This document is part of the Research and Innovation Action "European Live Translator (ELITR)". This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 825460.

Deliverable D3.2

Report 2 on Spoken Language Translation

Barry Haddow (UEDIN), Ondřej Bojar (CUNI), Dominik Macháček (CUNI), Peter Polák (CUNI), Sukanta Sen (UEDIN), Rico Sennrich (UEDIN), Felix Schneider (KIT), Joshua Wilkins (UEDIN), Biao Zhang (UEDIN)

Dissemination Level: Public

Final (Version 1.0), 31st March, 2022

Grant agreement no.	825460
Project acronym	ELITR
Project full title	European Live Translator
Type of action	Research and Innovation Action
Coordinator	doc. RNDr. Ondřej Bojar, PhD. (CUNI)
Start date, duration	1st January, 2019, 39 months
Dissemination level	Public
Contractual date of delivery	31st March, 2022
Actual date of delivery	31st March, 2022
Deliverable number	D3.2
Deliverable title	Report 2 on Spoken Language Translation
Type	Report
Status and version	Final (Version 1.0)
Number of pages	85
Contributing partners	UEDIN, CUNI, KIT
WP leader	UEDIN
Author(s)	Barry Haddow (UEDIN), Ondřej Bojar (CUNI), Dominik Macháček (CUNI), Peter Polák (CUNI), Sukanta Sen (UEDIN), Rico Sennrich (UEDIN), Felix Schneider (KIT), Joshua Wilkins (UEDIN), Biao Zhang (UEDIN)
EC project officer	Luis Eduardo Martinez Lafuente
The partners in ELITR are:	▪ Univerzita Karlova (CUNI), Czech Republic ▪ University of Edinburgh (UEDIN), United Kingdom ▪ Karlsruher Institut für Technologie (KIT), Germany ▪ PerVoice SPA (PV), Italy ▪ alfatraining Bildungszentrum GmbH (AV), Germany
Partially-participating party	▪ Nejvyšší kontrolní úřad (SAO), Czech Republic

For copies of reports, updates on project activities and other ELITR-related information, contact:

doc. RNDr. Ondřej Bojar, PhD., ÚFAL MFF UK
Malostranské náměstí 25
118 00 Praha, Czech Republic
bojar@ufal.mff.cuni.cz
Phone: +420 951 554 276
Fax: +420 257 223 293

Copies of reports and other material can also be accessed via the project's homepage:

`http://www.elitr.eu/`

Contents

1 Executive Summary

In this deliverable we describe our work on spoken language translation (SLT) in the second half of the project. SLT is a special form of machine translation (MT) which is designed to cope with spoken language. There are two paradigms for SLT – *cascade* systems where automatic speech recognition (ASR) first produces a transcription in the source language, then another system translates it to the target language; and *end-to-end* systems where the source speech is directly translated into the target language, without an intermediate transcription. Our research has been directed at improving both paradigms, although the production system in ELITR uses the cascade approach, a more mature technology.

A major consideration for the ELITR use-case is that SLT should be *online* or *simultaneous*, in other words the translation should be made available to the user as soon as possible, without excessive *latency*. This requires trade-offs in system design, because if the system produces translations too quickly, they may be of poor quality because the model does not have sufficient information to translate, or the system may need to update already-written translations, causing distracting flicker.

Much of the work in this package has therefore been directed at improving online translation. We have developed techniques that can use the source language to predict when to translate, and when to wait for more input, and we have developed data manipulation techniques that lead to more monotonic translation models, better suited to online SLT. In addition, we have been working on the problem of online SLT for continuous inputs, since most MT/SLT models are applied on a sentence-by-sentence basis. We have an approach to this problem based on sliding windows, and another approach based on a modified neural architecture.

Given that our SLT systems are aiming to complement and extend the work of human interpreters, another piece of work in this package has been using our recently developed corpus of interpretation to compare different approaches to extending interpretation. The scenario we have in mind is where there is a human simultaneous interpreter providing interpretation into one language, but we would like to extend to other target languages using SLT. We provide a comparison of applying SLT directly to the source, versus applying it to the human interpreter, and show the trade-offs.

For end-to-end models, one of their touted advantages is that by forgoing an intermediate source language transcription they can avoid committing too early and thereby propagating errors along the pipeline. The question we ask is, does this help the models be more robust to acoustic noise (as has been claimed in the literature). It turns out that a small advantage is visible, but only at high levels of acoustic noise.

Our work on improving end-to-end models has focused on multilingual (including zero-shot) modelling, and exploiting wider context. We find that we are able to improve performance on the former by using architectural improvements and data augmentation. For the latter, we are the first to show that extended context can be exploited successfully in end-to-end SLT (previous models are all sentence-by-sentence). Our final piece of work examines the typical training pipeline for end-to-end models and shows how it can be simplified.

2 Introduction

This is the final report on the research of the spoken language translation (SLT) work package (WP3). It presents the progress of the research in this work package since the previous report (D3.1) was submitted in June 2020.

From the Grant Agreement, the aim of this work package is:

> *To ensure high-quality MT for spoken language input, which lags behind the quality of text-to-text translation due to the noisy nature of ASR.*

Viewing this from a system-building perspective, WP3 is about improving the connection between ASR (as developed in WP2) and MT (as developed in WP4).

In the project plan, the work of WP3 was split into three tasks reflecting different ways of improving the ASR/MT interface. For the first two tasks we assume a "pipeline" (or "cascade") approach to SLT, where the discrete output of the ASR system is fed into the MT system. In task T3.1 (ASR transcript normalization) we develop methods for post-processing the output of the ASR so that it can be more accurately translated by a text-to-text MT system; whilst in task T3.2. we aim to make the MT system more robust to translating the output from ASR, especially when it contains errors. For task T3.3, we take a different approach, in that try to model ASR and MT jointly – in an end-to-end SLT system.

Below we give a brief explanation of how our work fits with the three tasks in this WP, and in the sections that follow we describe our progress in more detail. Since most of the work has already been described in (published or submitted) research papers, we include a summary of this work in the body of the deliverable, and add the full paper in an appendix.

T3.1: Normalization of ASR Normalisation of ASR output includes disfluency removal, punctuation, segmentation and truecasing. For these steps we have developed specialised components for our production system, and these were described in the interim report (D3.1). In this report we describe our attempts to build an MT system that can work directly on raw ASR (i.e. unpunctuated and uncased). The idea of this is to create a cascaded system that does not need a segmentation component between ASR and MT. Besides simplifying the pipeline, we hoped that this would be more robust to ASR segmentation errors (which can be very damaging to MT), especially in the online translation setting. This work is described in Section 3.1.

The other piece of work that fits into this task is error detection for ASR. The idea here is that if we can detect mis-transcriptions then we can take action either before they are passed to MT, or within MT itself. This work is described in Section 3.2.

T3.2: Robust Neural Machine Translation with Noisy Input Early in the project we realised that an important problem for the MT system was the fact that the transcriptions from ASR were delivered incrementally, and could be rewritten as the ASR component updated its hypotheses. To produce useful output for our use-case, it was essential that the translation was also output to the user in a timely fashion, without an excessive delay. In other words, the ELITR use-case required "online" or "simultaneous" SLT.

There are two basic approaches to simultaneous SLT: retranslation (Niehues et al., 2016, 2018) and streaming (Ma et al., 2019). In retranslation, the current segment is retranslated each time there is an update from ASR, whereas a streaming approach maintains the model's hidden state and on each update it decides whether to extend the translation (WRITE) or wait for more input (READ). Retranslation has the advantage of simplicity (it can work with an unmodified MT system) but it will cause changes in already-written translation output, leading to potentially annoying flicker for the users. Evaluation of retranslation thus requires us to consider three factors: quality (how good is the translation), latency (how timely does the translation appear), and flicker (how often translations are updated). In ELITR, we have developed SLTev (Ansari et al., 2021), a comprehensive evaluation tool for simultaneous SLT, which can measure all three factors, and is described in D1.6.

In Section 4 we describe several pieces of research aimed at improving simultaneous SLT, considering both streaming and retranslation approaches. In Section 4.1 we describe how retranslation can be adapted for use in a streaming step, i.e. when correcting previous output is disallowed. We then show in Section 4.2 that a self-training approach can be used to stabilise retranslation (i.e. to reduce flicker). Focusing on streaming approaches, we first show that *adaptive computation time* is an effective method and can be applied to unsegmented ASR output (Section 4.3), and then we explore how reinforcement learning and imitation learning can improve the streaming approach (Section 4.4).

Finally we consider the link between simultaneous SLT, as implemented by current systems, and human simultaneous interpreting, based on our study of the interpreters in the European Parliament (4.5).

T3.3: End-to-End Speech-to-Text Translation In the final task we consider end-to-end models for spoken language translation. We will present three research papers that have been published or submitted: using end-to-end models for multilingual and zero-shot translation scenarios (Section 5.1); including inter-sentential context in end-to-end models (Section 5.3); and an investigation into the necessity of ASR pre-training and rule-based feature extraction for end-to-end models (Section 5.4). We also present some further analysis on the robustness of end-to-end approaches to acoustic noise – preliminary results were presented at the first review, although not included in D3.1.

3 Normalisation and Segmentation of ASR

3.1 Window-based Approach to SLT

In a cascaded approach to speech translation, the speech is first transcribed using an ASR system, and then the transcription is translated by a text-to-text MT system. Before translating the ASR generated transcription, we need to sentence-segment it automatically. If the segmentation is not correct, the translation generated by the MT system may not be adequate. Also the ASR generated transcriptions often have incorrect punctuation which becomes problematic as MT systems are sensitive to noise. So we aim to build am SLT system where we do not need to rely on an automatic segmenter and ASR generated punctuation. We build a system that accepts raw ASR, i.e. lower-cased and unpunctuated, translating the text as a series of sliding windows. To avoid a mismatch between test and training, we propose a window-based training approach, where the training data is also converted to a sequence of sliding windows. This approach removes the need for segmentation and other post-processing of the ASR output before feeding to an MT system.

In our proposed window-based translation, we remove punctuation from the source text and also lowercase to simulate the ASR-generated text. After the preprocessing, we align the training data and convert it into a set of parallel windows. The windows are of 15-25 tokens in length. Once we have generated the window pairs we train a transformer-based MT system. At inference time, the ASR output is split into overlapping windows of fixed length. These windows are translated into target windows. As the windows are not sentences, and their translations overlap, we propose an algorithm to join the output windows back together into a target language stream.

We experiment using English-German and English-Czech language pairs and evaluate on the ESIC (Macháček et al., 2021) test set. We find that our proposed window-based approach outperforms the baseline approach using an automatic segmenter.

See Appendix A for full details.

3.2 Error Detection in ASR

If we can detect errors in ASR, then we could alert users to potential errors when we display the results of transcription/translation. We could also use this information to make the downstream

MT more robust to ASE errors.

Using large pretrained language models for downstream tasks has recently become popular. We were interested in whether it is possible to leverage such a model to identify errors in ASR transcripts, i.e., predict a correct/incorrect label for each word in the ASR transcript. As a pretrained language model, we use BERT (Devlin et al., 2018). The model is extended with a linear layer on top of the hidden states for token label prediction. We leverage an English speech recognition dataset Mozilla CommonVoice (Ardila et al., 2019). Using a pretrained ASR model Conformer (Gulati et al., 2020), we transcribed the recordings from the dataset. Using the word-level alignment of ASR and golden transcripts, we obtain correct/incorrect labels for each word in the ASR transcript (i.e., correct if the aligned golden word matches the word produced by ASR). We finetune the BERT extended with token classification head on the ASR transcripts as inputs and correctness labels as targets.

The results show an F1 score of 0.91 with a precision of 0.94 and a recall of 0.87. We identified two major drawbacks of the model. (1) Because the ASR error classifier does not have access to the original recording, it judges the correctness only based on the context of the utterance. (2) We found the classifier to have a problem with domain shift, e.g., identifying direct speech within reported speech sentences as incorrect. The pretraining uses data in the written domain, unlike the ASR transcripts that are from the speech domain.

In conclusion, the pretrained language models can be used for ASR error detection. However, the lack of the prior speech utterance for context, and pretraining domain shift lead to suboptimal results.

4 Online Spoken Language Translation

4.1 Using a Retranslation Approach in a Streaming Setting: IWSLT 2021 Submission

In the IWSLT 2021 simultaneous spoken language translation task (Anastasopoulos et al., 2021), the idea was to create systems that would produce the best possible translation at specified latency settings. We took part in the text-to-text English→German track where participants had to submit a dockerised model (or models) whose latency, as measured by Average Lagging (Ma et al., 2019) on the development set was either ≤ 3 (low), ≤ 6 (medium) or ≤ 15 (high). These models were then evaluated on a blind test set to provide quality measurements for each of the latency categories. Note that the evaluation process only supported streaming systems, so systems were not allowed to update translations after outputting them.

Our aim in building our submission (Sen et al., 2021) was to test how well a retranslation system could be adapted to a streaming scenario. The advantage of this approach is simplicity, although we do require some extra heuristics to ensure good quality. Since a retranslation system produces a fresh translation of the source sentence prefix every time the source is extended, there is a danger that translations of later prefixes are inconsistent with earlier ones that have already been committed. The simplest way to address this problem is to mask the last k words of the output on each update, reducing the likelihood of inconsistency, but this increases latency. In our submission we tested two methods of improving over this masking baseline: setting the mask dynamically by probing the translation of predicted extensions Yao and Haddow (2020); and using a source language model to predict when "meaningful units" of source text were completed. Both of these techniques produced gains (in terms of an improved quality-latency tradeoff) over the fixed mask baseline. The full paper is included in Appendix B.

4.2 Reducing Flicker in Retranslation-based Simultaneous SLT Using Self-training

Flicker in retranslation systems can be related to differences in word order between source and target languages, or simply by the system trying to make lexical decisions based on incomplete knowledge, and then having to update. In some cases this non-monotonicity or indecision is necessary, but we would ideally like to reduce it as much as possible, whilst maintaining quality.

In other words, a system that prefers monotonic translations where possible, and tends to choose a single best translation and stick to it, should have lower flicker.

In non-autoregressive translation a related problem is that of multimodality (Gu et al., 2018; Zhou et al., 2020). This is when the system is forced to choose between two different translations, of similar likelihood, resulting in an incoherent combination of the two. The solution in that case is to use sequence-level knowledge distillation (Kim and Rush, 2016), which is approximated as self-training. Using this inspiration, we applied self-training to the problem of restranslation-based simultaneous SLT. This works by first translating the whole training corpus, selecting the highest scoring translation in the beam for each source sentence, and creating a synthetic corpus for another round of training. The resulting MT system has similar quality, but reduced flicker for the equivalent latency (we use fixed masks to manipulate latency). The full paper is included in Appendix C.

4.3 Translating Streams Using Adaptive Computation Time

In this work (Schneider and Waibel, 2020), we turn our attention to streaming-based simultaneous SLT, where the system needs a policy to decide when to read more input, and when to extend the translated output. An ideal policy would offer an optimal tradeoff between quality and latency. A fixed policy such as wait-k (Ma et al., 2019) is relatively straightforward but the one-size-fits-all approach is not optimal, so researchers have investigated learned policies where the attention mechanism is used to make the read/write decision, e.g. MILk (Arivazhagan et al., 2019). However these attention-based learned systems suffer from numerical instability, and a weakness of most previous work is that it assumes that the source text is already segmented. As in Section 3.1, we would like to be able to run without a segmenter in order to simplify the setup, and as a step towards end-to-end simultaneous SLT.

An alternative way to learn a read/write policy is to apply *adaptive computation time* (ACT; Graves, 2016), and this can be extended to long sequences using the Transformer-XL (Dai et al., 2019). The original ACT was applied to RNNs, and allowed the network to "ponder" the input for several timesteps. We extended this to the case of an encoder-decoder, by allowing the decoder to "ponder" the input from the encoder for zero or more steps. Using ACT in combination with the Transformer-XL enables us to achieve a better quality-latency trade-off than MILk, and to process long input sequences without a separate segmentation model. The full paper is included in Appendix D.

4.4 Applying Reinforcement Learning to Simultaneous SLT

4.4.1 Introduction

Using reinforcement learning (RL) to learn adaptive policies for online text-to-text translation has been shown to provide better trade-offs between latency and translation quality than various fixed policy approaches (wait-if-worse and Oda et al.'s segmentation algorithm; Gu et al., 2017; Alinejad et al., 2018). However, despite the positive results, little has been done to explore RL's potential for SLT further. Our work looks to bridge this gap by applying adaptive RL policies to transformer models and evaluating their performance against state-of-the-art policies for text-to-text SLT.

4.4.2 Model

We now outline our simultaneous MT system that uses an adaptive policy learned via RL. We consider an RL agent operating in an *environment*, making decisions about specific *actions*. For the purposes of training the RL model, the MT model is fixed.

Actions: READ/WRITE. In other words, the agent has to make sequential decisions between two possible actions – either it can "read" more source, or it can "write" an update to the translation.

Environment: The RL environment consists of three main parts. Firstly, the target and source

sentences are updated based on the agent's action. Next, the rewards are calculated, and finally, the observation for the next timestep is generated from the MT model.

The environment receives one of two actions, READ or WRITE, per time step from the agent. READ unveils the next word within the source sentence, while WRITE outputs the current predicted target word to the target sentence. Rewards are calculated based on the agent's action. The rewards are designed to balance the translation quality against the latency and can be tuned to optimise for one over the other to allow for more or less latency. We perform the reward calculations according to Gu et al. (2017), that is, by summing the change in BLEU r^Q with the latency r^L at each time-step. We use a smoothed version of sentence BLEU to measure the quality of translations (Chen and Cherry, 2014), and for each time-step calculate the change in BLEU as the reward. Finally, once the target sentence is complete, we employ a brevity penalty, BP, to penalise sentences that are too short compared to the reference.

$$r_t^Q = \begin{cases} \Delta\text{BLEU}(Y, Y^*, t) & t < T \\ BP \cdot \text{BLEU}(Y, Y^*) & t = T, \end{cases} \tag{1}$$

where Y is the current target output for time t, T is the target length, Y^* is either the reference sentence or offline translation depending on the experiment setup, and $\Delta\text{BLEU}(Y, Y^*, t) = \text{BLEU}(Y, Y^*, t) - \text{BLEU}(Y, Y^*, t-1)$.

The latency is taken as the consecutive wait (CW) at each time-step, until the final time-step where we also include the average proportion (AP) for the entire episode (Cho and Esipova, 2016) in a weighted sum with the CW.

$$r_t^L = \begin{cases} \alpha[\text{sgn}(c_t - c^*)] & t < T \\ \alpha[\text{sgn}(c_t - c^*)] + \beta\lfloor d - d^* \rfloor_+, & t = T, \end{cases} \tag{2}$$

where α, $\beta \leq 0$. c_t is the CW for time-step t and d is the AP, calculated at the end of the episode. c^* and d^* are the target latency values that can be adjusted to change how much latency the model accepts. sgn is the sign function, required in order to keep the latency reward within the same range as the BLEU score ± 1.

Finally, as shown in figure 1, the environment generates the next observation from the MT model by concatenating the decoder's self attention vector, cross attention vector and hidden state with the next predicted target word embedding. Therefore, following every READ action the MT model re-encodes the available source sentence passing it to the decoder and after every WRITE action the MT model simply runs the decoder inputting the newly written target word as the previously outputted token.

Agent: Receives the current observation, reward for the previous action and information on whether the target sentence has been entirely written or not from the environment. Given the observation, the agent then makes its next decision as to whether READ or WRITE. This is achieved by passing the observation vector to a policy network. We ran experiments with different policy networks and discuss these within the experiments section below. The weights of the policy network are updated during training using the Advantage Actor-Critic (A2C) algorithm (Mnih et al., 2016). The idea is to update the weights of the policy network in a direction that increases the expected reward. The underlying theorem that enables this update is called the *policy gradient theorem* (Sutton et al., 1999), we give the result below:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a \mid s) Q^{\pi_\theta}(s, a)\right] \tag{3}$$

where $\nabla J(\theta) \in \mathbb{R}^{d'}$ is the gradient of an arbitrary scalar performance measure (i.e. the expectation of the reward) with respect to θ, π_θ represents the policy network with weights θ, and Q^{π_θ} is the state-action value function. The result shows that moving the policy network's weights in the direction that increases the probability of actions with the highest reward in a given state, will also move weights in direction that maximally increases the total expected reward, $J(\theta)$).

