, all the available bilingual corpora provided by WMT2021 includes: News Commentary v16 (0.32M) [6], Wiki Titles v3 (0.92M), UN Parallel Corpus V1.0 (15.9M), CCMT Corpus (8.9M), WikiMatrix (2.6M), Back-translated news (19.8M), and ParaCrawl v7.1 (14.2M). We use the `tst-COMMON` test set (including $2,841$ sentences) as the development set to validate our models.

Bilingual data		Size	Filtered
Oral	**MuST-C v2.0**	360K	7.8M
	CoVoST	870K	
	TED corpus	250K	
	OpenSubtitles2018	11.2M	
News	**WMT2021**	61.1M	45.3M
Total	-	75.32M	53.1M

Table 1: The statistical results of all available bilingual training sets.

Pre-processing. `Sacremoses` [7] is conducted to normalize and tokenize English sentences. We use the traditional and simplified conversion tool to convert traditional Chinese text to simplified, use the `jieba` [8] tool to segment Chinese sentences, and remove redundant spaces in the text.

Rule-based Filtering. The training set is filtered according to the following rules (the content in parentheses after each item indicates the number of sentence pairs remaining after the current step of filtering is performed):

- We remove duplicate sentence pairs and empty data in the training set (65.3M);
- We first use `fast_align` [9] to filter out sen-

[1] https://ict.fbk.eu/must-c/

[2] https://github.com/facebookresearch/covost

[3] https://wit3.fbk.eu/2017-01-c

[4] https://opus.nlpl.eu/OpenSubtitles2018.php

[5] https://www.statmt.org/wmt21/

[6] Numbers in parentheses indicate the number of parallel sentence pairs

[7] https://github.com/alvations/sacremoses

[8] https://github.com/fxsjy/jieba

[9] https://github.com/clab/fast_align

tence pairs with scores less than -7 and then use Language Identification (LangID) tool [10] to remove those sentence pairs that do not contain English or Chinese (55.9M);

- Sentence pairs in which more than 58% of the tokens in the source sentences appear in the target sentences are discarded (53.8M);
- Sentence pairs with a length ratio of source to target or a length ratio of target to source greater than 3.0, or sentence pairs containing sentences with a length of more than 100 tokens are discarded (53.1M).

The size statistics of the training set on domains **Oral** and **News** are shown in Table 1. The filtered training set on the two domains contains 53.1M sentence pairs, marked as s1 (as shown in Table 3).

Language-model-based Filtering. Drawing on the method of Axelrod et al. (2011), we train two 5-gram language models (denoted as lm^{in} and lm^{out}) on English sentences in the MuST-C v2.0 (oral domain) and s1 (news domain) training sets respectively. For each English sentence in s1, we use lm^{in} and lm^{out} to calculate ppl^{in} and ppl^{out} respectively. Sentence pairs in s1 are sorted in ascending order according to the value of $ppl^{in} - ppl^{out}$, and the first 30M are selected as the parallel corpus related to the oral domain. Finally, based on the pre-trained language model, s1 is filtered into a bilingual parallel corpus of size 30M related to the oral domain (**Fppl** shown in Table 3).

3 Configurations

3.1 Model Settings

For the implementation of Transformer, we use the code provided by fairseq[11] (Ott et al., 2019). The token-level batch size is set as about 250k on 8 GPUs for all the experiments. The learning rate is set as 1e-3 for all models, which is controlled by Adam optimizer (Kingma and Ba, 2014). To acquire strong baselines, dropout (Srivastava et al., 2014) is used and set as 0.05 for all the models. We use byte-pair encodings (BPE) (Sennrich et al., 2016b) with $32k$ for all models. All submitted models are trained by using s4 on 8 V100 GPUs or 8 A100 GPUs. For training each model, we run 100k steps and save the model every 2.5k steps with the early stop mechanism, which means that if there are 10 consecutive checkpoints with no improvement in BLEU on the development set, then the training is terminated. The sizes of English vocabulary and Chinese vocabulary are $33,512$ and $43,048$ respectively.