Previous work on RL for simultaneous translation used the REINFORCE algorithm, a vanilla implementation of the *policy gradient theorem*, to learn their policies (Gu et al., 2017; Alinejad

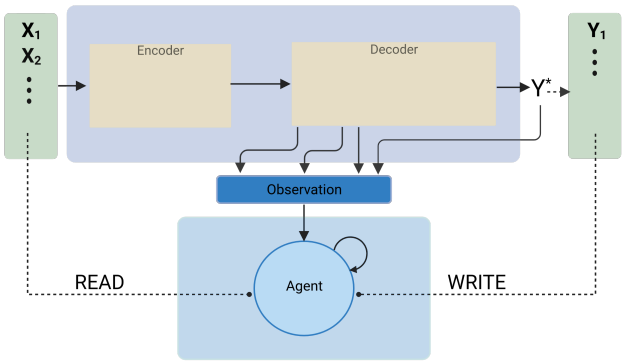


Figure 1: An illustration of the proposed model: at each time step the agent (bottom) receives an observation from the NMT environment (top) and makes a decison to either read another source word, X_i, or write the predicted target word, Y^*, to the target sentence.

et al., 2018). However, the REINFORCE algorithm requires the completion of a target sentence (i.e. an episode) before the policy network can be updated. That is, instead of using the expected state-action value function, $Q^{\pi_\theta}(s, a)$, the REINFORCE algorithm uses the observed complete returns, $G_t = r_t + \gamma r_{t+1} + \cdots + \gamma^T r_T$ (where T is the termination time-step for the episode). Updating at the end of the sentence means sampling complete action sequences from the policy network, which causes higher variance within the updates and, therefore, slower convergence to an optimal policy (Williams, 1992; Sehnke et al., 2010; Munos, 2006). Additionally, waiting until the end of the sentence creates a need for sentence boundaries, which would be a challenge when handling continuous speech input. Gu et al. (2017) included a baseline within their policy updates to reduce the variance, however, this does not completely solve the above mentioned issues with REINFORCE as it still requires waiting until the completion of a target sentence before updating. Therefore, we experiment with a policy gradient update called the actor-critic method that does not require the complete episodic sampling used by the REINFORCE algorithm.

Actor-critic methods employ an additional network known as the critic during the policy updates. As opposed to sampling complete episode returns like the REINFORCE algorithm, actor-critic methods estimate the expected returns with the critic network, V_ω, and use the error within the estimation for the policy updates. The error in the estimation of the expected return is called the temporal difference (TD) target. The TD target is the difference between the estimated expected return for some state s_t and the n-step complete return, $G_{t:t+n}$, where the first n steps are observed and the remaining are estimated. The idea behind the TD target is that the observed return from a state plus a later estimate gives a more accurate indication of what the actual expected return from state s_t is, and so the taking the difference between the less accurate estimate $v(s_t)$ and better estimate $G_{t:t+n}$ gives a target that we can move our current estimate towards. We write the n-step TD target as follows:

$$\delta_t = G_{t:t+n} - V_\omega(s_t) \tag{4}$$

$$= r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} V_\omega(s_{t+n}) - V_\omega(s_t) \tag{5}$$

where $V_\omega(s)$ is the estimate of the expected return from state s by the critic network with weights ω.

We can now write the result of the *policy gradient theorem* with the TD target instead, enabling the policy network to be updated after each time-step rather than at the end of each

episode.

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\nabla_\theta \pi_\theta(a_t \mid s_t) \delta_t \right] \tag{6}$$

Therefore, by the *policy gradient theorem*, increasing the weights of the agent's policy network in the direction of the target TD at each time-step allows us to increase the reward received at each time-step. By the design of the rewards, the translation quality is increased while the latency is kept to a minimum.

4.4.3 Experiments

The MT model consists of a 5-layer transformer encoder and a 5-layer transformer decoder, pre-trained on the WMT 19 parallel corpus (Ng et al., 2019). We used the fairseq library for the transformer implementation (Ott et al., 2019), and kept the MT model fixed during training. To implement the RL algorithms for our decision agent, we used the Stable Baselines3 library for the FFN policies and the Pytorch Implementations library for the RNN policies (Raffin et al., 2021; Kostrikov, 2018). Facilitating the connection between our MT environment and the RL algorithms within the two libraries was the OpenAI Gym API (Brockman et al., 2016). With the use of Gym method calls (step, reset, etc.), our environment could receive the actions from the RL agents and communicate back the observations and rewards in a consistent manner throughout training. We have made our code available for those interested in better understanding the implementation details[1].

We trained the RL model using two different types of data. Firstly, we experimented with using offline translations as the reference for the bleu score. Following this we also trained the RL agent with the human references from the WMT 19 parallel corpus. With the latter we expected the RL agent to simply find the best possible sequence of reads and writes to attain the offline translation. The latter however, provides an interesting question as to whether the RL agent would be able to learn to translate from the partially read source sentence into a target sentence that is still understood by humans but not a direct translation to save time on the delay of reading.

Our main aim through the research project was to achieve an RL setup that would work sufficiently with any given transformer MT model for simultaneous text to text tasks. Furthermore, following the success of previous work (e.g. Gu et al., 2017; Alinejad et al., 2018) we hoped that implementing an RL agent with a transformer MT model should allow for higher quality simultaneous translation model given the performance increase of transformers over RNNs on language translation tasks (Vaswani et al., 2017). In addition to this, we also looked to extend previous work by implementing less variant and more appropriate RL algorithms, experimenting with imitation learning, including additional actions (such as a re-translation), and experimenting with the design of the reward.

We initially started with a similar model within previous work (Gu et al., 2017); only we replaced the RNN MT model with the transformer MT model described above. However, simply replacing the RNN with a transformer proved to be insufficient. The recurrent nature of the RNN MT model allows for the agent's action history to be stored within the current observation, whereas, for the transformer MT model, this is not the case. For example, if the agent were to start by reading twice and then writing, within the RNN MT model, the current decoder hidden state would contain the previous attentions defined by the actions at those time steps. However, the transformer decoder cross attention would only contain the currently read source sentence. Therefore, our initial model did not uphold the required Markov property for RL.

To tackle the above issue, we tested several different versions of the model to allow the Markov property to be upheld with the transformer MT model. These included alternative observations and policies networks. For example, the cross attention within the decoder, at time step t, only contains information about the previously generated target word. Therefore, we needed to use the decoder's self-attention vector within the observation as it contains information

[1] `https://github.com/Joshwlks/RL-SLT/tree/main/fairseq/RL`

on the complete target sentence generated until that point. Furthermore, we experimented with a recurrent policy network that would allow the RL agent to keep track of the history of actions. Unfortunately, the results from these experiments did not show significant improvement in the agent's learned behaviour. An additional idea was to include an attention mechanism within the RNN policy to help the agent keep track of the most important; however, there was not enough time to implement and test this idea.

4.4.4 Conclusion

At the time this deliverable was prepared, we had not yet seen positive results from our RL implementation. As we explain above, we think that the architecture used for an RNN MT system cannot just be lifted directly to a transformer-based system. However, we believe that further work to design an observation that upholds the Markov property with the transformer MT model would achieve positive results given that the idea has been shown to work with RNN systems in previous works. Furthermore, future work could also explore the inclusion of additional actions such as an option for the agent to re-translate already outputted target words.

4.5 Extending Human Interpreting with SLT

In Macháček et al. (2021), we describe ESIC (European Simultaneous Interpreting Corpus), and present some experiments in using SLT to extend simultaneous interpretation. ESIC is a corpus derived from the proceedings of the European parliament, consisting of original speech and its interpretations, transcribed. This paper is included, and fully described, in D4.3, and also referred to in D1.5. Here we discuss the relevance of the paper to SLT.

We perform some experiments using the ESIC corpus, to examine ways of extending human interpreting using SLT. In our experiments, we assume that there is already a human interpreter providing simultaneous interpretation (SI) of original English speech into German, and we would like to also have SI into Czech. We consider three possible methods: (i) adding an additional human interpreter for Czech; (ii) using MT to translate from the German interpretation into Czech; or (iii) using SLT to translate from the original source into Czech. The model for (iii) is biased towards producing short translations, in an effort to reproduce the shortening often applied by human interpreters.

The evaluation using BLEU is unable to distinguish between the two automatic options, but that using the indirect option (ii) does increase latency, similar to relay interpreting. However the indirect approach resulted in shorter targets with simpler vocabulary, albeit with some evidence of information loss.

For full details, refer to D4.3, where the paper is also included.

5 End-to-End SLT

5.1 Multilingual End-to-end Models: IWSLT 2021 Submission

In the IWSLT multilingual speech translation task (Anastasopoulos et al., 2021), the idea was to translate speech in 4 languages (English, French, Portuguese and Italian) into text in 5 languages (English, Spanish, French, Portuguese and Italian). Some of the directions in the matrix were *zero-shot*, in the sense that no translations were provided in the training data – but transcriptions were provided for all source languages.

In our submission, we built end-to-end SLT systems for the constrained task setting, using high capacity models, and techniques to encourage transfer learning. To enable deeper models, we applied *depth scaled initialisation*(Zhang and Sennrich, 2019; Zhang et al., 2020b), which has previously been shown to allow transformer depths of up to 30 layers. For this shared task, we used this technique to allow the use of a transformer-big model to fully exploit the multilingual data.

In order to increase the amount of data available for training, and to provide data for the zero-short directions, we used data augmentation techniques similar to Zhang et al. (2020b). Concretely, we used a multilingual MT model to translate each source transcript into all other languages, and used these translations as the target side of a synthetic speech-to-text corpus.

We combined these techniques with adaptive feature selection (AFS; Zhang et al., 2020a) to produce an end-to-end multilingual SLT system which outperforms the organisers' baseline by over 15 BLEU on average, and outperforms our cascade by more than 2 BLEU.

The full paper Zhang and Sennrich (2021) is included in Appendix E.

5.2 Robustness of End-to-end Models to Acoustic Noise

5.2.1 Introduction

End-to-end (direct) systems have the advantage of simplicity (one model instead two) but an additional, often-claimed advantage is their lack of error propagation. In a cascade system, the output from ASR is generally a single transcription, but if this transcription is incorrect, then the subsequent MT component has no means to recover. When using an end-to-end model, however, the system does not have to commit to a single source-language transcription.

But whilst the literature on end-to-end SLT generally touts this lack of error propagation as an advantage, it is hard to find an explicit statement of what aspects of SLT are expected to benefit from it. In their review of end-to-end SLT research, Sperber and Paulik (2020) state that:

> ... models that do not suffer from erroneous early decisions will expectedly exhibit an advantage over other models especially for acoustically challenging inputs, and less so for inputs with clean acoustics.

So end-to-end models should perform better than "loosely coupled cascades" (Sperber and Paulik, 2020) on "acoustically challenging inputs". In Sperber et al. (2019), the authors compared a cascade system with an alternative that passed a continuous representation from the ASR model to the MT model, and found that the latter was less sensitive to simulated ASR errors.

Bentivogli et al. (2021) noted that end-to-end SLT systems promised "higher robustness to error propagation", and followed this up with an examination of "errors due to wrong audio understanding", implying that they expect such errors to be reduced in end-to-end models. They acknowledge that such errors are harder to spot in end-to-end systems, and analyse them using human evaluators. They claim that the number of "audio understanding errors" and the number of sentences they affect is "significantly lower" for direct models, although they do not do any statistical significance test, and they note that their analysis is "far from conclusive".

So does this lack of error propagation for end-to-end systems manifest itself as an increased robustness to acoustic noise? In order to investigate this, we designed a set of experiments where we manipulate the level of noise on the input audio, and compare the effect of this input noise on both direct and end-to-end systems.

5.2.2 Experiments

For these experiments we use the Must-C v1 (Di Gangi et al., 2019b) corpus and the fairseq (Ott et al., 2019) implementation of end-to-end SLT (Wang et al., 2020), as described in the Must-C example[2]. We use the `s2t_transformer_s` architecture, which consists of 2 convolutional layers, followed by a 12-layer transformer encoder Vaswani et al. (2017) and a 6 layer transformer decoder. For the end-to-end models we use ASR pre-training, i.e. we train an ASR system on the source transcripts using the same architecture, and then initialise the encoder of the SLT model using the encoder of the ASR model (averaged over the last 10 checkpoints of the ASR training run). For the ASR model, we preprocess the transcripts using a sentencepiece unigram

[2] `https://github.com/pytorch/fairseq/blob/main/examples/speech_to_text/docs/mustc_example.md`

model (Kudo and Richardson, 2018) and a vocabulary size of 5000, whereas we use a vocabulary size of 8000 for SLT training. For audio preprocessing, we use the scripts supplied with fairseq for preprocessing Must-C data.

For our cascade system, we use the same ASR as above, plus an MT system trained on the text-to-text data in Must-C. The MT system uses the `transformer` architecture of fairseq, and again we prepare data with sentencepiece unigram and a vocabulary size of 8000.

To introduce the noise, we apply echo, reverb and whitenoise to the audio files using the sox[3] tool. We use the high, medium and low settings from Cortès (2020), as given in Table 1.

	echo	reverb	whitenoise
low	1 0.8 150 0.2	0.5 0.5 1 1 0 2	0.02
medium	1 0.8 150 0.4	0.5 0.5 1 1 0 4	0.04
high	1 0.8 150 0.6	0.5 0.5 1 1 0 6	0.06

Table 1: The sox parameters used for adding noise to the audio signal. For whitenoise, we provide the volume, for the other effects, we provide all parameters used, and refer to the sox manual for details.

For each condition (pipeline, end-to-end) we train using four different types of noised audio (base – no noise, low, medium and high), and test the resulting models on the same four different types of noise applied to the test set. We use the tst-COMMON set from Must-C, and repeat the experiment for the 8 language pairs of Must-C, i.e. English to Dutch, French, German, Italian, Portuguese, Romanian, Russian and Spanish.

We show the complete results in Figures 2 and 3. The first thing to notice is that we observe a very similar pattern of results across all language pairs. Since the audio is virtually the same for all pairs, this indicates that it is the audio processing that has the main effect (and not the target language). In absolute terms, the end-to-end scores are always lower than the pipeline scores, but the systems in these experiments do not use any data outside of Must-C, and are not heavily optimised, as achieving state-of-the-art results was not our aim here.

On the main question, of whether end-to-end systems are less affected by acoustically challenging inputs, the evidence is not so easy to interpret. If we look at the graphs in the left-hand column of the two figures (where we train on the base, un-noised corpus), then we see that increasing the test noise does bring the two lines closer together. In other words, the advantage of our pipeline over end-to-end is reduced for noisy input. However if we look at the systems that were trained on noisy audio (in the right-most two columns of the figures) the gap between pipeline and end-to-end is larger, and changes little as we increase test set noise.

In order to get an alternative view on the results, we consider the BLEU difference between end-to-end and pipeline, and average the difference across all 8 language pairs. The results are shown in a heatmap in Figure 4, where we plot the BLEU difference against the different test and train conditions. The heatmap also suggests that at high test-time noise levels, our end-to-end systems are able to close the gap on pipeline systems, but that training end-to-end systems on noisy audio increases the gap.

5.2.3 Conclusion

We have compared a set of end-to-end SLT systems with a set of pipeline (cascade) systems trained on the same data. We find that the lack of error propogation in end-to-end systems may provide an advantage on acoustically confusable inputs, albeit a fairly small advantage, that is only visible at high levels of acoustic noise.

[3] `http://sox.sourceforge.net/`

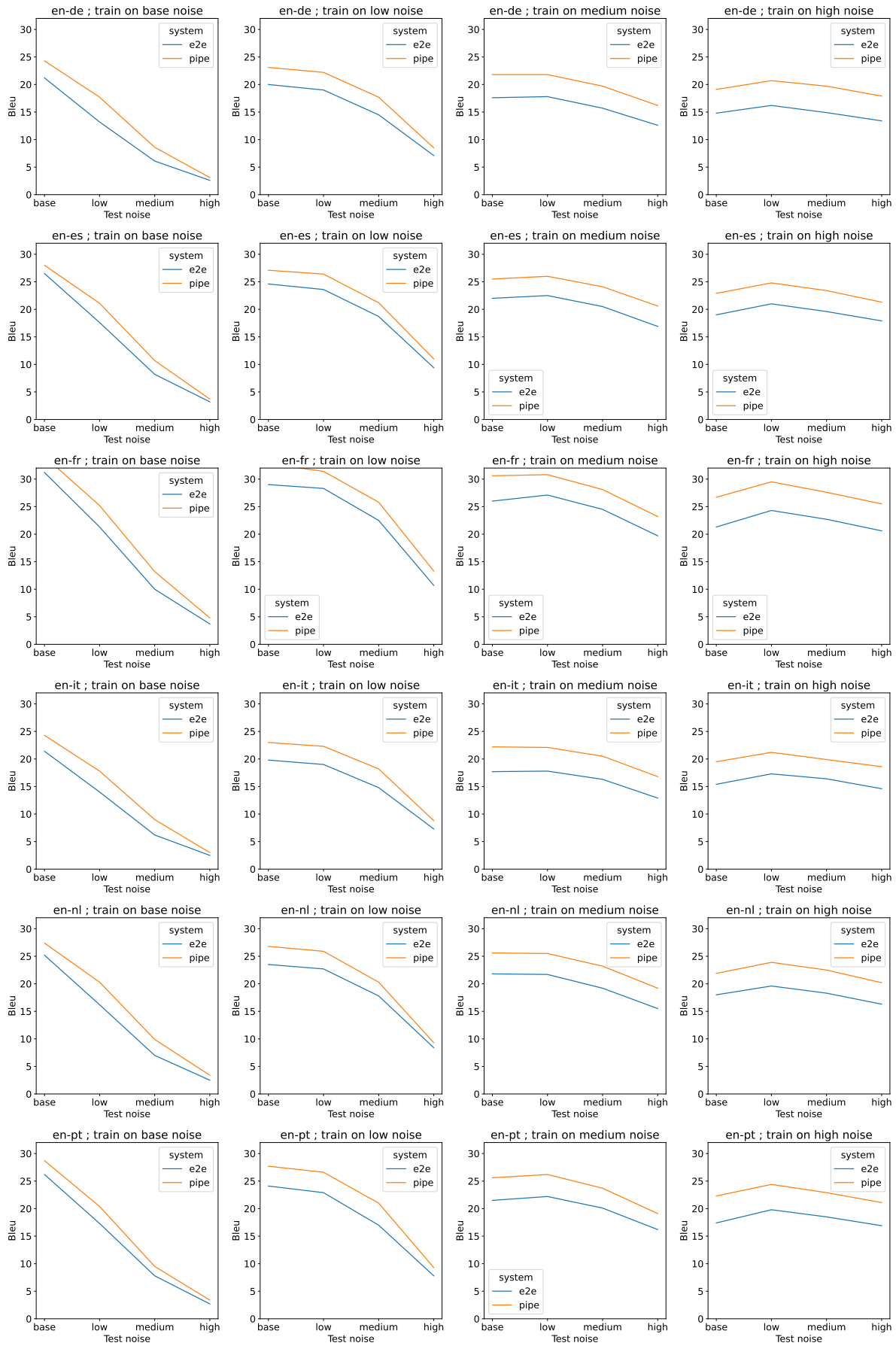


Figure 2: Comparison of the effect of training and test noise on end-to-end and pipeline (cascade) SLT systems. This shows en→{de,es,fr,it,nl,pt}

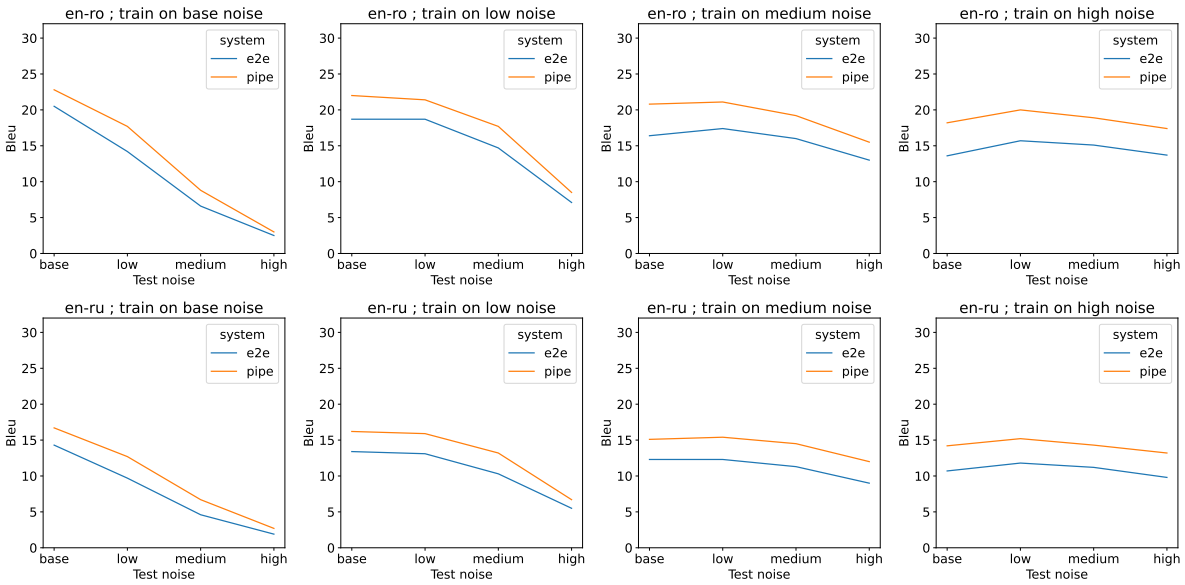


Figure 3: Comparison of the effect of training and test noise on end-to-end and pipeline (cascade) SLT systems.This shows en→{ro,ru}

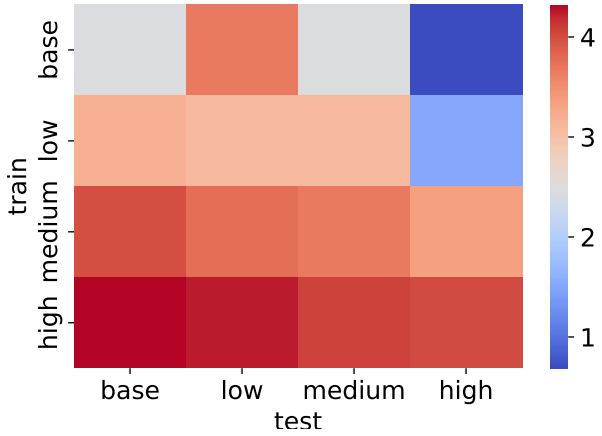


Figure 4: Mean bleu advantage of pipeline over end-to-end models, at different noise levels in training and test. The mean is over all language pairs in Must-C.

5.3 Including Wider Context in End-to-end Models

In (Zhang et al., 2021) we consider the problem of how to exploit inter-sentential context in end-to-end models. The data sets for training end-to-end models (such as Must-C) are supplied with manual segmentation in place, in other words sentence boundaries are marked both on the audio source and the textual target. Systems are generally trained to translate one sentence at a time, independently of other sentences. For Must-C this ignores the fact that the data sets consists of a series of talks, each consisting of several spoken sentences.

To include context in the translation, we apply the n-n model (Tiedemann and Scherrer, 2017) from text-to-text translation. In this model, instead of training the system to translate one sentence at a time, it is trained to translate n sentences into n sentences, where we set $n = 3$. In order to cope with the longer sequences required for this model, we apply adaptive feature selection (Zhang et al., 2020a) to reduce the sequence length required to represent the audio. At inference time, we found that In-model Ensemble Decoding (IMED) worked best – this is where we interpolate the context-sensitive model prediction with a prediction from the

same model, but without context.

We show that our n-n model with IMED improves performance across the Must-C language pairs, as measured by BLEU and a measure called APT which measures accuracy of pronoun translation (Miculicich Werlen and Popescu-Belis, 2017). In further experiments using dummy context we show that the model is indeed exploiting the inter-sentential context, and we also demonstrate that a context-sensitive model may improve homophone translation, retranslation-based simultaneous SLT and is more robust to segmentation errors.

The full paper is included in Appendix F.

5.4 Fully End-to-end Models Trained from Scratch

In this paper we revisit the typical training regime for end-to-end models to ask whether a more direct regime is possible. In particular, we first consider whether ASR pre-training is really necessary. In most work on end-to-end SLT, researchers first train an ASR system using source transcripts (see e.g. Di Gangi et al., 2019a), and then use this to initialise the end-to-end SLT model. We also examine the audio feature extraction pipeline, which includes the calculation of log mel-scale filterbank features (MFCC). We replace this with two feed-forward neural blocks which extract features from raw audio, and are trained along with the model.

To show that ASR pre-training is not necessary, we start from a baseline model (without pre-training), and make a series of modelling changes to improve its performance. We find that by following this process we can close the gap between the model using pre-training, and the model that does not use pre-training. We found that the most important modelling changes were (i) an improved architecture which includes a deeper encoder, larger feed-forward dimension with post-LN (layer normalisation) and depth-scaled initialisation (see Section 5.1); (ii) CTC (Graves et al., 2006) based regularisation of the encoder layer using the target text; and (iii) a parameterised distance penalty. Testing across 23 language pairs shows that these innovations allow from-scratch models to be competitive with pre-trained models in all scenarios except for very low-resource. Testing the neural acoustic feature model (NAFM) shows that it can perform equivalently to the MFCC feature extraction.

The full paper is included in Appendix G.

6 Conclusion

In this deliverable we have described the work of the SLT work package in the second half of Elitr. Our research has focused mainly on improving simultaneous (online) SLT and on improving end-to-end SLT, resulting in several publications in both areas.

Papers

The following papers have resulted from the work of WP3 in the second half of the project, i.e since July 2020. They are all available in the appendices.

- *The University of Edinburgh's Submission to the IWSLT21 Simultaneous Translation Task* Sen et al. (2021) (Published in IWSLT)
- *Towards Stream Translation: Adaptive Computation Time for Simultaneous Machine Translation* Schneider and Waibel (2020) (Published in IWSLT)
- *Knowledge Distillation Improves Stability in Retranslation-based Simultaneous Translation* (Under review in ACL Rolling Review)
- *Beyond Sentence-Level End-to-End Speech Translation: Context Helps* Zhang et al. (2021) (Published at ACL)
- *Edinburgh's End-to-End Multilingual Speech Translation System for IWSLT 2021* Zhang and Sennrich (2021) (Published at IWSLT)
- *Revisiting End-to-End Speech-to-Text Translation From Scratch* (Submitted to ICML)
- *Simultaneous Translation for Unsegmented Input: A Sliding Window Approach* (To be submitted to ACL Rolling review in April)

References

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

Antonios Anastasopoulos, Ondřej Bojar, Jacob Bremerman, Roldano Cattoni, Maha Elbayad, Marcello Federico, Xutai Ma, Satoshi Nakamura, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Alexander Waibel, Changhan Wang, and Matthew Wiesner. FINDINGS OF THE IWSLT 2021 EVALUATION CAMPAIGN. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 1–29, Bangkok, Thailand (online), August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.iwslt-1.1. URL `https://aclanthology.org/2021.iwslt-1.1`.

Ebrahim Ansari, Ondřej Bojar, Barry Haddow, and Mohammad Mahmoudi. SLTEV: Comprehensive evaluation of spoken language translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 71–79, Online, April 2021. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2021.eacl-demos.9`.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670*, 2019.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. Monotonic Infinite Lookback Attention for Simultaneous Machine Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1313–1323, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1126. URL `https://www.aclweb.org/anthology/P19-1126`.

Luisa Bentivogli, Mauro Cettolo, Marco Gaido, Alina Karakanta, Alberto Martinelli, Matteo Negri, and Marco Turchi. Cascade versus Direct Speech Translation: Do the Differences Still Make a Difference? In *Proceedings of ACL*, June 2021. URL https://arxiv.org/abs/2106.01045.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Boxing Chen and Colin Cherry. A systematic comparison of smoothing techniques for sentence-level BLEU. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 362–367, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3346. URL https://aclanthology.org/W14-3346.

Kyunghyun Cho and Masha Esipova. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*, 2016.

Guillem Cortès. Towards robust end-to-end speech translation. Master's thesis, Universitat Politècnica de Catalunya, 2020. URL http://hdl.handle.net/2117/330400.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Mattia Di Gangi, Matteo Negri, Roldano Cattoni, Roberto Dessi, and Marco Turchi. Enhancing Transformer for End-to-end Speech-to-Text Translation. In *Proceedings of MT Summit*, 2019a.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. MuST-C: a Multilingual Speech Translation Corpus. 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 2012–2017, Minneapolis, Minnesota, June 2019b. Association for Computational Linguistics. doi: 10.18653/v1/N19-1202. URL https://www.aclweb.org/anthology/N19-1202.

A. Graves, Santiago Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of ICML*, 2006. doi: 10.1145/1143844.1143891.

Alex Graves. Adaptive Computation Time for Recurrent Neural Networks. March 2016. arXiv: 1603.08983v6.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1053–1062, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-1099.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. Non-Autoregressive Neural Machine Translation. In *Proceedings of ICLR*, February 2018. URL https://openreview.net/forum?id=B1l8BtlCb.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040, 2020. doi: 10.21437/Interspeech.2020-3015. URL http://dx.doi.org/10.21437/Interspeech.2020-3015.

Yoon Kim and Alexander M. Rush. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1139. URL http://aclweb.org/anthology/D16-1139. event-place: Austin, Texas.

Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail, 2018.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://www.aclweb.org/anthology/D18-2012.

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. 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, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1289. URL https://www.aclweb.org/anthology/P19-1289.

Dominik Macháček, Žilinec, Matúš, and Bojar, Ondřej. Lost in Interpreting: Speech Translation from Source or Interpreter? In *Proceedings of Interspeech*, 2021.

Lesly Miculicich Werlen and Andrei Popescu-Belis. Using coreference links to improve Spanish-to-English machine translation. In *Proceedings of the 2nd Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2017)*, pages 30–40, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1505. URL https://aclanthology.org/W17-1505.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

Rémi Munos. Policy gradient in continuous time. *Journal of Machine Learning Research*, 7:771–791, 2006.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook FAIR's WMT19 news translation task submission. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5333. URL https://aclanthology.org/W19-5333.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. Dynamic Transcription for Low-latency Speech Translation. In *Proceedings of Interspeech*, 2016.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. Low-latency neural speech translation. In *Proceedings of Interspeech*, 2018.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Optimizing segmentation strategies for simultaneous speech translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 551–556, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2090. URL https://aclanthology.org/P14-2090.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

Felix Schneider and Alexander Waibel. Towards stream translation: Adaptive computation time for simultaneous machine translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 228–236, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.iwslt-1.28. URL https://aclanthology.org/2020.iwslt-1.28.

Frank Sehnke, Christian Osendorfer, Thomas Rückstieß, Alex Graves, Jan Peters, and Jürgen Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010.

Sukanta Sen, Ulrich Germann, and Barry Haddow. The University of Edinburgh's submission to the IWSLT21 simultaneous translation task. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 46–51, Bangkok, Thailand (online), August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.iwslt-1.4. URL https://aclanthology.org/2021.iwslt-1.4.

Matthias Sperber and Matthias Paulik. Speech Translation and the End-to-End Promise: Taking Stock of Where We Are. In *Proceedings of ACL*, April 2020.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. Attention-Passing Models for Robust and Data-Efficient End-to-End Speech Translation. *Transactions of the Association for Computational Linguistics*, 7:313–325, March 2019. doi: 10.1162/tacl_a_00270. URL https://www.aclweb.org/anthology/Q19-1020.

Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pages 1057–1063. Citeseer, 1999.

Jörg Tiedemann and Yves Scherrer. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4811. URL https://aclanthology.org/W17-4811.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. fairseq s2t: Fast speech-to-text modeling with fairseq. In *Proceedings of the 2020 Conference of the Asian Chapter of the Association for Computational Linguistics (AACL): System Demonstrations*, 2020.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Yuekun Yao and Barry Haddow. Dynamic Masking for Improved Stability in Online Spoken Language Translation. In *Proceedings of AMTA*, 2020.

Biao Zhang and Rico Sennrich. A lightweight recurrent network for sequence modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1538–1548, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1149. URL https://aclanthology.org/P19-1149.

Biao Zhang and Rico Sennrich. Edinburgh's end-to-end multilingual speech translation system for IWSLT 2021. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 160–168, Bangkok, Thailand (online), August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.iwslt-1.19. URL https://aclanthology.org/2021.iwslt-1.19.

Biao Zhang, Ivan Titov, Barry Haddow, and Rico Sennrich. Adaptive feature selection for end-to-end speech translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2533–2544, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.230. URL https://aclanthology.org/2020.findings-emnlp.230.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online, July 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.148. URL https://aclanthology.org/2020.acl-main.148.

Biao Zhang, Ivan Titov, Barry Haddow, and Rico Sennrich. Beyond sentence-level end-to-end speech translation: Context helps. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2566–2578, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.200. URL https://aclanthology.org/2021.acl-long.200.

Chunting Zhou, Jiatao Gu, and Graham Neubig. Understanding knowledge distillation in non-autoregressive machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BygFVAEKDH.

A Simultaneous Translation for Unsegmented Input: A Sliding Window Approach

Simultaneous Translation for Unsegmented Input: A Sliding Window Approach

Anonymous ACL-IJCNLP submission

Abstract

In the cascaded approach to speech translation, the quality of the final output depends a lot on the quality of automatic segmentation of the ASR output. Erroneous segmentation which happens due to poor sentence-final punctuation by the ASR system, leads to degradation in translation quality. To reduce the influence of automatic segmentation, we present a window-based approach to translate real ASR outputs in simultaneous translation without needing to rely on an automatic segmenter. We train translation models using parallel windows (instead of parallel sentences) extracted from the original training data. At test time, we translate at the window level, and join the translated windows using a simple approach to generate the final translation. Our experiments on English-German and English-Czech show we do not need to explicitly sentence-segment the ASR output, instead, a fixed length window can perform even better than segmentation, for both online and offline translation, achieving improvement of 1.3-2.0 BLEU points and greatly reducing the flicker over the baseline system.

1 Introduction

For machine translation (MT) with textual input, it is usual to segment the text into sentences before translation, with the sentences in most text types being indicated by punctuation. For spoken language translation (SLT), in contrast, the input is audio so there is no punctuation provided to assist segmentation. Furthermore, for many speech genres the input cannot easily be segmented into the type well-formed sentences found in MT training data, creating a mismatch between training and test.

In order to address the segmentation problem in SLT, systems typically include a segmentation componenet in their pipeline (e.g. Cho et al. (2017)). In other words, a typical *cascaded* SLT system consists of automatic speech recognition (ASR – which outputs lowercased, unpunctuated text) a punctuator/segmenter (which inserts punctuation and uses it to define segments) and an MT system. The segmentation can be a neural sequence-sequence model, and training data is easily synthesised from punctuated text. However adding segmentation as an extra neural model has the disadvantage of introducing and extra neural component to be managed and deployed. Furthermore, errors in segmentation have been shown to contribute significantly to overal errors in SLT (Li et al., 2021), since neural MT is known to be susceptible to noisy input (Khayrallah and Koehn, 2018).

These issues with segmentations can be exacerbated in the *online* or *simultaneous* setting. This is an important use-case for SLT where we want to produce the translations from live speech, as the speaker is talking. In order to mimimise the lag, or latency, of the translation, we would like to start translating before speaker has finished their sentence. Online low-latency ASR will typically revise its output after it has been produced, creating additional difficulties for the downstream components. In this scenario the segmentation into sentences will be more uncertain and we are faced with the choice of waiting for the input to stabilise (and so increasing latency) or translating early (and potentially introducing more errors, or having to make large-scale corrections when the ASR is extended and updated).

Our approach to translation of unsegmented speech is to use a *sliding window* approach. In this approach, we translate the ASR output as a series of overlapping windows, using a merging algorithm to turn the translated windows into a continuous stream. In order to address the train-test mismatch, we convert our sentence-aligned training data into window-window pairs, and remove punctuation and casing from the source. We explain our algorithms in detail in Section 2.

For online SLT, we use a *retranslation* approach (Niehues et al., 2016; Arivazhagan et al., 2020), where the MT system retranslates each time there is an update from ASR. This approach has the advantage that it can use standard approaches to MT inference, including beam search, and does not require a modified inference engine as in streamimg approaches (e.g. Ma et al. (2019)). Retranslation may introduce flicker, as outputs are updated, but this can be traded off with latency by masking the last k words of the outpuut (Arivazhagan et al., 2020). The sliding window approach is easily combined with retranslation to create an online SLT system which can operate on unsegmented ASR. Each time there is an update from ASR, we retranslate the last n tokens and merge the latest translation into the output stream.

We will show that our sliding window approach is able to provide quality improvements both to online and offline SLT. For the online case, our method improves the tradeoff between latency and flicker. Our experiments on English-Czech and English-German show an improvement of 1.3-2 BLEU points in quality on translation and significant reduction in flicker.

2 Window-based Translation

In this section, we describe how we convert the parallel training data into parallel windows after preprocessing, and then how we train the MT system, and finally, how we join the windows to generate the output.

2.1 Preprocessing

We process the parallel corpus before converting it to a set of source-target window pairs. We

- remove punctuation from the source sentences. To do this, we replace a punctuation (and other special characters) with a space[1] and then remove the extra spaces in a sentence.
- lowercase the source.

The objective of this step is to simulate source-side ASR output as we do not have such data.

2.2 Generating the Window Pairs for Training

To convert the parallel sentences into a set of parallel windows, we use a word-alignment based approach. First, we word-align the pre-proprocessed

[1]Otherwise, we can simply use empty string but hyphenated words become problematic.

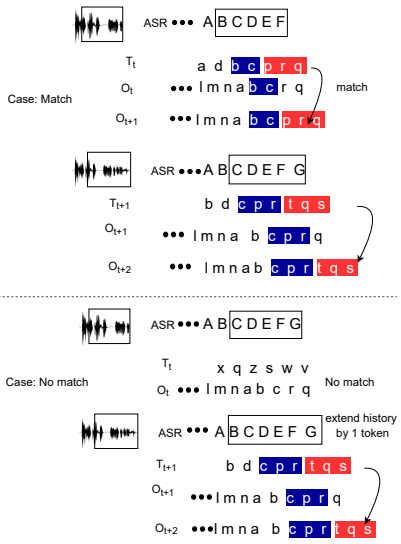


Figure 1: Example of how our proposed window-based translation works at test time in case of a match and no-match. The text inside a rectangular box is the input window at time t, which is translated (T_t) by the MT system. The text in blue shade shows the common segment between the MT output (T_t) and the output stream (O_t) at time t. The text in red shade shows the segment coming from the output window. ••• indicates there are more tokens.

parallel corpus D to obtain A using $fast\_align$ (Dyer et al., 2013). Then we concatenate all the source-target sentence pairs (s_k, t_k) into a single pair (s, t) and subsequently, revise the alignment using the Algorithm 1, so that the indexes are still correct in the concatenated corpus.

Once we have combined the parallel corpus into a pair of sentences (s, t), we use the revised alignment $A^{'}$ to generate parallel windows of length 15-25 tokens using the Algorithm 2.

Algorithm 1 Pseudo code for collapsing the parallel corpus into a pair of sentences and revising the corresponding word alignments.