[10] `https://github.com/saffsd/langid.py`

[11] `https://github.com/pytorch/fairseq`

3.2 Evaluation

Following official automatic evaluation criteria, we use BLEU score (Papineni et al., 2002) to evaluate our system for translation quality. For translation latency, standard metrics average lagging (AL) (Ma et al., 2020) is applied for simultaneous machine translation. In order to simulate the speech-to-text translation latency for a text-to-text task, we also use the officially provided noisy test set `tst-COMMON` to simulate non-computation-aware AL (NCA-AL), which are decoded with the streaming ASR model and contain the source timestamps [12]. SimulEval [13] open-source tool is employed to calculate BLEU and AL.

	base_eadb	**big_exdy**
Encoder layers	a	x
Decoder layers	b	y
Embedding Dim	512	2048
FFN Dim	1024	4096
Number of Heads	8	16

Table 2: The configurations of our deep Transformer models. Note that the **base_eadb** model has an a-layer encoder and a b-layer decoder, the encoder and decoder of the **big_exdy** model have x and y layers respectively. "Dim" means the dimension size.

4 Techniques

In this section, we elaborate the models we use and the employed techniques.

4.1 Deep Architecture

Our submitted system uses two deep Transformer models, named base_eadb and big_exdy. We use the deep-norm technique proposed by Wang et al. (2022) to train the deep models. The base_eadb models we adopt contain an a-layer encoder and a b-layer decoder with `Transformer-base` setting. For big_exdy, we train deep Transformer models with an x-layer encoder and a y-layer decoder by leveraging `Transformer-big` setting. The detailed model configuration is shown in Table 2. Our

[12] `https://github.com/facebookresearch/SimulEval/blob/main/docs/timestamps.md`

[13] `https://github.com/facebookresearch/SimulEval`

Name	**Oral** (7.8M)	**News** (45.3M)	**Fppl** (30M)	**Foral** (6.5M)	**Size**
s1	P	P	-	-	53.1M
s2	P+TaggedBT+KD	P+TaggedBT+KD	-	-	150M
s3	-	-	1KD	2TaggedBTv1+3KDv1	48M
s4	-	-	1KD	2P+2TaggedBTv2+3KDv2	58M

Table 3: Four training sets obtained according to different combinations of datasets. The detailed description of **Oral** and **News** can be seen from Table 1. "P" means parallel data. "TaggedBT" represents tagged back-translation. The numbers in front of "TaggedBT" or "KD" denote the number of models used to conduct back-translation and knowledge distillation respectively. "v1" and "v2" respectively indicate that the first and second iteration of data augmentation on the data in the corresponding columns. For rows s3 and s4 of the **Fppl** column, the 1KD data is translated by using the en2zh_base_e25d6_s1 model.

final submitted system contains only 2 deep models: en2zh_base_e40d6 [14] and en2zh_big_e12d6, with 210M and 370M parameters, respectively.

4.2 R-Drop

All models are trained by using the R-Drop training algorithm with the weight α set to be 5. More detailed description of the R-Drop training algorithm can be found in paper Liang et al. (2021).

4.3 `Wait-k` Strategies

Based on the naive `wait-k` algorithm proposed by Ma et al. (2019), we build our systems and make inference by using two variants of the `wait-k` algorithm, the details are as follows.

Training. The first is effective `wait-k` proposed by Elbayad et al. (2020), which means a fixed `k` value is selected during training (named as `wait(k)`), and the models are trained to generate the target sentence concurrently with the source sentence, but always `k` words behind. The second is multi-path `wait-k` policies introduced by Elbayad et al. (2020), which dynamically and randomly select a value within the `k`-value interval (such as `[k, k+t]`) for each batch during training (named as `wait(k)-(k+t)`).

Inference. At inference, we use two strategies: `single-k` and adaptive-ensemble. For `single-k`, corresponding to efficient `wait-k`, a fixed value of `k` is set during decoding. When the number of source tokens read minus the number of target tokens output is greater than or equal to `k`, the decoding is performed to output a token. In addition, we conduct the `waitmore` strategy. Specifically, when the read words are prepositions, punctuation, and other meaningless words, we make $\mathtt{k}+1$, that is, wait for one more source token. When the source has been read, we switch to the regular model to do the rest of the decoding.