Require: Parallel corpus $D = \{(s_1, t_1), (s_2, t_2), ..., (s_n, t_n)\}$, alignment $A = \{a_1, a_2, ..., a_n\}, s = \epsilon, \quad t = \epsilon$, revised alignment $A^{'} = \{\}$

for $k \leftarrow 1$ to $|D|$ **do**
  for each $i, j \in a_k$ **do**
    $i \leftarrow i + |s|$
    $j \leftarrow j + |t|$
    $A^{'} \leftarrow A^{'} \cup (i, j)$
  end for
  $s \leftarrow s + s_k$ {concatenation}
  $t \leftarrow t + t_k$ {concatenation}
end for
return $s, \quad t, \quad A^{'}$

Algorithm 2 Pseudo code for generating windows for training.

Require: Unsegmented source s, target t, and alignment $A^{'}$

Initialize: $idx \leftarrow 0$
while $idx < |t|$ **do**
  $l \leftarrow random(Window_{min}, Window_{max})$
  $W_t \leftarrow t[idx : idx + l]$ {target window}
  $p = \min_i\{(i, j) \in A^{'}, idx \leq j < idx + l\}$
  $q = \max_i\{(i, j) \in A^{'}, idx \leq j < idx + l\}$
  $W_s \leftarrow s[p : q]$ {source window}
  $idx \leftarrow idx + l$
end while

2.3 Translating Input Windows

In our simultaneous MT setting, we assume that the ASR system is transcribing incoming the speech signal into a continuous stream of text. The MT system splits the stream in to a series of overlapping fixed length sliding windows. To obtain a new input to the MT system, the window is shifted by one token to right every time the stream is extended by one or more tokens. For every input window the MT system translates it and sends it to the module that joins the output windows to the output stream. We describe this the next section.

2.4 Joining the Output Windows

Our proposed approach works at window level, where the MT system takes a source window as input and translate into an window. As the output windows are not independent units, we need to join them to a continuous stream of output. Since two consecutive input windows are overlapping, the corresponding translated windows also have a overlap. We use this overlap to join a translated window to the output stream.

We have shown the pseudo code of merging a window to the output stream in Algorithm 3. We assume that we have a stream of ASR output I which is continuously growing by one token at a time. Our algorithm requires a window length w_l and the current state of the output stream O_t. For every new token in the I, our merge module in Algorithm 3 triggers. The decoder D translates the last w_l tokens of I to a target window T_t. For any translated window T_t and output stream O_t at that time step, we find the longest continuous matching segment s. This this match can be empty sometimes. So we set a threshold r, based on which we decide whether to merge the current target window T_t or extend input window history by 1 token (to the left). In our experiments, we extend the history to maximum of 5 tokens until we have found a significant match. A higher r assures that the translation of the current window will not accidentally match a random segment in the stream, and as the successive windows are just 1 token apart, we find a match almost always. Once we have found a significant match, we merge T_t with O_t around the match, chopping the part before match. This approach of joining windows is able to handle both the online and offline situations as we are sliding a the input window by just 1 token each time.

Algorithm 3 Pseudo code for online translation – merging newly translated window into existing output.

Require: The current state of the output stream O_t, the input stream I, MT decoder D, window length w_l, threshold $r \in (0, 1)$.

$k = 0$
while true **do**
  $T_t \leftarrow D(I[|I| - (w_l + k) : |I|])$
  $O_t^{'} \leftarrow O_t[|O_t| - |T_t| : |O_t|]$
  $s, i, j \leftarrow T_t \Psi O_t^{'}$ {s is longest continuous matching subsequence. i and j are the start indices of matching in $O_t^{'}$ and T_t}
  $k \leftarrow k + 1$
  if $|s| \geq |T_t| * r$ or $k > 5$ **then**
    break
  end if
end while
$O_{t+1} \leftarrow O_t[0 : |O_t| - |T_t| + i] + T_t[j : |T_t|]$
return O_{t+1}

3 Experiments

3.1 Datasets and Experimental Settings

For training, we use parallel datasets from WMT 2020 (Barrault et al., 2020) for English-German and from WMT 2021 (Akhbardeh et al., 2021) for English-Czech. The details of the data are shown in Table 1. For the validation set, we use the concatenation of IWSLT 2014,15 test sets for English-German, and WMT 2021 development set for English-Czech. We use the ESIC test set for evaluation. ESIC is a corpus derived from the European parliament proceedings which has transcripts of source English speech and interpreted German and Czech transcripts. This test set is aligned at document level.

We use the SentencePiece (Kudo and Richardson, 2018) tokenizer for preprocessing the windows with a shared subword (Sennrich et al., 2016) vocabulary size of 32k. We train transformer-based (Vaswani et al., 2017) NMT models using Marian (Junczys-Dowmunt et al., 2018) toolkit. MT models are trained to convergence (using early stopping of 10) with a learning rate of 0.0003, and translate using a beam of 6. We train following two type of models:

- Baseline: Trained and evaluated on segmented data and evaluated on segmented data generated by the ASR system.

Corpus	Sentence pairs
English-German	
Europarl	1.79 M
Rapid	1.45 M
News Commentary	0.35 M
OpenSubtitle	22.51 M
TED corpus	206 K
MuST-C.v2	248 K
English-Czech	
Europarl	645 K
ParaCrawl	14 M
CommonCrawl	161 K
News Commentary	260 K
CzEng2.0	36 M[2]
Wikititles	410 K
Rapid	452 K

Table 1: Corpora used in training the systems

- Window: Trained on windows of 15-25 tokens and evaluated on fixed length windows.

4 Results

We evaluate both the offline and online SLT. For offline SLT, the baseline system is trained using parallel sentences, and for the online version, the baseline system is a prefix-prefix retranslation system (Niehues et al., 2016; Arivazhagan et al., 2020). For our proposed window-based system, the offline and online are the same system. We evaluate our proposed approach on simultaneous interpreted ESIC test set using Sacrebleu (Papineni et al., 2002; Post, 2018) score. As the test set is not sentence aligned, we translate each document and then align the output sentences (hypothesis) to corresponding reference document using mwerSegmenter (Matusov et al., 2005). After aligning, we calculate sentence level Sacrebleu score on the concatenation of the documents.

The baseline (*Seq*) is a segment level system, where we translate the test set at the segmented (sentence) level as generated by the ASR. For our proposed window-based method, we evaluate using different fixed-size windows of length 8, 10, 12, ..., 20 tokens. The results are shown in Table 2. From Table 2, we observe that the proposed method outperforms the baseline with significant BLEU points of 1.3 to 2. The improvement in translation quality (measured using BLEU score) perhaps happens because our approach implicitly performs an ensembling on the windows through

	Baseline		Window						
Pair	SF	SO	8	10	12	14	16	18	20
en-de	11.2	11.4	12.5	12.8	13.0	13.0	13.1	13.2	13.2
en-cs	9.4	9.4	10.0	10.2	10.4	10.5	10.6	10.6	10.7

Table 2: Comparison between segmented and window based approaches. Sacrebleu computed after sentence aligning each document using mwerSegmenter. SO: Online segment level. SF: Offline segment level.

maintaining a match (threshold) between two successive windows.

These BLEU scores in Table 2 across different window length are the best scores obtained after exploring different threshold (r) of match (refer to line 7 of Algorithm 3).We have shown the BLEU scores for each threshold in Table 3. In our experiments, we found that a threshold of 0.4 yields lowest flicker for a given window length. Also, at higher threshold, the merge operation does a lot of unnecessary retranslations (we have shown the statistics in Table 4). We can further reduce the flicker by masking out last few tokens of the output stream. We calculated the flicker after applying different masks of 1,2, ..., 10 tokens and plot them in Figure 2. We notice from Figure 2 that our proposed method outperformed the baseline for any window length with mask length > 1. The reason of having lower flicker with our proposed approach is that our approach doesn't allow updating the output beyond the last few tokens equal to number of token in the output window at any point of the merge operation. The update is restricted by the output window length.

5 Conclusion

In cascaded speech translation, the role of an automatic sentence segmenter is very important. Incorrect segmentation may results in wrong translation. In this paper, we proposed window-based approach which works at window (of fixed length of tokens) level, and removes the need of automatic sentence-segmentation of ASR output. We experiment with English-German and English-Czech language pairs and found that our proposed approach significantly performs better than the segmentation based translation obtaining an improvement of 1.3-2 BLEU points. We also observed that applying masking on the output reduced the flicker by a significant margin as compared to the baseline.

	Match Threshold (r)				
Window(w_l)	0.1	0.2	0.4	0.6	0.8
	en→de				
8	10.8	11.3	12.3	12.4	12.5
10	12.0	12.3	12.7	12.8	12.7
12	12.5	12.7	12.9	13.0	12.9
14	12.7	12.8	13.0	12.9	12.8
16	12.9	12.9	13.1	13.1	13.0
18	13.0	13.0	13.2	13.2	13.1
20	13.1	13.0	13.2	13.2	13.2
	en→cs				
8	8.3	9.1	9.8	10.0	9.9
10	9.5	9.7	10.2	10.2	10.2
12	10.0	10.2	10.4	10.4	10.4
14	10.2	10.4	10.5	10.5	10.4
16	10.5	10.6	10.6	10.6	10.5
18	10.5	10.5	10.6	10.6	10.5
20	10.5	10.6	10.7	10.5	10.5

Table 3: Results with different window length and threshold. Sacrebleu computed after sentence aligning each document using mwerSegmenter. Bleu scores in green have the lowest flickers.

	Match Threshold (r)					
w_l	0.1	0.2	0.4	0.6	0.8	#windows
	en→de					
8	1724	10513	66471	140889	200441	45879
10	1303	7352	50345	118991	185398	45497
12	956	6394	46528	110669	178805	45115
14	702	4809	42886	105207	173017	44733
16	432	4098	40447	100591	167585	44351
18	308	3809	38774	99410	163935	43969
20	215	3407	37358	96701	162025	43587
	en→cs					
8	2388	14757	74465	148605	206238	45879
10	1257	8906	53651	120135	188964	45497
12	1374	7170	44905	105294	176580	45115
14	1094	5825	40480	97436	169418	44733
16	806	4762	37067	92346	163384	44351
18	489	4118	34710	89392	158114	43969
20	292	3807	33440	87418	154827	43587

Table 4: Number of mismatches (extra retranslations due to history extension). w_l is window length.

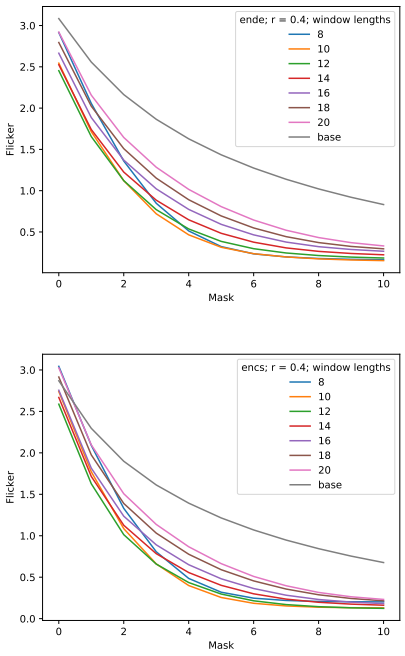


Figure 2: Mask vs Flicker plots at threshold $r = 0.4$.

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–88, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Te I, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020. Re-Translation Strategies For Long Form, Simultaneous, Spoken Language Translation. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Eunah Cho, Jan Niehues, and Alex Waibel. 2017. NMT-Based Segmentation and Punctuation Insertion for Real-Time Spoken Language Translation. *Proceedings of Interspeech*, pages 2645–2649.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Huda Khayrallah and Philipp Koehn. 2018. On the impact of various types of noise on neural machine

translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 74–83, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Daniel Li, I Te, Naveen Arivazhagan, Colin Cherry, and Dirk Padfield. 2021. Sentence boundary augmentation for neural machine translation robustness. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7553–7557. IEEE.

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.

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Hermann Ney. 2005. Evaluating machine translation output with automatic sentence segmentation. In *Proceedings of the Second International Workshop on Spoken Language Translation*, Pittsburgh, Pennsylvania, USA.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Muüller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic transcription for low-latency speech translation. In *Proceedings of Interspeech*.

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.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR*, abs/1706.03762.

B The University of Edinburgh's Submission to the IWSLT21 Simultaneous Translation Task

The University of Edinburgh's Submission to the IWSLT21 Simultaneous Translation Task

Sukanta Sen, Ulrich Germann and Barry Haddow
University of Edinburgh
{ssen, ugermann, bhaddow}@inf.ed.ac.uk

Abstract

We describe our submission to the IWSLT 2021 shared task[1] on simultaneous text-to-text English-German translation. Our system is based on the re-translation approach where the agent re-translates the whole source prefix each time it receives a new source token. This approach has the advantage of being able to use a standard neural machine translation (NMT) inference engine with beam search, however, there is a risk that incompatibility between successive re-translations will degrade the output. To improve the quality of the translations, we experiment with various approaches: we use a fixed size wait at the beginning of the sentence, we use a language model score to detect translatable units, and we apply dynamic masking to determine when the translation is unstable. We find that a combination of dynamic masking and language model score obtains the best latency-quality trade-off.

1 Introduction

In spoken language translation (SLT), there is often a need to produce translations *simultaneously*, without waiting for the speaker to finish. For example, we may be targeting live events such as conferences or meetings where excessive latency will disrupt the user experience. In order to achieve low latency SLT, however, translation systems must be able to cope well with incomplete utterances, and we find that we need to trade off latency for translation quality. In research on simultaneous SLT, we would like to understand how to produce the best possible trade-off between these two measures. In the IWSLT 2021 shared task on simultaneous translation, the aim was to build and evaluate simultaneous SLT systems at three different latency regimes (low, medium and high), as measured using the Average Lagging (AL; Ma et al. (2019)).

[1] https://iwslt.org/2021/

There are two main approaches to simultaneous translation: streaming (Cho and Esipova, 2016; Ma et al., 2019) where the system appends the output to a growing hypothesis as new inputs are available, and re-translation (Niehues et al., 2016, 2018; Arivazhagan et al., 2020a,b), where, as the name suggests, the system re-translates the whole prefix on every update to a completely new output. Re-translation approach has the advantage that we can use an unmodified, general purpose, optimised MT engine with beam-search, but we have to address the problem of *flicker*. That is to say, the translation of a prefix may be changed by the translation of an extended prefix. Recent work by Arivazhagan et al. (2020a) has shown that, if measures are taken to mitigate flicker, then re-translation produces results comparable to streaming approach. Since the shared task does not permit any revision of a committed hypothesis (i.e. flicker is not allowed) we focus on adapting the re-translation approach for our submission without introducing any flicker into a growing hypothesis.

2 Overview of Our Submission

We participated in the English→German text-to-text simultaneous task. Since we re-translate the incomplete input (know as a prefix) each time it is updated, our system will try to modify the translations produced from earlier prefixes. But as the task is evaluated using SimulEval (Ma et al., 2020) which does not permit the modification of committed output (also known as flickering), we use a simple approach to generate incremental output at each re-translation step.

Concretely, we apply a method inspired by the wait-k streaming approach (Ma et al., 2019) in our re-translation system in the following manner. In the task, a simultaneous SLT system is implemented as an agent which must choose between

Proceedings of the 18th International Conference on Spoken Language Translation, pages 46–51
Bangkok, Thailand (Online), August 5–6, 2021.

READ (read more input) and WRITE (append to the current translation hypothesis) operations. Our overall approach is shown in Algorithm 1. The agent first performs k consecutive READ operations and then alternatively READs and WRITEs until the full input sentence is read. Once the input is consumed, the agent keeps performing WRITE operations until it reaches the end of the translated sentence. The WRITE operation involves re-translating the prefix S and finding the next output word w from output prefix T. If the output prefix T has a length longer than the committed hypothesis H, it picks the $(i+1)$th word of T, else sends READ signal to the agent, i being the length of the current hypothesis.

Algorithm 1 Our Re-translation Approach

Require: NMT system ϕ, k
1: Initialize: $S \leftarrow \{\}, H \leftarrow \{\}, w \leftarrow \varepsilon$
2: **while** w is not $\langle$eos$\rangle$ **do**
3: **if** $|S| - |H| < k$ and not finished reading **then**
4: READ next input s
5: $S \leftarrow S \cup \{s\}$
6: **else**
7: $T \leftarrow \phi(S)$
8: **if** $|T| > |H|$ **then**
9: $w \leftarrow T[|H|+1]$
10: **else**
11: $w \leftarrow \varepsilon$
12: **end if**
13: **if** w is not ε or finished reading **then**
14: $H \leftarrow H \cup \{w\}$
15: WRITE w
16: **end if**
17: **end if**
18: **end while**

However, there is a potential problem with this approach. In each WRITE step, the output word w is selected from the $(|H|+1)$th position of output prefix T. Thus if any correction is made by a re-translation in the initial $|H|$ words, the WRITE operation won't be able to recover the mistake. In other words, our approach is able to suppress the flicker caused by re-translation, but could end up gluing together incompatible fragments of the hypothesis. This problem can be worse when the output prefix T flickers too much. To improve translation quality, we employ two approaches which aim at detecting meaningful units (MU) and allowing extra READs when inside an MU. An MU is a chunk of words that has a definite translation and can be translated independently without having to wait for more input words (Zhang et al., 2020).

Our first method of detecting MUs relies on the language model (LM) score. The agent keeps track of the language model (LM) score of the previous token and compares it with the score of the current token. If the LM score is higher than the previous token, it keeps reading more tokens and does a re-translation only when this condition is not met. Here the LM score is the log probability of the current token given the context. Though LM score doesn't guarantee to find meaningful unit every time but this simple approach shows it is better than the baseline approach in terms of BLEU score.

Our second method of stabilising the re-translation approach is based on the idea of dynamic masking (Yao and Haddow, 2020). The dynamic mask approach finds the stable part of the target prefix by comparing the translation of the current prefix, with the translation of an extension of the current prefix. The longest common prefix (LCP) of the two translations is taken as the stable part. Figure 1 shows how dynamic masking works in general. Yao and Haddow (2020) showed that using dynamic mask could give a better flicker-latency trade-off than using a fixed mask, without affecting the translation quality of full sentences.

For our IWSLT submission, we generate the extended prefixes for dynamic mask simply by appending *UNK* (i.e the unknown word symbol) to the prefix. In Figure 2, we show an example of how dynamic mask stabilises the translation, by masking the least stable part of the MT output. This translation-with-dynamic-mask provides a drop-in replacement for the MT system $\phi()$ in line 7 of Algorithm 1, except when the agent has read the full input sentence, when we do not need to apply any mask.

3 Experimental Details

We use only the officially allowed IWSLT 2021 data sets. The training data include high quality English-German parallel data from WMT 2020 (Barrault et al., 2020), English-German data from MuST-C.v2 (Di Gangi et al., 2019), the TED corpus (Cettolo et al., 2012) and OpenSubtitle (Lison and Tiedemann, 2016). For development, we use the concatenation of IWSLT test sets from 2014 and 2015. We test on IWSLT 2018 test set and tst-

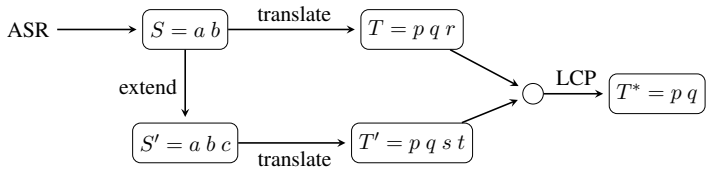


Figure 1: Dynamic Masking. The string $a\ b$ is provided as input to the agent (in a full SLT system it would come from ASR). The MT system then produces translations of the string and its extension, compares them, and outputs the longest common prefix (LCP)

	Source	Translation	MT Output
prefix	Back in New York,	Zurück in New York,	
extension	Back in New York, UNK	Damals in New York, in	
prefix	Back in New York, I	Damals in New York have ich	
extension	Back in New York, I UNK	Damals in New York war I	Damals in New York

Figure 2: An example of dynamic mask applied during translation. For the first prefix, the translation of the prefix and its extension disagree, so no output is produced (i.e. all output is masked). For the second prefix, the translation is more stable.

COMMON from MuST-C.v2. As the there is a significant overlap between MuST-C.v2 and tst-20{14,15,18}, we remove the overlaps from the MuST-C.v2 training data before training.

For preprocessing we rely only on SentencePiece tokenization (Kudo and Richardson, 2018); no other preprocessing tools are applied. We use a shared vocabulary size of 32k. Standard NMT models perform well when translation is done on a full sentence but as our approach is based on re-translation, we use training data that is a 1:1 mix of full sentences and prefix pairs (Niehues et al., 2018; Arivazhagan et al., 2020a). This ensures that our model can translate both full sentences and prefixes. To create prefix pairs, we first randomly choose a position in the source sentence and then take the proportionate length of the target sentence. Along with that we also add modified prefix pairs in which the source side has a shorter target prefix appended with the source prefix. The purpose of these modified prefix pairs was to investigate an alternative type of stabilisation, where the previous target prefix is fed into the translation of the current source prefix, but in early testing this method did not work well, so we did not pursue it further. The validation data is also pre-processed similarly to the training set. Note that this preprocessed validation set is used at training for early stopping and not for reporting the validation scores in the Table 2.

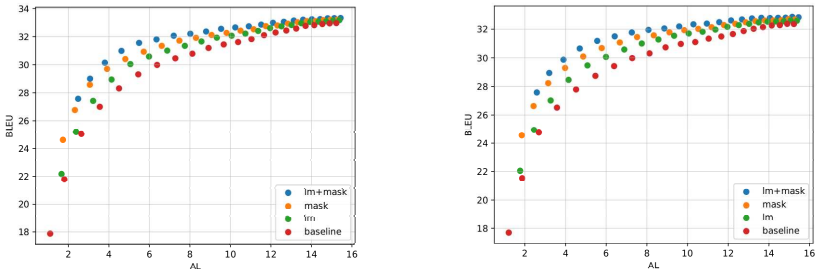
For training, we use the Marian toolkit (Junczys-Dowmunt et al., 2018) with the 'base' transformer architecture (Vaswani et al., 2017). First, we train a model using the aforementioned pre-processed training data and then fine-tune the model using MuST-C.v2 training data which is more of a domain specific data for simultaneous translation task. To train the language model for stabilisation, we use KenLM (Heafield, 2011) to train a 6-gram language model on the source-side training data. We have shown the number of sentences in each corpus in Table 1.

Corpus	Sentence pairs
Europarl	1.79 M
Rapid	1.45 M
News Commentary	0.35 M
OpenSubtitle	22.51 M
TED corpus	206 K
MuST-C.v2	248 K

Table 1: Corpora used in training the systems

4 Result and Analysis

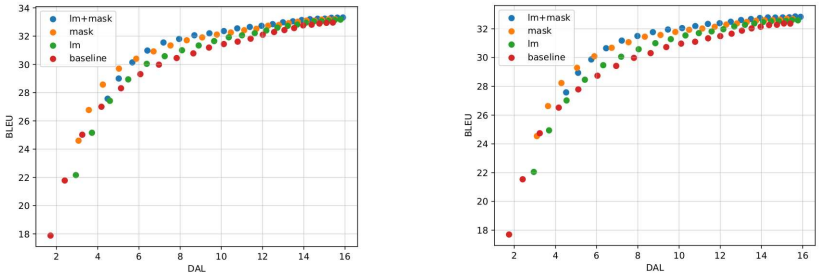
We evaluate the model's performance on the full sentence translation before doing actual simultaneous translation. For this evaluation we use SacreBLEU (Post, 2018) on the MuST-C.v2 and TED 2018 test sets. The results on full sentence is shown in the Table 2. We see there is a significant improve-



(a) Beam size = 12, Normalization = 1.0

(b) Beam size = 12, Normalization = 0.6

Figure 3: BLEU vs AL plots for English-German with different beam sizes and length normalization.



(a) Beam size = 12, Normalization = 1.0

(b) Beam size = 12, Normalization = 0.6

Figure 4: BLEU vs DAL plots for English-German with different beam sizes and length normalization.

ment after fine-tuning. For full sentence (or prefix in case of re-translation) translation we set beam size 12 and length normalization 1.0 in Marian.

	Validation	Test	
	TED 2014,15	TED 2018	MuST-C.v2
Baseline	30.8	27.5	32.7
Fine-tuned	31.9	29.4	33.6

Decoder settings: Beam size = 12; Normalization = 1.0

Table 2: BLEU scores on full sentence translation, computed with SacreBLEU.[a]

[a] BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.5.1

For evaluating the simultaneous translation, we use SimulEval (Ma et al., 2020) which calculates SacreBLEU for quality and Average Lagging (AL) (Ma et al., 2019), differential AL (DAL) (Cherry and Foster, 2019), and average proportion (AP) (Cho and Esipova, 2016) for latency. The official evaluation uses a blind test set, however, for submission purpose, we evaluate it on the MuST.v2 test set (tst-COMMON) set. We have following settings for re-translation:

Type	k	AL	BLEU	Approach
Full Sentence	-	-	33.60	-
High	20	14.73	33.09	lm
High	21	14.94	33.2	mask
High	20	14.8	**33.3**	lm+mask
Medium	6	5.98	30.58	lm
Medium	6	5.72	30.92	mask
Medium	5	5.49	**31.55**	lm+mask
Low	2	2.38	25.16	lm
Low	2	2.32	26.77	mask
Low	1	2.48	**27.57**	lm+mask

Table 3: AL vs BLEU scores for three regimes (Low, Medium, High) on MuST-C.v2 test set using beam size 12 and normalization 1.0. Best scores are in bold.

- *baseline*: The agent waits for initial k tokens and then alternates between READ and WRITE (using re-translation). This is similar to the wait-k approach by Ma et al. (2019).
- *lm*: After the initial k tokens, the agent uses the language model to determine the "mean-

ingful unit" boundaries, and only WRITEs when at a boundary.

- *mask*: This is similar to the baseline, except that the agent applies dynamic masking to produce a more stable translation.

- *lm+mask*: Combination of *lm* and *mask*. Thus in this approach, the agent first uses the *lm* score to decide whether to translate, and then uses dynamic mask to obtain a more stable translation.

The official evaluation has three regimes of latency: low (AL$\leq$ 3), medium (AL$\leq$ 6) and high (AL$\leq$ 15). In Table 3, we show the AL and BLEU scores for the three regimes with different approaches. We find that LM score and Dynamic masking combined achieve the best AL-BLEU trade-off.

To gain a fuller comparison of approaches, we calculate AL vs. BLEU and DAL vs. BLEU for a range of k values, and different stabilisation approaches and plot them as shown in Figures 3 and 4. Whilst for any given k, the *lm+mask* approach has higher AL (because it adds WAIT operations), we can see from the trajectory of the plot in Figure 3 that the *lm+mask* approach has the best AL-BLEU trade-off. While training the models, we set the length normalization to 0.6 which is used for scoring the development set for the purpose of early-stopping. However, we find that a normalization 1.0 performs slightly better than normalization 0.6 when doing re-translation. We show the plots for both normalization values in figures 3 and 4.

When the AL is 15, for many sentences it is a full sentence translation and thus all the approaches have similar BLEU scores. We also notice many sentences have negative AL scores. As the corpus AL scores is the average of the sentence level AL scores, negative scores can reduce the actual AL score. To address this shortcoming of AL, Cherry and Foster (2019), propose *Differentiable Average Lagging* (DAL) as an alternative. In Figure 4, we show the DAL vs BLEU scores. In Figure 4, we also observe that the proposed LM and masking improve the baseline by a significant margin in DAL-BLEU trade-off.

5 Conclusion

In this paper, we describe our submission to the IWSLT 2021 shared task on simultaneous text-to-text German-English translation. We work with a re-translation approach, enabling use to use an unmodified MT inference engine, together with an adaptation of wait k to trade off quality and latency. Additionally we proposed two techniques (dynamic masking and LM score) to improve translation quality by reducing the potential for flicker. We find that the combination of the proposed approaches achieves the best AL-BLEU trade-off.

Acknowledgments

This work was performed using resources provided by the Cambridge Service for Data Driven Discovery (CSD3) operated by the University of Cambridge Research Computing Service (http://www.csd3.cam.ac.uk/), provided by Dell EMC and Intel using Tier-2 funding from the Engineering and Physical Sciences Research Council (capital grant EP/P020259/1), and DiRAC funding from the Science and Technology Facilities Council (www.dirac.ac.uk).


This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreements No 825460 (Elitr) and No 825627 (European Language Grid).

References

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020a. Re-translation versus streaming for simultaneous translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 220–227, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Isabelle Te, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020b. Re-translation strategies for long form, simultaneous, spoken language translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7919–7923. IEEE.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. In *Conference of european association for machine translation*, pages 261–268.

Colin Cherry and George Foster. 2019. Thinking slow about latency evaluation for simultaneous machine translation. *arXiv preprint arXiv:1906.00048*.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Mattia A Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. Must-c: a multilingual speech translation corpus. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2012–2017. Association for Computational Linguistics.

Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Pierre Lison and Jörg Tiedemann. 2016. OpenSubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).

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 EMNLP*.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic transcription for low-latency speech translation. In *Interspeech*, pages 2513–2517.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. *arXiv preprint arXiv:1808.00491*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Yuekun Yao and Barry Haddow. 2020. Dynamic masking for improved stability in online spoken language translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 123–136, Virtual. Association for Machine Translation in the Americas.

Ruiqing Zhang, Chuanqiang Zhang, Zhongjun He, Hua Wu, and Haifeng Wang. 2020. Learning adaptive segmentation policy for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2280–2289, Online. Association for Computational Linguistics.

C Knowledge Distillation Improves Stability in Retranslation-based Simultaneous Translation

Knowledge Distillation Improves Stability in Retranslation-based Simultaneous Translation

Anonymous ACL submission

Abstract

In simultaneous translation, the *retranslation* approach has the advantage of requiring no modifications to the inference engine. However in order to reduce the undesirable instability (flicker) in the output, previous work has resorted to increasing the latency through masking, and introducing specialised inference, losing the simplicity of the approach. In this paper, we argue that the flicker is caused by both non-monotonicity of the training data, and by non-determinism of the resulting model. Both of these can be addressed using knowledge distillation. We evaluate our approach using simultaneously interpreted test sets for English-German and English-Czech and demonstrate that the distilled models have an improved flicker-latency tradeoff, with quality similar to the original.

1 Introduction

Simultaneous machine translation systems, which process their input word by word instead of sentence by sentence, must strike a balance between producing output immediately (and so reducing quality because of incomplete input) and waiting for further input (and so increasing latency). A good simultaneous translation system will provide a pareto-optimal tradeoff between quality and latency. A straightforward way of doing simultaneous translation is *retranslation* (Niehues et al., 2016), which has the advantage that it can be used with an unmodified machine translation (MT) inference engine, and can perform better than the alternative, streaming-based approaches (Arivazhagan et al., 2020b). The disadvantage is that retranslation may change previous output causing *flicker*, leading to a poor user experience, and needs to be balanced with latency and quality.

We argue that flickering is caused by two different (but related) issues: (i) instability of the translation – the system "changes its mind" as more source is revealed; (ii) non-monotonicity of the translation – the system favours a non-monotonic translation, which means it needs high latency in order to avoid flicker. Some of this instability and non-monotonicity is necessary – forced by syntactic differences between source and target, and lack of information in the prefixes – but some is due to arbitrary choices of the model and we aim to reduce these as much as possible.

Researchers in non-autogressive translation (NAT) have identified a related problem, known as the "multimodality" problem (Gu et al., 2018), where the model has two or more high scoring translations but outputs a poor quality mixture of them (because of the independence assumptions in NAT). The solution to this problem is to use sequence-level knowledge distillation (Kim and Rush, 2016), which was also shown to result in more monotonic translations (Zhou et al., 2020). In simultaneous translation, we observe a different type of multimodality (see Table 4), where the model has two competing translations (which may be synonyms) and flips between the two, unnecessarily. We therefore investigate whether the same solution as proposed there, i.e. knowledge distillation or teacher-student models, can also reduce flicker in simultaneous translation. We will show that an appropriately trained student model, in other words a model trained on a synthetic corpus created by translating using a teacher model, is able to achieve the same quality as the teacher, but with substantially lower flicker.

2 Background

We focus on simultaneous translation using the retranslation approach, and in particular how to stabilise the output, without reducing quality, and without sacrificing the simplicity of the inference.

The problem of reducing flicker was considered by Arivazhagan et al. (2020a), who showed that masking the last k words of the output, combined

with biasing the beam search towards the previously translated prefix could improve the flicker-latency tradeoff, although this required modifications to the inference engine. To set the mask dynamically, Yao and Haddow (2020) showed that the system could make predictions of the continuation of the prefix, and compare the translations of these continuations to the translations of the current prefix. However this method has the disadvantage of requiring extra translation inference, making it less efficient at runtime.

Evaluation of simultaneous translation requires that we consider more than just the quality of translation, we must also consider the latency, and if we are using retranslation, we should consider flicker. The quality of the translation can evaluated by comparing the final output of each sentence with a reference – we will show BLEU (Papineni et al., 2002; Post, 2018), CHRF (Popovi, 2015) and COMET (Rei et al., 2020) scores. For evaluation of flicker, we will use *normalised erasure* (Arivazhagan et al., 2020a), which measures the number of tokens that must be deleted from the suffix of the previous translation to produce the next, normalised by sentence length. The measurement of latency has been the subject of some debate in the literature, with several different measures proposed (Ma et al., 2019a; Cherry and Foster, 2019; Ansari et al., 2021), and for retranslation systems there is the further question of whether to use the time that a word appears, or the time that it stabilises, in the latency calculation. In our experiments, we will vary the amount of output masking, and observe the effect on flicker. The amount of masking is a clear measure of how much delay there is in the translation, and is easily controllable. The aim is to improve the mask-flicker tradeoff curve, and so be able to use a shorter mask with the same flicker budget.

In sequence-level knowledge distillation (Kim and Rush, 2016), a smaller *student* model is created using data generated by the larger *teacher* model. This has found application in MT efficiency (Junczys-Dowmunt et al., 2018), where the small size of the student models ensure that they make inference much faster, and they can also be run using a small beam. In non-autoregressive translation, teacher-student models are able to reduce the multimodality problem – by reducing the number of possible translations favoured by the model, the effect of the conditional independence assumption in NAT is mitigated (Zhou et al., 2020).

For our purposes, teacher-student methods play a similar role. Because the student model tends to prefer a single translation hypothesis, the model is less likely to swap between translation hypotheses unnecessarily as the source prefix is extended. Also, since the student model is trained on MT output, where the target order tends to be similar to the source order, the student is more likely to avoid unnecessary reorderings, generating a more monotone translation, which can be built up incrementally. We will demonstrate these points experimentally in the next section.

Recently, Chen et al. (2021) also proposed to use pseudo-reference sentences obtained through forward translation of the source sentences to improve simultaneous translation. Unlike our work, they considered a streaming approach (specifically wait-k (Ma et al., 2019b)) where the system can only append to the output, it does not flicker like retranslation. They showed that they could improve the quality-latency tradeoff of wait-k using their distillation approach, but to create the training data for the student system they used wait-k and filtering – we avoid these complications by just using the baseline system as the teacher.

3 Experiments

3.1 Data

In much of the previous work on simultaneous MT, models are evaluated on translations that were produced offline, where the translators could access the full sentence. As pointed out by Zhao et al. (2021), this may not be a realistic evaluation. So in this work, we test on the recently released ESIC corpus (Macháek et al., 2021), a corpus derived from the European parliament proceedings which contains both transcripts of the original speeches, and transcripts of the simultaneous interpretation of those speeches. ESIC also contains the corresponding text-based records, which can be considered as offline translations. ESIC is available for English→Czech and English→German, and it is aligned at the document level, but not at the sentence level. We use the test portion for evaluation.

We train our systems using offline translations, as there are no large corpora of simultaneous interpretation for training. For English→German, we use the IWSLT 2021 data sets (Anastasopoulos et al., 2021). This includes the English→German

data from WMT 2020 (Barrault et al., 2020). For development, we use the concatenation of IWSLT test sets from 2014 and 2015. We removed the train/test overlaps – between MuST-C.v2 and earlier IWSLT test sets, and between europarl and ESIC. For English→Czech, we use the training and valid set from WMT21 (Akhbardeh et al., 2021). Training data sizes are shown in Table 3.

3.2 Teacher System

Our initial system, which will later be used as a teacher model (Section 3.3), is a transformer base model[1] (Vaswani et al., 2017) trained with marian (Junczys-Dowmunt et al., 2018). We use *prefix training* to reduce the mismatch between sentence-level training data and prefix-based inference at test time (Niehues et al., 2018). For each parallel sentence pair in the training set, we generate a corresponding prefix pair by truncating using a randomly chosen proportionate length.

All data is pre-processed using a unigram language model (Kudo, 2018) with SentencePiece (Kudo and Richardson, 2018) with a shared subword (Sennrich et al., 2016) vocabulary size of 32k. We train the MT models to convergence (using early stopping of 10) with a learning rate of 0.0003, and translate using a beam of 6.

3.3 Teacher-Student Training

In order to create a more stable system, we use the teacher model in the previous section to generate training data for student models. These student models are trained in the same way, with the same architecture, but with training data synthesised by the teacher. For each source sentence, we generate n-best translations and then select the best translation that has highest score against the reference translation. In our experiments we consider 8-best translation. We use three different scores (BLEU, CHRF, and model[2] score), to select distilled training data.

In order to calculate the monotonicity of the training data, we use Kendall's tau distance. To compute the distance, we first align the parallel data using *fast_align* (Dyer et al., 2013) and then find the source permutation π of a target sentence as

$$\pi = \{j : i^{th} \text{ target word is aligned to } j^{th} \text{ source word}\}$$

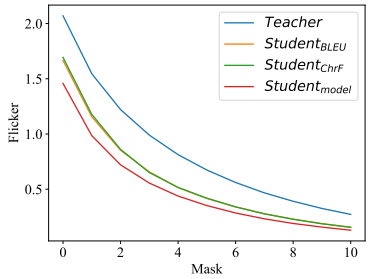
We calculate the Kendall's tau distance between π and $\pi^{'}$, where

$$\pi^{'} = \{i : i^{th} \text{ target word }\}$$

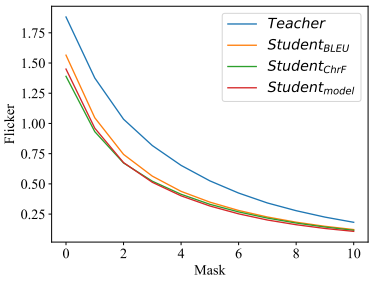
The scores are calculated at the sentence level and then averaged over a parallel corpus. The higher tau score indicates more monotonicity.

In our experiments, we find the distance between

- the source and reference (Source-Reference)
- the source and 1-best distilled target (Source-Distilled$_{\text{model}}$)
- the source and distilled target obtained from n-best using BLEU score (Source-Distilled$_{\text{BLEU}}$)
- the source and distilled target obtained from n-best using ChrF score (Source-Distilled$_{\text{ChrF}}$)



(a) En→De



(b) En→Cs

Figure 1: Sentence level Flicker vs Latency plot. The y-axis represents flicker and the x-axis represents the number of words that are masked.

[1] With 65 million parameters.

[2] For distillation using model score, we do not compare with a reference translation. Instead, each source is forward translated into the target language by the teacher model and we take the highest scoring translation.