Another strategy is adaptive-ensemble. Specifically, for multiple `wait-k` models, we test their performance on each `k` value in the interval $[1, 19]$, and then determine the top three models corresponding to each `k` value according to the model confidence (log-probability). During the decoding process, the `k` value starts from 1, and the upper bound is 19. At the current value of `k`, the top three models corresponding to the `k` value are used for ensemble decoding, and the top-1 probability value in the probability distribution is used as the confidence. If it is higher than the preset threshold, the decoding result is output, otherwise, the value of `k` is incremented by 1. The settings are the same as Zheng et al. (2020).

4.4 Data Augmentation

Back-translation (BT) (Sennrich et al., 2016a) and knowledge distillation (KD) are very effective data augmentation methods for the naive NMT model [15]. Here we empirically use the TaggedBT technique proposed by Caswell et al. (2019), which has been validated and concluded to be superior to BT. In particular, we add a reserved tag <BT> at the beginning of the source sentence in the training data synthesized by BT, and the tag is treated in the same way as all other tokens. Given the success of Nguyen et al. (2020) and Wang et al. (2020), we also adopt the ensemble method based on data diversification. The details of our approach are as follows.

Based on s1, we first train three English-to-Chinese models and two Chinese-to-English mod-

[14]en2zh_base_e40d6 means the English-to-Chinese translation model including a 40-layer encoder and a 6-layer decoder with `Transformer-base` setting.

[15]Compared with the `wait-k` model, we refer to the original NMT model as the naive NMT model.

els. We translate the **Fppl** training set by using above 5 models, and construct two BT data (noted as 2TaggedBT) and three KD data (noted as 3KD), then merge **Fppl**, 2TaggedBT and 3KD before deduplication to build corpus s2. For the **Oral** training set, we use the existing model to translate English into Chinese and sort in descending order according to sentence-level BLEU, then save 6.5M parallel corpus (denoted as **Foral**). Similarly, we perform the first iteration on the **Foral** data, obtaining two BT data (2TaggedBTv1) and three KD data (3KDv1). We finally merge 1KD, 2TaggedBTv1, and 3KDv1 before deduplication to build corpus s3. Finally, we perform the second iteration (Hoang et al., 2018) on the **Foral** data to obtain two BT data (2TaggedBTv2) and three KD data (3KDv2). 1KD, two copies of **Foral** data, 2TaggedBTv2, and 3KDv2 are merged before deduplication to generate the training set s4.

Our final submission system contains the following deep models: en2zh_base_e40d6_s4 [16] and en2zh_big_e12d6_s4, both of which are trained on data s4.

4.5 Robustness to ASR Noise

We propose two methods to improve the robustness of the system to ASR output noise, and the two methods are orthogonal.

Synthetic Noise Generation. The training set **Foral** is further filtered to 5.6M based on the sentence-level BLEU score between candidate and reference. We randomly generate synthetic noise on the English sentences in the filtered **Foral** to form synthetic bilingual data, then merge it with the authentic bilingual data to obtain final bilingual data s5 (including 11M sentence pairs).

The specific process of generating noise is as follows: for a word w, the Double Metaphone[17] and CMU pronouncing dictionary[18] are first used to obtain the consonants of w, and then words with the same consonants will be clustered together to form cluster C_w, note that $w \notin C_w$. Finally, with a probability of 5%, we either insert a word after w, delete w, or replace w with the corresponding homophone, which is the word in C_w with the smallest edit distance from w. en2zh_base_e40d6_s4 and en2zh_big_e12d6_s4 are finetuned on s5.