	Model	BLEU	ChrF	COMET-qe	Flicker
			En→De		
Interpreted	Teacher	17.6	59.0	0.539	2.07
	$\text{Student}_{\text{model}}$	17.5	58.9	0.530	1.46 (29.46% ↓)
	$\text{Student}_{\text{BLEU}}$	17.6	58.9	0.527	1.67 (19.32% ↓)
	$\text{Student}_{\text{ChrF}}$	17.6	59.0	0.530	1.69 (18.35% ↓)
			En→Cs		
	Teacher	14.6	51.7	0.680	1.88
	$\text{Student}_{\text{model}}$	14.6	51.7	0.660	1.45 (22.87% ↓)
	$\text{Student}_{\text{BLEU}}$	14.6	51.7	0.670	1.56 (17.02% ↓)
	$\text{Student}_{\text{ChrF}}$	14.7	51.8	0.661	1.39 (26.06% ↓)
			En→De		
Translated	Teacher	36.4	63.7	0.540	2.61
	$\text{Student}_{\text{model}}$	36.0	63.4	0.533	1.70 (34.86% ↓)
	$\text{Student}_{\text{BLEU}}$	36.4	63.6	0.534	1.94 (25.67% ↓)
	$\text{Student}_{\text{ChrF}}$	36.6	63.9	0.532	2.02 (22.60% ↓)
			En→Cs		
	Teacher	33.9	60.0	0.721	2.33
	$\text{Student}_{\text{model}}$	33.3	59.7	0.693	1.62 (30.47% ↓)
	$\text{Student}_{\text{BLEU}}$	33.9	60.1	0.701	1.81 (22.31% ↓)
	$\text{Student}_{\text{ChrF}}$	34.0	60.2	0.694	1.66 (28.75% ↓)

Table 1: Comparison between different approaches on ESIC test set. BLEU and ChrF scores are calculated at document level for Interpreted category and at sentence level for translated category using Sacrebleu. The COMET-qe score is calculated between source and the hypothesis using reference-less *wmt20-comet-qe-da* model. We use reference-less scoring as we do not have equal number source and reference lines for interpreted ESIC corpus. The flicker scores are calculated at sentence level on outputs without any mask. In parentheses, we show relative reduction in flicker.

Model	Pair	Distance
En→De	Source-Reference	0.793
	$\text{Source-Distilled}_{\text{BLEU}}$	0.826
	$\text{Source-Distilled}_{\text{ChrF}}$	0.848
	$\text{Source-Distilled}_{\text{model}}$	0.857
En→Cs	Source-Reference	0.849
	$\text{Source-Distilled}_{\text{BLEU}}$	0.900
	$\text{Source-Distilled}_{\text{ChrF}}$	0.904
	$\text{Source-Distilled}_{\text{model}}$	0.906

Table 2: Kendall's tau distances. Higher scores indicate more monotonicity.

We have presented the tau scores in Table 2. From Table 2, we observe that the distillation makes the training data more monotonic and 1-best distilled data has the best tau distance.[3]

3.4 Stability of Student Models

We calculate the BLEU score at sentence and document level using Sacrebleu for translated and interpreted ESIC testset, respectively, and flicker at sentence level using SLTev toolkit (Ansari et al., 2021). We compare the quality of teacher and student models in Table 1.

We observe that student models have a substantially reduced flicker (by 17-34%) with no loss in either document or sentence-level BLEU or ChrF scores, although there is a moderate drop in COMET-qe. The flicker can be further reduced with masking the subsequent output prefixes. We apply different fixed mask of length 1-10 and plot the flicker (measure using normalized erasure) against each fixed mask in Figure 1. Masking helps reducing the flicker and the student models flicker less than the teacher for a given mask length. Since quality is calculated on the final output, masking does not impact BLEU/chrF/COMET.

[3] Additionally, we use tau distance to filter the 1-best distilled data, and then we train more models on the filtered data. For filtering purpose, we sort the distilled parallel corpus by monotonicity and take top 90, 80, 70, and 60% parallel sentences for training student models. But this did not reduce the flicker further significantly.

4 Conclusion

In this paper, we proposed to reduce the flicker in retranslation-based simultaneous translation through knowledge distillation. We use different metrics to select the synthetic target-side data, which are monotonic measured using Kendall's tau distance, from n-best forward translations. We use the synthetic data to train the retranslation-based simultaneous translation system. Our evaluation on interpreted testsets for English-German and English-Czech show significant reduction in the flicker with similar quality as the teacher.

References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proceedings of the Sixth Conference on Machine Translation*, pages 1–93, Online. Association for Computational Linguistics.

Antonios Anastasopoulos, Ondřej Bojar, Jacob Bremerman, Roldano Cattoni, Maha Elbayad, Marcello Federico, Xutai Ma, Satoshi Nakamura, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Alexander Waibel, Changhan Wang, and Matthew Wiesner. 2021. Findings of the IWSLT 2021 Evaluation Campaign. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 1–29, Bangkok, Thailand (online). Association for Computational Linguistics.

Ebrahim Ansari, Ondřej Bojar, Barry Haddow, and Mohammad Mahmoudi. 2021. SLTEV: Comprehensive evaluation of spoken language translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 71–79, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Te I, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020a. Re-Translation Strategies For Long Form, Simultaneous, Spoken Language Translation. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020b. Re-translation versus Streaming for Simultaneous Translation. ArXiv: 2004.03643v2.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proceedings of the Fifth Conference on Machine Translation*, pages 1–55, Online. Association for Computational Linguistics.

Junkun Chen, Renjie Zheng, Atsuhito Kita, Mingbo Ma, and Liang Huang. 2021. Improving simultaneous translation by incorporating pseudo-references with fewer reorderings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5857–5864, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Colin Cherry and George Foster. 2019. Thinking Slow about Latency Evaluation for Simultaneous Machine Translation. ArXiv: 1906.00048v1.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-Autoregressive Neural Machine Translation.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018. Marian: Cost-effective High-Quality Neural Machine Translation in C++. In *Proceedings of WNMT*.

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. Association for Computational Linguistics. Event-place: Austin, Texas.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

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. 2019a. 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.

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. 2019b. 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.

Dominik Macháek, Matú ilinec, and Ondej Bojar. 2021. Lost in Interpreting: Speech Translation from Source or Interpreter? In *Proceedings of Interspeech*.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic Transcription for Low-latency Speech Translation. In *Proceedings of Interspeech*.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. In *Proceedings of Interspeech*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. Type: Conference proceedings (article).

Maja Popovi. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR*, abs/1706.03762.

Yuekun Yao and Barry Haddow. 2020. Dynamic Masking for Improved Stability in Online Spoken Language Translation. In *Proceedings of AMTA*.

Jinming Zhao, Philip Arthur, Gholamreza Haffari, Trevor Cohn, and Ehsan Shareghi. 2021. It is Not as Good as You Think! Evaluating Simultaneous Machine Translation on Interpretation Data. *arXiv:2110.05213 [cs]*. ArXiv: 2110.05213.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Appendix

Corpus	Sentence pairs
English-German	
Europarl	1.79 M
Rapid	1.45 M
News Commentary	0.35 M
OpenSubtitle	22.51 M
TED corpus	206 K
MuST-C.v2	248 K
English-Czech	
Europarl	645 K
ParaCrawl	14 M
CommonCrawl	161 K
News Commentary	260 K
CzEng2.0	36 M[4]
Wikititles	410 K
Rapid	452 K

Table 3: Corpora used in training the systems

Source	I hope you will have a little time and energy to focus on another report which is, despite its technicality, quite important for all of us.
Traget:	Ich Ich hoffe, Ich hoffe, Sie Ich hoffe, Sie Ich hoffe, Sie haben Ich hoffe, Sie haben ein Ich hoffe, Sie werden ein wenig Zeit Ich hoffe, Sie haben etwas Zeit Ich hoffe, Sie haben etwas Zeit und Ich hoffe, Sie werden etwas Zeit und Energie haben, Ich hoffe, Sie haben etwas Zeit und Energie, um sich Ich hoffe, Sie haben etwas Zeit und Energie, um sich auf Ich hoffe, Sie werden ein wenig Zeit und Energie haben, um sich auf ein anderes Thema Ich hoffe, Sie haben etwas Zeit und Energie, um sich auf einen weiteren Bericht zu konzentrieren, Ich hoffe, Sie haben etwas Zeit und Energie, um sich auf einen anderen Bericht zu konzentrieren, ⋮ Ich hoffe, Sie werden ein wenig Zeit und Energie haben, um sich auf einen anderen Bericht zu konzentrieren, der trotz seiner Formalität für uns alle sehr wichtig ist.

Table 4: Examples of flicker caused by the teacher model. *Source* is the original full sentence which is input as a growing input prefix. *Target* is the output prefix in successive retranslations.

D Towards Stream Translation: Adaptive Computation Time for Simultaneous Machine Translation

Towards Stream Translation: Adaptive Computation Time for Simultaneous Machine Translation

Felix Schneider
Karlsruhe Institute of Technology
felix.schneider@kit.edu

Alexander Waibel
Karlsruhe Institute of Technology
alexander.waibel@kit.edu

Abstract

Simultaneous machine translation systems rely on a policy to schedule read and write operations in order to begin translating a source sentence before it is complete. In this paper, we demonstrate the use of Adaptive Computation Time (ACT) as an adaptive, learned policy for simultaneous machine translation using the transformer model and as a more numerically stable alternative to Monotonic Infinite Lookback Attention (MILk). We achieve state-of-the-art results in terms of latency-quality tradeoffs. We also propose a method to use our model on unsegmented input, i. e. without sentence boundaries, simulating the condition of translating output from automatic speech recognition. We present first benchmark results on this task.

1 Introduction

Simultaneous machine translation (MT) must accomplish two tasks: First, it must deliver correct translations on incomplete input as early as possible, i. e. before the source sentence is completely spoken. Second, in a realistic usage scenario, it must deal with unsegmented input, either speech directly or automatic transcriptions without punctuation or sentence boundaries. Until now, staged models (Niehues et al., 2016), which have a separate component to insert punctuation (Cho et al., 2012) achieved the best results in this task. In this paper, we will present the first step towards an end-to-end approach.

In recent years, a number of approaches for neural simultaneous machine translation have been proposed. They generally build on the common encoder-decoder framework (Sutskever et al., 2014), with the decoder deciding at each step whether to output a target language token based on the currently available information (WRITE) or to wait for one more encoder step in order to have more information available (READ).

In order to do this, the decoder relies on a *wait policy*. The published policies can be broadly divided into two categories:

- Fixed policies, which rely on pre-programmed rules to schedule the read and write operations, such as wait-k (Ma et al., 2019a) and wait-if (Cho and Esipova, 2016).
- Learned policies, which are trained either jointly with the translation model or separately. Examples include MILk (Arivazhagan et al., 2019) and the models of Satija and Pineau (2016) and Alinejad et al. (2018)

However, all of the above approaches train and evaluate their models on individual sentences. We want to work towards a translation system that can work on a continuous stream of input, such as text without punctuation and sentence segmentation. In a realistic usage scenario, segmentation information is not available and an end-to-end solution without a separate segmentation component is desirable. We therefore propose the use of Adaptive Computation Time (Graves, 2016) for simultaneous machine translation. This method achieves a better latency-quality trade-off than the previous best model, MILk, on segmented WMT 2014 German-to-English data. By extending this model with Transformer-XL-style memory (Dai et al., 2019), we are able to apply it directly to unsegmented text.

2 Background

As Arivazhagan et al. (2019) point out, most previous work in simultaneous machine translation focuses on segmenting continuous input into parts that can be translated, whether it is utterances speech or sentences for text (Cho et al., 2012, 2017;

Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT), pages 228–236

https://doi.org/10.18653/v1/P17

Fügen et al., 2007; Oda et al., 2014; Yarmohammadi et al., 2013). For statistical machine translations, some approaches for stream translation without segmentation were known (Kolss et al., 2008). The more recent neural simultaneous MT approaches simply take this segmentation as given and focus on translating simultaneously within a sentence.

Several approaches (Grissom II et al., 2014; Niehues et al., 2018; Alinejad et al., 2018) try to predict the whole target sentence in advance, before the input is complete. It may be possible to extend such approaches to work on an input stream, but they have the undesirable property of overriding their old output, which can make reading the translation difficult to follow for a human.

Satija and Pineau (2016) train the wait policy as an agent with reinforcement learning, considering the pre-trained and fixed MT system as part of the environment. Such an agent could learn to also predict the end of sentences and thus extend to stream translation, but it would be effectively the same as an explicit segmentation.

Cho and Esipova (2016) and Ma et al. (2019a) each define their own fixed policy for simultaneous MT. Wait-k in particular is attractive because of its simplicity and ease of training. However, we believe that for very long input streams, an adaptive policy is necessary to make sure that the decoder never "falls behind" the input stream.

Most recently, the best results are produced by monotonic attention approaches (Raffel et al., 2017; Chiu and Raffel, 2017), in particular Arivazhagan et al. (2019). Their approach uses RNNs, whereas we would like to use the state-of-the-art Transformer architecture (Vaswani et al., 2017). Unfortunately, we were unable to transfer their results to the Transformer, largely due to numerical instability problems. Ma et al. (2019b) claim to have done this, but we were unable to reproduce their results either. We therefore propose our own, more stable, architecture based on Adaptive Computation Time (ACT, Graves (2016))

3 Model

A machine translation model transforms a source sequence $\mathrm{x} = \{x_1, x_2, \dots x_{|\mathrm{x}|}\}$ into a target sequence $\mathrm{y} = \{y_1, y_2, \dots y_{|\mathrm{y}|}\}$, where, generally, $|\mathrm{x}| \neq |\mathrm{y}|$. Our model is based on the Transformer model (Vaswani et al., 2017), consisting of an encoder and a decoder. The encoder produces a vector representation for each input token, the decoder autoregressively produces the target sequence. The decoder makes use of the source information via an attention mechanism (Bahdanau et al., 2015), which calculates a context vector from the encoder hidden states.

$$h_{1\dots|\mathrm{x}|} = \textsc{Encoder}(x_{1\dots|\mathrm{x}|}) \quad (1)$$
$$c_i = \textsc{Attention}(y_{i-1}, h_{1\dots|\mathrm{x}|}) \quad (2)$$
$$y_i = \textsc{Decoder}(y_{i-1}, c_i) \quad (3)$$

In the offline case, the encoder has access to all inputs at once and the attention has access to all encoder hidden states. The standard soft attention calculates the context vector as a linear combination of all hidden states:

$$e_i^n = \textsc{Energy}(y_{i-1}, h_n) \quad (4)$$
$$w_i^n = \frac{\exp(e_i^n)}{\sum_{k=1}^{|\mathrm{x}|} \exp(e_i^k)} \quad (5)$$
$$c_i = \sum_{n=1}^{|\mathrm{x}|} w_i^k h_n \quad (6)$$

Here, Energy could be a multi-layer perceptron or, in the case of Transformer, a projection followed by a dot product.

In the simultaneous case, there are additional constraints: Each encoder state must only depend on the representations before it and the inputs up to the current one as input becomes available incrementally. In addition, we require a *wait policy* which decides in each step whether to READ another encoder state or to WRITE a decoder output. Each READ incurs a delay, but gives the decoder more information to work with. We denote the encoder step at which the policy decides to WRITE in decoder step i as $N(i)$.

$$h_j = \textsc{Encoder}(h_{j-1}, x_j) \quad (7)$$
$$a_i^n = \textsc{Policy}(y_{i-1}, h_n) \quad (8)$$
$$N(i) = \min\{n : a_i^n = \text{ WRITE }\} \quad (9)$$
$$c_i = \textsc{Attention}(y_{i-1}, h_{1\dots N(i)}) \quad (10)$$
$$y_i = \textsc{Decoder}(y_{i-1}, c_i) \quad (11)$$

Note that this kind of discrete decision-making process is not differentiable. Some approaches using reinforcement learning have been proposed

(Grissom II et al., 2014; Satija and Pineau, 2016), but we will focus on the monotonic attention approaches.

3.1 Monotonic Attention

In monotonic attention (Raffel et al., 2017), the context is exactly the encoder state at $N(i)$. Additionally, $N(i)$ increases monotonically. For each encoder and decoder step, the policy predicts p_i^n, the probability that we will WRITE at encoder step n. During inference, we simply follow this (non-differentiable) stochastic process[1]. During training, we instead train with the expected value of c_i. To that end, we calculate α_i^n, the probability that decoder step i will attend to encoder step n.

$$p_i^n = \sigma(\text{ENERGY}(s_{i-1}, h_n)) \tag{12}$$

$$a_i^n \sim \text{Bernoulli}(p_i^n) \quad \text{Inference only} \tag{13}$$

$$\alpha_i^n = p_i^n \left((1 - p_i^{n-1}) \frac{\alpha_i^{n-1}}{p_i^{n-1}} + \alpha_{i-1}^n \right) \tag{14}$$

$$c_i = \sum_{n=1}^{|\mathrm{x}|} \alpha_i^n h_n \tag{15}$$

This model needs no additional loss function besides the translation loss. It is not incentivised to READ any further than it has to because the model can only attend to one token at a time. At the same time, this is a weakness of the model, as it has access to only a very narrow portion of the input at a time.

To address this, two extensions to monotonic attention have been proposed: Monotonic Chunkwise Attention (MoChA, Chiu and Raffel (2017)) and Monotonic Infinite Lookback Attention (MILk, Arivazhagan et al. (2019)), which we will look at in more detail here.

3.2 Monotonic Infinite Lookback Attention

Monotonic Infinite Lookback Attention (MILk) combines soft and monotonic attention. The attention can look at all hidden states from the start of the input up to $N(i)$, which is determined by a monotonic attention module. The model is once again trained in expectation, with p_i^n and α_i^n calculated as in eqs. (12) and (14). The attention energies e_i^n are calculated as in equation (4).

[1] Although we encourage the model to make clear decisions by adding noise in the policy, see the original paper for more details.

$$\beta_i^n = \sum_{k=n}^{|\mathrm{x}|} \left(\frac{\alpha_i^k \exp(e_i^n)}{\sum_{l=1}^{k} \exp(e_i^l)} \right) \tag{16}$$

$$c_i = \sum_{n=1}^{|\mathrm{x}|} \beta_i^n h_n \tag{17}$$

This method does however introduce the need for a second loss function, as the monotonic attention head can simply always decide to advance to the end of the input where the soft attention can attend to the whole sequence. Therefore, in addition to the typical log-likelihood loss, the authors introduce a loss derived from $\mathrm{n} = \{N(1), \ldots N(|\mathrm{y}|)\}$, weighted by a hyperparameter λ:

$$L(\theta) = -\sum_{(\mathrm{x},\mathrm{y})} \log p(\mathrm{y}|\mathrm{x};\theta) + \lambda \mathcal{C}(\mathrm{n}) \tag{18}$$

Unfortunately, despite following all advice from Raffel et al. (2017), applying gradient clipping and different energy functions from Arivazhagan et al. (2019), we were not able to adapt MILk for use with the transformer model, largely due to the numerical instability of calculating α_i^n (see Raffel et al. (2017) for more details on this problem). We therefore turn to a different method which has so far not been applied to simultaneous machine translation, namely Adaptive Computation Time (ACT, (Graves, 2016)).

3.3 Adaptive Computation Time

Originally formulated for RNNs without the encoder-decoder framework, Adaptive Computation Time is a method that allows the RNN to "ponder" the same input for several timesteps, effectively creating sub-timesteps. We will first go over the original use-case, although we intentionally match the notation above. At each timestep i, we determine $N(i)$, the number of timesteps spent pondering the current input. We do so by predicting a probability at each sub-timestep s_i^n. We stop once the sum of these probabilities exceeds a threshold. We also calculate a *remainder* $R(i)$. Eqns. (19) through (22) are adapted from Graves (2016) and apply to RNNs:

$$p_i^n = \sigma(\text{ENERGY}(s_i^n)) \tag{19}$$

$$N(i) = \min\{n' : \sum_{n=1}^{n'} p_i^n \geq 1 - \epsilon\} \tag{20}$$

$$R(i) = 1 - \sum_{n=1}^{N(i)-1} p_i^n \tag{21}$$

$$\alpha_i^n = \begin{cases} R(i) & \text{if } n = N(i) \\ p_i^n & \text{otherwise} \end{cases} \tag{22}$$

It follows directly from the definition that α_i is a valid probability distribution. Compared to monotonic attention, ACT directly predicts the expected value for the amount of steps that the model takes, rather than calculating it from stopping probabilites. As-is, the model has no incentive to keep the ponder times short, so we introduce an additional loss:

$$\mathcal{C} = \sum_{i=1}^{|x|} N(i) + R(i) \tag{23}$$

Note that the computation for $N(i)$ is not differentiable so it is treated as a constant and the loss is equivalent to just summing the remainders.

We now go on to transfer ACT to the encoder-decoder domain. Now, instead of pondering the input to an RNN, like in original ACT, the decoder ponders over zero or more encoder steps. The encoder still works as in eq. (7) and does not use ACT. Instead, we apply the ACT ponder mechanism to the monotonic encoder-decoder attention. Let $N(i)$ denote the last encoder step to which we can attend. We make sure that $N(i)$ advances monotonically:

$$p_i^n = \sigma(\text{ENERGY}(y_{i-1}, h_n)) \tag{24}$$

$$N(i) = \min\{n' : \sum_{n=N(i-1)}^{n'} p_i^n \geq 1 - \epsilon\} \tag{25}$$

$$\alpha_i^n = \begin{cases} R(i) & \text{if } n = N(i) \\ p_i^n & \text{if } N(i-1) \leq n < N(i) \\ 0 & \text{otherwise} \end{cases} \tag{26}$$

Then we proceed as in equations (16) and (17). Note that in this formulation, it is possible that $N(i) = N(i-1)$ (i. e. the model pondering for zero steps), indicating consecutive WRITEs. In original ACT, it is not possible to ponder the input for zero steps. Also, similar to MILk, we consider $p_i^{|x|}$ to be 1 always. See figure 2 for a visualisation of α_i^n on a concrete example.

3.4 Transformer XL

Finally, we introduce two aspects of the Transformer XL language model (Dai et al., 2019) into our model: Relative attention and memory.

We replace the Transformer self-attention in both encoder and decoder with relative attention. In relative self-attention, we calculate ENERGY as follows:

$$\begin{aligned} \text{ENERGY}(x_i, x_j) = {} & x_i^\top W_q^\top W_E\, x_j \\ & + x_i^\top W_q^\top W_R\, R_{i-j} \\ & + u^\top W_E\, x_j \\ & + v^\top W_R\, R_{i-j} \end{aligned} \tag{27}$$

Where W_q, W_e, W_R, u, v are learnable parameters and R are relative position encodings. Afterwards, we proceed as in equation (16) and (17) for simultaneous models or eqautions (5) and (6) for offline models.

For our streaming model, we also use Transformer XL-style memory during training. This means that we keep the hidden states of both encoder and decoder from the previous training step during training. Both self-attention and encoder-decoder attention are able to attend to these states as well as the current input sentence. However, no gradients can flow through the old states to the model parameters.

3.5 Stream Translation

Our stream translation model should not rely on any segmentation information of the input and must be able to translate a test set as a single, continuous sequence. To achieve this, we extend the standard transformer model in the following ways:

- We use ACT monotonic attention to constrain the encoder-decoder attention. The position of the monotonic attention head also gives us a pointer to the model's current read position in the input stream that advances token by token, and not sentence by sentence and therefore requires no sentence segmentation.
- We change all self-attentions to relative attention, as well as removing absolute position encodings. We could encode positions as absolute since the beginning of the stream. However, Neishi and Yoshinaga (2019) showed that Transformer with absolute position encodings generalizes poorly to unseen sequence

lengths. In a continuous stream, relative encodings are the more logical choice.

- We add Transformer XL-style history to the model so that even the first positions of a sample have a history buffer for self-attention. This simulates the evaluation condition where we don't restart the model each sentence.
- During inference, we cannot cut off the history at sentence boundaries (such as keeping exactly the last sentence) because this information is not available. Instead, we adopt a rolling history buffer approach, keeping n_h previous positions for the self-attention. To simulate this condition in training, we apply a mask to the self-attention, masking out positions more than n_h positions in the past.
- During training, we concatenate multiple samples to a length of at least n_h tokens. This is to allow the model to READ past the end of an input sentence into the next one. Normally, this is prevented by setting $p_i^{|x|} = 1$. However during inference, $|x|$ is not available and therefore the model should learn to stop READing at appropriate times even across sentence boundaries.
- We use the ponder loss of equation (23) in addition to the cross-entropy translation loss with a weighting parameter λ as in equation (18).

4 Experiments

4.1 Segmented Translation

In our first set of experiments, we demonstrate the ability of ACT to produce state-of-the art results in sentence-based simultaneous machine translation. For comparison to Arivazhagan et al. (2019), we choose the same dataset: WMT2014 German-to-English (4.5M sentences). As they report their delay metrics on tokenized data, we also use the same tokenization and vocabulary.

All models follow the Transformer "base" configuration (Vaswani et al., 2017) and are implemented in fairseq (Ott et al., 2019). In addition to the simultaneous models, we train a baseline Transformer model. All models except the baseline use relative self-attention. We pre-train an offline model with future-masking in the encoder as a common basis for all simultaneous models.

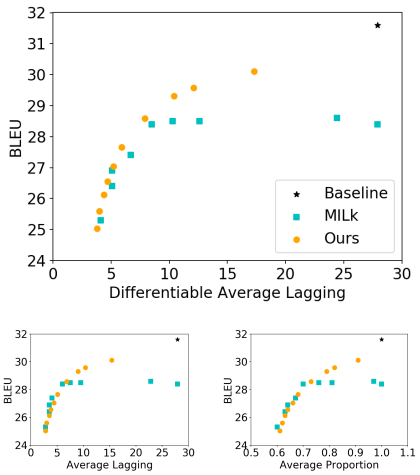


Figure 1: Quality-Latency comparison for German-to-English `newstest2015` in tokenized DAL (top), AL (bottom left) and AP (bottom right)

For the simultaneous models, we vary the value of λ and initialize the parameters from the pre-trained model. We found that training from the start with the latency loss can cause extreme latency behaviour, where the model either reads no input from the source at all or always waits until the end. We theorize that the best strategy would be to introduce the latency loss gradually during training, but leave that experiment for future work.

All models are trained using the Adam Optimizer (Kingma and Ba, 2015). For the pre-training model, we vary the learning rate using a cosine schedule from $2.5 \cdot 10^{-4}$ to 0 over 200k steps. For the ACT model, we start the learning rate at $4 \cdot 10^{-5}$ and use inverse square root decay (Vaswani et al., 2017) for 1000 steps.

We measure translation quality in detokenized, cased BLEU using `sacrebleu`[2] (Post, 2018). We measure latency in Average Lagging (Ma et al., 2019a), Differentiable Average Lagging (Arivazhagan et al., 2019) and Average Proportion (Cho and Esipova, 2016). For direct comparison, we report the tokenized latency metrics, but we provide the detokenized metrics in the appendix.

Figure 1 shows our results for this task. We generally achieve a better quality-latency tradeoff

[2] `BLEU+case.mixed+lang.de-en+numrefs.1 +smooth.exp+test.wmt15+tok.13a +version.1.4.3`

as measured by DAL, and a comparable one as measured by AP and AL. We note also that the ceiling for quality of ACT is higher than that of MILk. Whereas MILk loses two BLEU points to their baseline model even when given full attention ($\lambda = 0.0$), our model would seem to get closer to the performance of the baseline with decreasing λ.

4.2 Stream Translation

In this set of experiments, we demonstrate our model's ability to translate continuous streams of input with no sentence segmentation. For training, we use the IWSLT 2020 simultaneous translation data (which includes all WMT2019 data) with 37.6M sentences total. We choose this dataset because of a larger amount of document-level data (3.8M sentences). Because we will use Transformer XL-style memory, we depend on as much contextual data as possible. We evaluate on the IWSLT tst2010 test set in German to English. On the source side, we convert to lower case and remove all punctuation.

In addition to the baseline normal Transformer model, we train our model in three steps: First an offline, sentence-based relative self-attention Transformer, then the Transformer XL and finally the ACT+XL model, each one initializing its parameters on the last one. Both the relative model and the Transformer XL use the cosine schedule starting at $2.5 \cdot 10^{-4}$ and training for 200k and 40k steps, respectively. The ACT+XL model uses inverse square root decay, starting at $4 \cdot 10^{-5}$ as above and trains for 1000 steps. We also experiment with training ACT+XL directly from the relative model.

We evaluate as before[3], treating the test set as a single sequence. BLEU scores are calculated by re-segmenting the output according to the original reference based on Word Error Rate (Matusov et al., 2005). All reported metrics are detokenized. The baseline and relative models use beam search, the others use greedy decoding.

Unfortunately, the range of the λ parameter that produces sensible results is much more restricted than for the sentence-based model (see "Analysis", below). We report results with $\lambda = 0.25$ and 0.3.

Table 1 shows our results. There is a drop of 4 BLEU points when moving to simultaneous translation, which is similar to our experiments on segmented text. While there is room for improvement, these are promising results, and, to the best of our knowledge, the first demonstration of unsegmented end-to-end stream translation.

Model	AP	AL	DAL	BLEU
Baseline	—	—	—	32.0
Relative	—	—	—	33.1
XL	—	—	—	34.4
ACT+XL				
$\lambda = 0.25$	0.5	206	329	30.2
$\lambda = 0.3$	0.5	107	180	30.3
ACT+XL	*directly from relative*			
$\lambda = 0.25$	0.5	222	394	26.4

Table 1: Results for the stream translation experiment

4.3 Analysis

For the segmented translation, we compare two different latency schedules in figure 2. Both schedules advance relatively homogenously. This may indicate that the ACT attention layer needs to be expanded to extract more grammatical information and make more informed decisions on waiting. Nevertheless, the model produces good results and we even observe implicit verb prediction as in Ma et al. (2019a). We also note that the high latency models' latency graph tends to describe a curve, whereas the low latency models tend to uniformly advance by one token per output token.

This behaviour can be explained by the properties of Differentiable Average Lagging. The ponder loss objective that ACT is trained on may seem very different, but actually produces somewhat similar gradients to DAL [4], so the model incidentally also learns a behaviour that optimizes DAL.

DAL is monotonically increasing, i. e. the model can never "catch up" any delay by WRITing multiple tokens without READing (assuming $|y| = |x|$). It achieves the same DAL but with better translation by always READing one token when it WRITEs. Therefore, to achieve DAL $= k$ for a given k, the ideal waiting strategy is wait-k.

In the case of stream translation, we make two important observations: First, that systems with $\lambda < 0.25$ do not produce acceptable results (BLEU scores < 10). This is because they fall behind the input by waiting too much and have to skip sentences to catch back up. Once an input word is more than n_h tokens behind, it is removed from

[3] `BLEU+case.mixed+lang.de-en+numrefs.1 +smooth.exp+iwslt17/tst2010+tok.13a +version.1.4.3`

[4] $\frac{\partial \text{DAL}}{\partial \alpha_i^n} = i - N(i) - 1$, $\frac{\partial \text{ACT}}{\partial \alpha_i^n} = -1$ for $N(i-1) \leq i \leq N(i)$, else 0

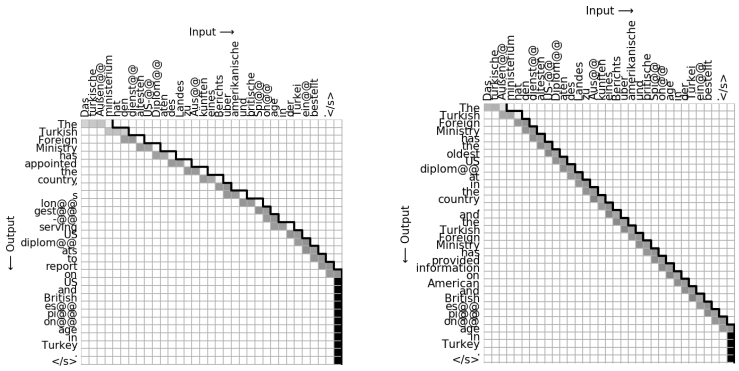


Figure 2: The same sentence from `newstest2015` translated by an ACT system with $\lambda = 0.1$ (left) and $\lambda = 0.4$ (right). The shading indicates the α_i^n as predicted by the ACT attention module (darker = higher probability), the black line indicates the hard attention cutoff. The low-latency model approaches the behaviour of a wait-4 model. Note the (incorrect) attempt of the left model to predict the verb "einbestellt" = "summons", whereas the right model takes the first half of the sentence as complete, leaving out the verb.

the memory and if it is not translated by then, it may be forgotten. Therefore, we found it essential to train more aggressive latency regimes. On the other hand, systems with $\lambda > 0.3$ sometimes read too little source information or stop reading new source words altogether.

Second, that the established latency metrics do not perform well on the very long sequence (with our tokenization, the source is 29 317 tokens long). While on single sentences, an AL score of 4 might indicate quite consistently a lag of around 4 tokens, a manual analysis of the output of our $\lambda = 0.3$ system shows a delay of between 40 and 60 words, quite far away from the automatic metrics of AL=107 and DAL=180. Average proportion in particular breaks down under these conditions and always reports 0.5.[5]

5 Conclusion and Future work

We have presented Adaptive Compuation Time (ACT) for simultaneous machine translation and demonstrated its ability to translate continuous, unsegmented streams of input text. To the best of our knowledge, this is the first end-to-end NMT model to do so. While stream translation model still loses a lot of performance compared to the sentence-based models, we see this as an important step towards end-to-end simultaneous stream translation.

We see several possibilites for future work on this model: Training the whole model in one training rather than the multiple rounds of pre-training may be possible by gradually introducing the latency loss during training. Perhaps the latency decisions can be improved by adding extra layers to the ACT attention module.

But most importantly, we believe the model must be adapted to the speech domain. Recently (see e. g. Di Gangi et al. (2019)), the Transformer has shown promising results for speech translation. For a realistic application we believe that a simultaneous translation model must work with speech input.

[5] The full output of the $\lambda = 0.3$ model can be found here: https://gist.github.com/felix-schneider/1462d855808e582aa19307f6b0d576e1

Acknowledgments

The work leading to these results has received funding from the European Union under grant agreement Nº 825460.

References

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of*

the 57th Annual Meeting of the Association for Computational Linguistics, pages 1313–1323.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Chung-Cheng Chiu and Colin Raffel. 2017. Monotonic chunkwise attention. *arXiv preprint arXiv:1712.05382*.

Eunah Cho, Jan Niehues, and Alex Waibel. 2012. Segmentation and punctuation prediction in speech language translation using a monolingual translation system. In *International Workshop on Spoken Language Translation (IWSLT) 2012*.

Eunah Cho, Jan Niehues, and Alex Waibel. 2017. Nmt-based segmentation and punctuation insertion for real-time spoken language translation. In *INTERSPEECH*, pages 2645–2649.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Mattia A Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting transformer to end-to-end spoken language translation. In *INTERSPEECH 2019*, pages 1133–1137. International Speech Communication Association (ISCA).

Christian Fügen, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine translation*, 21(4):209–252.

Alex Graves. 2016. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on empirical methods in natural language processing (EMNLP)*, pages 1342–1352.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Muntsin Kolss, Stephan Vogel, and Alex Waibel. 2008. Stream decoding for simultaneous spoken language translation. In *Ninth Annual Conference of the International Speech Communication Association*.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, et al. 2019a. 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.

Xutai Ma, Juan Pino, James Cross, Liezl Puzon, and Jiatao Gu. 2019b. Monotonic multihead attention. *arXiv preprint arXiv:1909.12406*.

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Hermann Ney. 2005. Evaluating machine translation output with automatic sentence segmentation. In *International Workshop on Spoken Language Translation (IWSLT) 2005*.

Masato Neishi and Naoki Yoshinaga. 2019. On the relation between position information and sentence length in neural machine translation. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 328–338.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic transcription for low-latency speech translation. In *Interspeech*, pages 2513–2517.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-latency neural speech translation. In *Proc. Interspeech 2018*, pages 1293–1297.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 551–556.

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 NAACL-HLT 2019: Demonstrations*.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Colin Raffel, Minh-Thang Luong, Peter J Liu, Ron J Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2837–2846. JMLR. org.

Harsh Satija and Joelle Pineau. 2016. Simultaneous machine translation using deep reinforcement learning. In *ICML 2016 Workshop on Abstraction in Reinforcement Learning*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Mahsa Yarmohammadi, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1032–1036.

A Segmented Translation Results

λ	Tokenized		
	AP	**AL**	**DAL**
Baseline	1.0	27.9	27.9
0.0	0.91	15.4	17.3
0.01	0.82	10.4	12.1
0.05	0.79	9.0	10.4
0.1	0.73	6.8	7.9
0.15	0.68	5.1	5.9
0.2	0.66	4.4	5.2
0.25	0.64	3.8	4.7
0.3	0.63	3.5	4.4
0.4	0.62	3.0	4.0
0.5	0.61	2.8	3.8

Table 2: Tokenized metrics for `newstest2015` backing figure 1

λ	Detokenized			
	AP	**AL**	**DAL**	**BLEU**
Baseline	1.0	18.6	18.6	31.6
0.0	0.93	10.4	11.7	30.1
0.01	0.84	7.2	8.5	29.6
0.05	0.81	6.3	7.5	29.3
0.1	0.76	4.9	6.0	28.6
0.15	0.71	3.8	4.8	27.7
0.2	0.69	3.4	4.4	27.0
0.25	0.68	3.0	4.1	26.6
0.3	0.67	2.9	3.9	26.1
0.4	0.66	2.6	3.7	25.6
0.5	0.65	2.4	3.6	25.0

Table 3: Detokenized metrics for `newstest2015`

E Edinburgh's End-to-End Multilingual Speech Translation System for IWSLT 2021

Edinburgh's End-to-End Multilingual Speech Translation System for IWSLT 2021

Biao Zhang[1] **Rico Sennrich**[2,1]

[1]School of Informatics, University of Edinburgh
[2]Department of Computational Linguistics, University of Zurich
B.Zhang@ed.ac.uk, sennrich@cl.uzh.ch

Abstract

This paper describes Edinburgh's submissions to the IWSLT2021 multilingual speech translation (ST) task. We aim at improving multilingual translation and zero-shot performance in the constrained setting (without using any extra training data) through methods that encourage transfer learning and larger capacity modeling with advanced neural components. We build our end-to-end multilingual ST model based on Transformer, integrating techniques including adaptive speech feature selection, language-specific modeling, multi-task learning, deep and big Transformer, sparsified linear attention and root mean square layer normalization. We adopt data augmentation using machine translation models for ST which converts the zero-shot problem into a zero-resource one. Experimental results show that these methods deliver substantial improvements, surpassing the official baseline by > 15 average BLEU and outperforming our cascading system by > 2 average BLEU. Our final submission achieves competitive performance (runner up).[1]

1 Introduction

Although end-to-end (E2E) speech translation (ST) has achieved great success in recent years, outperforming its cascading counterpart and delivering state-of-the-art performance on several benchmarks (Ansari et al., 2020; Zhang et al., 2020a; Zhao et al., 2020), it still suffers from the relatively low amounts of dedicated speech-to-translation parallel training data (Salesky et al., 2021). In text-based machine translation (MT), one solution to lack of training data is to jointly perform multilingual translation with the benefit of transferring knowledge across similar languages and to low-resource directions, and even enabling zero-shot translation, i.e. direct translation between language pairs unseen in training (Firat et al., 2016; Johnson et al., 2017). However, whether and how to obtain similar success in very low-resource (and practical) scenario for multilingual ST with E2E models remains an open question.

To address this question, we participated in the IWSLT2021 multilingual speech translation task, which focuses on low-resource ST language pairs in a multilingual setup. Apart from *supervised* evaluation, the task also offers *zero-shot* condition with a particular emphasis where only automatic speech recognition (ASR) training data is provided for some languages (without any direct ST parallel data). The task is organized in two settings: *constrained* setting and *unconstrained* setting. The former restricts participants to use the given multilingual TEDx data (Salesky et al., 2021) alone for experiment; while the latter allows for additional ASR/ST/MT/others training data. In this paper, we address the constrained one.

Our E2E multilingual ST model takes Transformer (Vaswani et al., 2017) as the backbone, and follows the adaptive feature selection (AFS) framework (Zhang et al., 2020a,b) as shown in Figure 1. AFS is capable of filtering out uninformative speech features contributing little to ASR, effectively reducing speech redundancy and improving ST performance (Zhang et al., 2020a). We adapt AFS to multilingual ST, and further incorporate several techniques that encourage transfer learning and larger capacity modeling, ranging from language-specific modeling, multi-task learning, deep and big Transformer, sparsified linear attention (ReLA) (Zhang et al., 2021b) to root mean square layer normalization (RMSNorm) (Zhang and Sennrich, 2019b). Inspired by Zhang et al. (2020c), we convert the zero-shot translation problem into a zero-resource one via data augmentation with multilingual MT models.