Error Correction Model. For the specific scenario of streaming ASR, we construct examples based on English sentences in **Foral** to train an error correction model: 1) insert, delete, replace or reorder the characters in the words randomly, and generate two noisy datasets on the entire sentence pairs and one noisy dataset on the prefix pairs [19]; 2) use the method proposed by Lee et al. (2018) to generate the pronunciation sequence of each sentence (with spaces reserved), and train a model to generate subword sequences from the pronunciation sequence (BLEU score is 96), then we randomly insert or delete spaces on the pronunciation sequence to simulate the noise of speech segmentation, and use the trained model to decode the noisy pronunciation sequence, finally reserve the decoding result different from the original sentence (4M) as noise data; 3) up-sample 3 copies of the authentic bilingual data in the entire sentence part, then up-sample 2 copies of the authentic bilingual data in the prefix part, and finally merge all bilingual data (including 48M sentence pairs) and train a char-to-subwords Transformer model for error correction.

Models	BLEU
en2zh_big_e6d6_s1	28.05
en2zh_big_e6d6_s3	28.94
en2zh_big_e6d6_s4	28.97

Table 4: The effect of training sets constructed with different data augmentation strategies on model performance.

5 Experimental Results

5.1 Main Results

To verify the impact of each dataset on model performance, we train three en2zh_big_e6d6 models on s1, s3 and s4. Note that we also train a deep model en2zh_big_e36d6 on s2, and the result is 28.90, which is comparable to the en2zh_big_e6d6 model on s4. Therefore, due to the large amount of s2, we only use en2zh_big_e36d6_s2 for subsequent data filtering and construction. The experimental results are listed in Table 4. As can be seen that the domain-related data augmentation

[16]en2zh_base_e40d6_s4 means the English-to-Chinese translation model which contains 40-layer encoder and 6-layer decoder and adopts `Transformer-base` setting, the model is trained on s4.

[17]Double Metaphone is a phonetic algorithm for indexing words by their English pronunciation.

[18]https://github.com/cmusphinx/cmudict

[19]We randomly truncate the prefix of the sentence pair to make the model aware of the scenario of streaming ASR.

(**Foral**) boosts the baseline by 0.89 BLEU score, but the iterative data augmentation does not seem to bring more gains. In addition, we also explore iterative data augmentation on en2zh_base_e40d6_s4 model, and the improvement is also not particularly obvious (28.94->29.07), so our final submitted systems do not use iterative data augmentation. We argue that the effectiveness of iterative data augmentation is strongly related to both the training sets and the model architectures.

According to the official, the latency thresholds are determined by the NCA-AL, which represents the delay to the perfect real time system. We finally submit two systems, a single-model system for CA scenarios and another adaptive-ensemble system for NCA scenarios. More experimental results can be found in (Anastasopoulos et al., 2022).

Models	BLEU
en2zh_big_e6d6_s1	27.96
en2zh_big_e6d6_s1 + R-Drop	28.37
en2zh_big_e20d6_s1 + R-Drop	28.55
en2zh_big_e25d6_s1 + R-Drop	28.77

Table 5: The impact of R-Drop and deep models on translation quality on the clean tst-COMMON test set.

5.2 Validation of R-Drop and Deep Model

For this ablation study, we train several models on data s1 and use the clean development set to verify the effectiveness of the R-Drop technique and deep models. The experimental results are shown in Table 5. It can be seen that the R-Drop technology improves our strong baseline by 0.41 points, and the deep model further improves 0.4 BLEU scores. We employ both techniques in all subsequent experiments.

5.3 Choice of k value

We empirically choose the optimal k-value or k-value interval based on the quality-latency ratio (QLR) on the development set.

Firstly, we train multiple en2zh_big_e6d6 models on the training set s1 (including 53.1 sentence pairs) using different k-values under effective wait-k policy and different k-value intervals under multi-path wait-k policy [20], then explore the impact of different k-values and different k-value intervals on QLR of decoding development

[20] Effective and multi-path wait-k policies correspond to wait(k) and wait(k)-(k+t) as defined in the **Training** paragraph in Section 4.3, respectively.

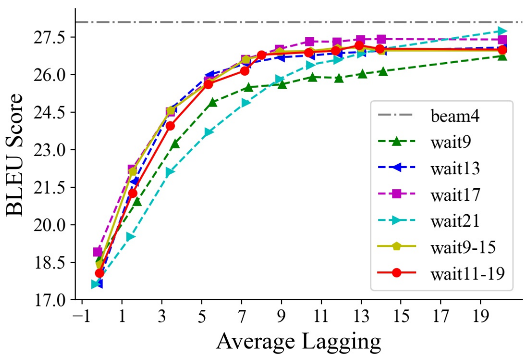

Figure 1: Comparison of QLR curves of different wait-k strategies on the development set. “beam4” denotes the naive decoding strategy with beam size 4.

set. For each policy, we test the BLEU scores under different average laggings on the development set, and draw the QLR curve, then compare the pros. and cons. of different strategies, as shown in Figure 1. As can be seen from Figure 1, when the value of k is too small or too large, the overall effect is relatively poor (for example, k=9 and k=21 correspond to the green and blue dashed lines in the figure, both of which are located at the bottom right). While wait17, wait9-15 and wait11-19 perform relatively well. Multi-path wait-k has almost the same effect as the effective wait-k policy, but has better robustness than the effective wait-k. Based on the above verification, our final submitted system includes the following 1 naive model and 6 wait-k models:

- en2zh_big_e12d6_s4
- en2zh_base_e40d6_s4_wait17
- en2zh_base_e40d6_s4_wait9-15
- en2zh_base_e40d6_s4_wait11-19
- en2zh_big_e12d6_s4_wait17
- en2zh_big_e12d6_s4_wait9-17
- en2zh_big_e12d6_s4_wait11-19

Models	BLEU
Baseline	19.02
+ Synthetic Noise Generation	19.23
+ Error Correction Model	20.28

Table 6: Performance comparison of different methods to improve the model’s robustness to ASR noise.

5.4 Robustness to ASR Noise

We explore the performance of our two methods on the noisy tst-COMMON test set provided by the

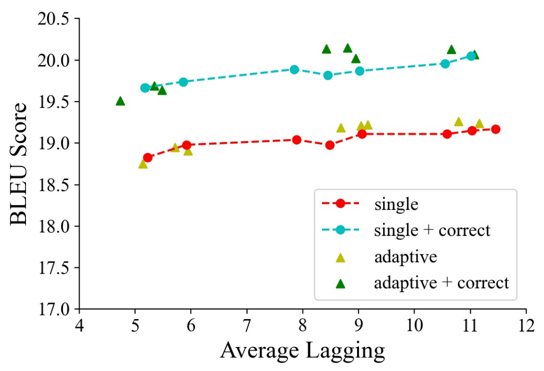

Figure 2: The benefits of the error correction model under the two inference strategies of single-k and adaptive-ensemble.

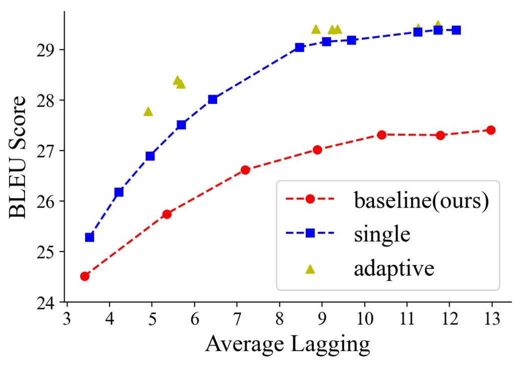

Figure 3: Comparison of QLR curves of baseline model, single-k decoding and adaptive-ensemble decoding on the development set.

official, and the results are shown in Table 6. It can be seen that the data-driven method has an improvement of 0.21 points compared to the baseline model. The error correction model is leveraged to correct the input before feeding the input into the translation model, which can further bring an improvement of 1.05 BLEU scores. We also verify the effect of the error correction model on the single model and ensemble model under different average laggings, the results are shown in Figure 2. It can be seen that the error correction model can significantly and consistently improve translation quality at both high and low latency, whether on single-k or adaptive-ensemble strategies.

5.5 Effect of Adaptive-ensemble

We use the inference strategy of single-k and adaptive-ensemble (introduced in the **Inference** paragraph in Section 4.3) to decode the development set, respectively, and then compare these two methods with the baseline model, and the results are shown in Figure 3. It can be seen that the QLR of the single-k strategy is significantly improved compared to the baseline model, and the adaptive-ensemble strategy brings further improvement.

6 Conclusion

We elaborate on the Xiaomi Text-to-Text Simultaneous Speech Translation System for the IWSLT 2022 in this paper. We first investigate the current mainstream techniques such as deep model and R-drop to construct a relatively strong baseline model, then explore various data augmentation techniques such as TaggedBT, KD, and iterative BT to further improve the translation quality of the deep model.

Then, we adopt the efficient `wait-k` strategy and the multi-path `wait-k` strategy to improve the translation quality of the system on the streaming output text which simulates the ASR output, and design some rule-based inference algorithms to remedy the obvious translation errors under low latency.

In order to alleviate the negative impact of the noise contained in the streaming ASR output on our system, we propose two error correction methods to improve the robustness of the model, so that the system has a significant improvement on the noisy inputs.

In the future, we will explore the effect of ways to mitigate exposure bias (Zhang et al., 2019) and pre-trained models, such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020), on the text-to-text simultaneous speech translation task.

References

Antonios Anastasopoulos, Luisa Bentivogli, Marcely Z. Boito, Ondřej Bojar, Roldano Cattoni, Anna Currey, Georgiana Dinu, Kevin Duh, Maha Elbayad, Marcello Federico, Christian Federmann, Hongyu Gong, Roman Grundkiewicz, Barry Haddow, Benjamin Hsu, Dávid Javorský, Věra Kloudová, Surafel M. Lakew, Xutai Ma, Prashant Mathur, Paul McNamee, Kenton Murray, Maria Nădejde, Satoshi Nakamura, Matteo Negri, Jan Niehues, Xing Niu, Juan Pino, Elizabeth Salesky, Jiatong Shi, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Yogesh Virkar, Alex Waibel, Changhan Wang, and Shinji Watanabe. 2022. FINDINGS OF THE IWSLT 2022 EVALUATION CAMPAIGN. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, Dublin, Ireland. Association for Computational Linguistics.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Isaac Caswell, Ciprian Chelba, and David Grangier. 2019. Tagged back-translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 53–63, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2020. Efficient Wait-k Models for Simultaneous Machine Translation. In *Proc. Interspeech 2020*, pages 1461–1465.

Vu Cong Duy Hoang, Philipp Koehn, Gholamreza Haffari, and Trevor Cohn. 2018. Iterative back-translation for neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 18–24, Melbourne, Australia. Association for Computational Linguistics.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Younggun Lee, Suwon Shon, and Taesu Kim. 2018. Learning pronunciation from a foreign language in speech synthesis networks. *arXiv preprint arXiv:1811.09364*.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. R-drop: Regularized dropout for neural networks. In *Advances in Neural Information Processing Systems*.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. 2020. SIMULEVAL: An evaluation toolkit for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 144–150, Online. Association for Computational Linguistics.

Xuan-Phi Nguyen, Shafiq Joty, Kui Wu, and Ai Ti Aw. 2020. Data diversification: A simple strategy for neural machine translation. In *Advances in Neural Information Processing Systems*, volume 33, pages 10018–10029. Curran Associates, Inc.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages

86–96, Berlin, Germany. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. 2022. Deepnet: Scaling transformers to 1,000 layers. *arXiv preprint arXiv:2203.00555*.

Yiren Wang, Lijun Wu, Yingce Xia, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. 2020. Transductive ensemble learning for neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):6291–6298.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343, Florence, Italy. Association for Computational Linguistics.

Baigong Zheng, Kaibo Liu, Renjie Zheng, Mingbo Ma, Hairong Liu, and Liang Huang. 2020. Simultaneous translation policies: From fixed to adaptive. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2847–2853, Online. Association for Computational Linguistics.