[1]Source code and pretrained models are available at https://github.com/bzhangGo/zero.

Proceedings of the 18th International Conference on Spoken Language Translation, pages 160–168
Bangkok, Thailand (Online), August 5–6, 2021.

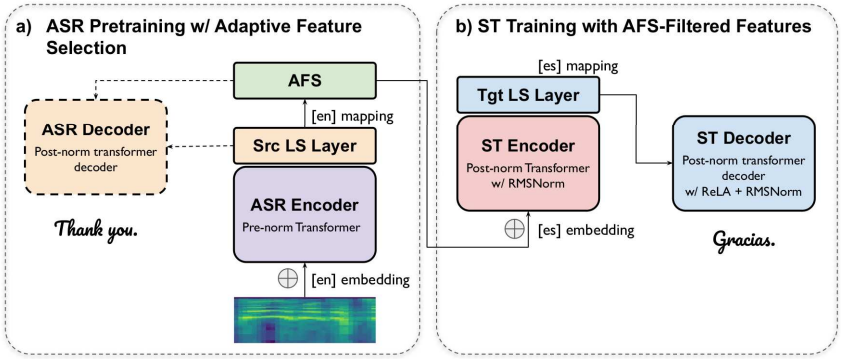


Figure 1: Overview of our multilingual ST model for an English-Spanish example. We first pretrain the ASR encoder paired with adaptive feature selection (AFS) to induce informative speech features (a), which are then carried over to the ST encoder-decoder model for translation (b). We adopt language embedding and language-specific (LS) linear mapping before and after ASR/ST encoder, respectively, to strengthen source/target (Src/Tgt) language modeling. The ASR decoder is discarded and the other ASR modules are frozen after the pretraining. Solid arrows illustrate the E2E translation procedure.

We integrate all these methods into one model for our submission. Our results reveal that:

- These methods are complementary in improving translation performance, where data augmentation and larger-capacity modeling contribute a lot.
- Low-resource E2E ST benefits greatly from multilingual modeling; our E2E multilingual ST performs very well in this task, outperforming its cascading counterpart by 2 average BLEU.

2 Methods

In this section, we elaborate crucial ingredients in our E2E multilingual ST, which individually have already been proven successful for ST or (multilingual) MT. We put them together to improve multilingual ST as shown in Figure 1. Note all encoder/decoder modules are based on Transformer (Vaswani et al., 2017).

2.1 Adaptive Feature Selection

Speech is lengthy and noisy compared to its text transcription. Also, information in an audio often distributes unevenly. All these increase the difficulty of extracting informative speech features. To solve this issue, researchers resort to methods compressing and grouping speech features (Salesky et al., 2019; Gaido et al., 2021). Particularly, Zhang et al. (2020a) propose adaptive feature selection (AFS) to sparsify speech encodings by pruning out those uninformative ones contributing little to ASR based on L_0DROP (Zhang et al., 2020b). Using AFS, Zhang et al. (2020a) observe significant performance improvements ($>$ 1 BLEU) with the removal of $\sim$84% speech features on bilingual ST.

Our model follows the AFS framework, which includes three steps: 1) pretraining the ASR encoder-decoder model; then 2) finetuning the ASR model with AFS; and 3) training ST model with the ASR encoder and the AFS module frozen.

2.2 Deep Transformer Modeling

Neural models often benefit from increased modeling capacity, and one way to achieve this is to deepen the models (He et al., 2015; Zhang et al., 2020d). However, simply increasing model depth for Transformer results in optimization failure, caused by gradient vanishing (Zhang et al., 2019a). To enable deep Transformer, Zhang et al. (2019a) propose depth-scaled initialization (DS-Init) that only requires changing parameter initialization without any architectural modification. DS-Init successfully helps to train up to 30-layer Transformer, substantially improving bilingual and also massively multilingual translation (Zhang et al., 2019a, 2020c). We adopt this strategy for all deep Transformer experiments.

Apart from DS-Init, researchers also find that changing the post-norm structure to its pre-norm alternative improves Transformer's robustness to deep modeling, albeit slightly reducing quality (Wang et al., 2019; Zhang et al., 2019a). We

keep using post-norm Transformer for most modules but apply the pre-norm structure to the ASR encoder to stabilize the encoding of speeches from different languages.

2.3 Language-Specific Modeling

Analogous to multi-task learning, multilingual translation benefits from inter-task transfer learning but suffers from task interference. How to balance between shared modeling and language-specific (LS) modeling so as to maximize the transfer effect and avoid the interference remains challenging. A recent study suggests that scheduling language-specific modeling to top and/or bottom encoder/decoder sub-layers benefits translation the most (Zhang et al., 2021a), resonating with the findings of Zhang et al. (2020c). In particular, Zhang et al. (2020c) propose language-aware linear transformation, a language-specific linear mapping inserted in-between the encoder and the decoder which greatly improves massively multilingual translation.

We adopt such language-specific linear mapping and apply it to both ASR and ST encoders. We ground such modeling in the ASR and ST encoder to the source and target language, respectively. Following multilingual translation (Johnson et al., 2017; Gangi et al., 2019; Inaguma et al., 2019), we adopt language embedding (such as "*[en]*, *[es]*") but add it to the inputs rather than appending an extra token.

2.4 Sparsified Linear Attention

Attention, as the key component in Transformer, takes the main responsibility to capture token-wise dependencies. However, not all tokens are semantically correlated, inspiring follow-up studies on sparsified attention that could explicitly zero-out some attention probabilities (Peters et al., 2019; Zhang et al., 2021b). Recently, Zhang et al. (2021b) propose rectified linear attention (ReLA) which directly induces sparse structures by enforcing ReLU activation on the attention logits. ReLA has achieved comparable performance on several MT tasks with the advantage of high computational efficiency against the sparsified softmax models (Peters et al., 2019).

Results on MT show that ReLA delivers better performance when applied to Transformer decoder (Zhang et al., 2021b). We follow this practice and apply it to the ST decoder. Our study also demonstrates that ReLA generalizes well to ST.

2.5 Root Mean Square Layer Normalization

Layer normalization (LayerNorm) stabilizes network activations and improves model performance (Ba et al., 2016), but raises non-negligible computational overheads reducing net efficiency, particularly to recurrent models (Zhang and Sennrich, 2019a). To overcome such overhead, Zhang and Sennrich (2019b) propose root mean square layer normalization (RMSNorm) which relies on root mean square statistic alone to regularize activations and is a drop-in replacement to LayerNorm. RMSNorm yields comparable performance to LayerNorm in a series of experiments (Zhang and Sennrich, 2019b) and show great scalability in large-scale pretraining (Narang et al., 2021).

We apply RMSNorm to the ST encoder and decoder, which benefits the training of deep and big Transformers.

2.6 Data Augmentation

Data augmentation (DA) is an effective strategy for low-resource tasks by increasing the training corpus with pseudo-labelled samples (Sennrich et al., 2016a; Zhang and Zong, 2016). Methods for generating such samples vary greatly, and we adopt the one following knowledge distillation (Kim and Rush, 2016). Note, prior to our study, knowledge distillation has already been successfully applied to ST tasks (Liu et al., 2019; Gaido et al., 2020). We regard the multilingual MT as the teacher since text-based translation is much easier than and almost upper-bounds the speech-based counterpart (Zhang et al., 2020a), and transfer its knowledge into our multilingual ST (student).

Concretely, we first train a multilingual MT model and then use it to translate each source transcript into all possible ST directions, including the zero-shot ones, based on beam search algorithm. We directly concatenate the generated pseudo speech-translation pairs with the original training corpus for multilingual ST training. This will convert the zero-shot translation problem into a zero-resource one for ST, which has been demonstrated effective in massively multilingual MT (Zhang et al., 2020c).

2.7 Multi-Task Learning

Multi-task learning aims at improving task performance by jointly modeling different tasks within one framework. Particularly, when tasks are of high correlation, they tend to benefit each other and de-

Speech	Target Languages				
	En	Es	Fr	Pt	It
Es	36K/102K	102K/-	3.6K/102K	21K/102K	5.6K/102K
Fr	30K/116K	21K/116K	116K/-	13K/116K	-/116K
Pt	31K/90K	-/90K	-/90K	90K/-	-/90K
It	-/50K	-/50K	-/50K	-/50K	50K/-

Table 1: Statistics for ST training data used for the IWSLT2021 multilingual ST task. "-": denotes no data available. "*a*/*b*": "*a*" denotes genuine data while "*b*" is for augmented data.

liver positive knowledge transfer. With datasets of different tasks combined, this also partially alleviates data scarcity.

We adopt multi-task learning by augmenting translation tasks with transcription tasks. We incorporate the ASR tasks for multilingual ST, and auto-encoding tasks (transcript-to-transcript in the same language) for multilingual MT.

3 Experimental Settings

In this section, we explain the used datasets, model architectures, optimization details and evaluation metrics in our experiments. All implementations are based on the *zero*[2] toolkit (Zhang et al., 2018).

Data We participate in the constrained setting, where only the provided data, i.e. Multilingual TEDx (Salesky et al., 2021), is permitted. Multilingual TEDx collects audios from TEDx talks in 8 source languages (Spanish/Es, French/Fr, Portuguese/Pt, Italian/It, Russian/Ru, Greek/El, Arabic/Ar, German/De) paired with their manual transcriptions, covering translations into 5 target languages (English/En, Es, Fr, Pt, It). It contains supervised training data for 13 ST directions, three of which (Pt-Es, It-En, It-Es) are masked-out for zero-shot evaluation. ASR training data is given for all 8 source languages. Overall, Multilingual TEDx is a small-scale dataset, whose ST training data size ranges from 5K utterances (It-Es) to at most 39K utterances (Es-En). Thus, studying and improving transfer across different languages is of great significance. The IWSLT2021 task requires participants to model translations from 4 source languages (Es, Fr, Pt, It), where the final evaluation only targets translations into En and Es. The statistics of ST (genuine and augmented) training data are shown in Table 1.

Regarding audio preprocessing, we use the given audio segmentation (train/dev/test) for experiments. We extract 40-dimensional log-Mel filterbanks with a step size of 10ms and window size of 25ms as the acoustic features, followed by feature expansion via second-order derivatives and mean-variance normalization. The final acoustic input is 360-dimensional, a concatenation of the features corresponding to three consecutive and non-overlapping frames. We tokenize and truecase all text data using Moses scripts (Koehn et al., 2007). We adopt subword processing (Sennrich et al., 2016b) with 8K merging operations (Sennrich and Zhang, 2019) on these texts to handle rare words. Note we use different subword models (but with the same vocabulary size) for ST, ASR and MT.

Architecture The architecture for ASR and ST is illustrated in Figure 1, while our MT model follows Zhang et al. (2020c). We apply AFS to ASR encoder outputs (after language-specific mapping) along both temporal and feature dimensions. By default, we adopt Transformer-base setting (Vaswani et al., 2017): we use 6 encoder/decoder layers and 8 attention heads with a model dimension of 512/2048. For deep Transformer, we equally increase the encoder and decoder depth, and adopt DS-Init for training. We also use Transformer-big for ST, where the number of attention heads and model dimension are doubled, increased to 16 and 1024/4096, respectively.

Optimization We train MT models with the maximum likelihood objective ($\mathcal{L}_{\text{MLE}}$). Apart from $\mathcal{L}_{\text{MLE}}$, we also incorporate the CTC loss (Graves et al., 2006) for ASR pretraining with a weight value of 0.3 following Zhang et al. (2020a). During AFS finetuning, the CTC loss is discarded and replaced with the L_0DROP sparsification loss (Zhang et al., 2020b) weighted by 0.5. We employ label smoothing of value 0.1 for $\mathcal{L}_{\text{MLE}}$.

We adopt Adam (β_1=0.9, β_2=0.98) for parameter tuning with a warmup step of 4K. We train all models (ASR, ST and MT) for 100K steps, and finetune AFS for 10K steps. We group instances of around 25K target subwords into one mini-batch. We apply dropout to attention weights and residual connections with a rate of 0.1 and 0.2, respectively. Dropout rate on residual connections is increased to 0.3 for ST big models to avoid overfitting, and to 0.5 for MT models inspired by low-resource MT (Sennrich and Zhang, 2019). Except dropout, we use *no* other regularization techniques. We use beam search for decoding, and set the beam size and length penalty to 4 and 0.6, separately. The

[2] https://github.com/bzhangGo/zero

Model	Es-En	Es-Pt	Es-Fr	Fr-En	Fr-Es	Fr-Pt	Pt-En	Pt-Es	It-En	It-Es	Avg
Bilingual Models⋆	25.5	39.3	2.0	28.3	30.5	19.0	27.9	29.9	18.9	1.0	22.23
Multilingual Models⋆	24.6	37.3	18.1	28.2	32.1	30.6	28.8	38.4	20.9	25.1	28.41
Our Multilingual MT											
+ 6 layers	28.7	42.1	29.3	33.6	38.3	36.7	33.2	42.9	20.3	32.7	33.78
+ 12 layers	31.8	44.7	31.7	36.4	40.9	39.9	35.6	44.0	23.0	34.9	36.29
+ 24 layers	32.8	44.9	32.4	37.3	41.8	40.7	36.8	43.2	23.2	35.3	**36.84**
Ablation Study											
+ 6 layers w/o LS layer	28.6	41.8	29.0	33.7	38.2	36.3	33.2	42.5	20.7	32.6	33.66
+ 6 layer + RoBT	28.1	40.3	28.6	34.1	38.3	33.6	33.6	42.7	21.1	32.9	33.33

Table 2: SacreBLEU↑ for MT on Multilingual TEDx testsets. ⋆: results reported by Salesky et al. (2021). Note the results for Pt-Es, It-En and It-Es translation in our model are based on zero-shot evaluation. In spite of this unfairness, our model still substantially outperforms the supervised baseline (Salesky et al., 2021) by a large margin, +8.43 BLEU. *RoBT*: random online back-translation (Zhang et al., 2020c). Best average BLEU is highlighted in **bold**. Columns in red denote zero-shot evaluation.

Model	Es	Fr	Pt	It	Ru	El	Ar	De	Avg
Hybrid LF-MMI⋆	16.2	19.4	20.2	16.4	28.4	25.0	80.8	42.3	**31.09**
Transformer⋆	46.4	45.6	54.8	48.0	74.7	109.5	104.4	111.1	74.31
Our Multilingual ASR									
+ 6 layers	17.6	19.5	23.1	20.8	39.8	33.0	104.3	57.8	**39.49**
Ablation Study									
+ 6 layers w/o LS layer	18.0	19.5	23.2	21.6	40.8	35.2	97.8	62.6	39.84

Table 3: WER↓ for ASR on Multilingual TEDx testsets. ⋆: results reported by Salesky et al. (2021). Best results are highlighted in **bold**.

model used for evaluation is averaged over the last 5 checkpoints.

Note, while the training data size varies across languages, we follow the original data distribution and adopt *no* specific sampling strategies for all multilingual experiments.

Evaluation We evaluate translation quality using tokenized case-sensitive (Sacre)BLEU (Papineni et al., 2002; Post, 2018), and report WER for ASR performance without punctuation on lowercased text. In ST experiments, we observe some repeated translations decreasing BLEU. We automatically post-process translations by removing repeated chunks of up to 10 words.

4 Results

4.1 Multilingual MT

Table 2 shows the results for text-based translation. Our best model, achieved with 24 layers, largely surpasses the official baseline (Salesky et al., 2021) by $>$ 8 average BLEU. With 6 layers, our model still largely surpasses this baseline by 5.37 average BLEU, suggesting the superiority of our model.

Increasing model depth greatly benefits multilingual MT (+2.51 average BLEU, 6 layers $\rightarrow$ 12 layers), even though the dataset is small. Note the benefit from increased depth diminishes as the depth goes larger (+0.55 average BLEU, 12 layers $\rightarrow$ 24 layers). We find that language-specific modeling slightly improves translation performance (+0.12 average BLEU). Such improvement seems uninteresting particularly compared to the significant gains on massively multilingual MT (Zhang et al., 2020c), but we ascribe this to the high language similarity in Multilingual TEDx and the relative small number of languages. We also confirm the effectiveness of random online back-translation (RoBT), which improves zero-shot translation via pseudo sentence pair augmentation (Zhang et al., 2020c). Table 2 shows that RoBT indeed benefits zero-shot translation, but sacrifices overall quality (-0.45 average BLEU).

Overall, our results reveal very positive transfer between these languages, and also great zero-shot translation performance. This is an encouraging finding for multilingual ST. We use our 24-layer model for data augmentation distillation in the following ST experiments.

Model	Es-En	Es-Pt	Es-Fr	Fr-En	Fr-Es	Fr-Pt	Pt-En	Pt-Es	It-En	It-Es	Avg
Multilingual Models*	12.3	17.4	6.1	12.0	13.6	13.2	12.0	13.7	10.7	13.1	12.41
Cascades with Multilingual MT*	21.5	26.5	23.4	25.3	26.9	23.3	22.3	26.3	21.9	28.4	24.58
Our Multilingual MT, w/ AFS, LS layer, DA, ReLA (decoder self-attention) and RMSNorm											
+ 6 layers	24.9	34.8	26.6	30.0	33.8	33.2	27.4	33.9	20.7	30.8	29.61
+ 12 layers	24.6	35.6	26.7	29.9	33.7	33.5	28.5	34.4	21.1	30.6	29.86
+ 6 layers + big model	26.1	36.2	27.5	31.0	34.9	34.3	28.7	35.1	21.6	31.5	**30.69**
Ablation Study											
+ 6 layers w/o AFS	25.2	35.1	26.4	29.9	33.2	32.7	28.4	33.7	20.3	29.6	29.45
+ 6 layers w/o AFS & DA	20.8	30.9	18.5	24.7	27.6	27.0	23.8	27.2	13.8	20.0	23.43
+ 6 layers w/o ReLA & RMSNorm	24.2	34.8	26.4	29.5	34.1	33.4	27.5	33.7	20.7	30.3	29.46
+ 6 layers + ReLA on cross-att.	24.8	35.3	27.1	30.2	34.3	33.8	27.6	34.1	20.5	30.5	**29.82**
Our Cascade Model w/ Multilingual ASR + 24-layer Multilingual MT											
	24.8	33.7	25.3	29.2	32.7	32.2	26.9	31.7	18.5	27.1	28.21
Final Submission: Ensemble of 4 base model, 1 12-layer model and 1 big model w/ length penalty of 0.9											
	26.6	36.6	27.9	31.8	35.6	35.4	29.7	35.8	22.0	32.0	**31.34**

Table 4: SacreBLEU↑ for ST on Multilingual TEDx testsets. *: results reported by Salesky et al. (2021). Note the results for Pt-Es, It-En and It-Es translation in our model are based on zero-shot evaluation. Our model substantially outperforms the official baseline (Salesky et al., 2021) by $>$ 10 average BLEU. *DA*: data augmentation. Best average BLEU is highlighted in **bold**.

4.2 Multilingual ASR

Table 3 shows the ASR performance. Following previous studies (Salesky et al., 2021; Zhang et al., 2020a), we experiment with the Transformer base setting. Our multilingual ASR model yields an average WER of 39.49, substantially outperforming the official baseline (Salesky et al., 2021) by 34.82 and narrowing the performance gap against the hybrid model to $\sim$ 8 WER. Note lower WER indicates better quality. We ascribe this large quality gain to the dedicated multilingual ASR model architecture, the better optimization, and particularly the incorporation of the CTC objective.

Removing the language-specific layer slightly hurts recognition performance (+0.35 average WER). It largely benefits ASR for Ar (-6.5 WER), but hurts that for De (+4.8 WER), showing the difficulty of multilingual modeling: it's hard to balance between different tasks (translation directions). We adopt the model with language-specific projection for AFS and ST.

Notice that we still include Ru, El, Ar and De for the ASR training, although they are not a part of the evaluation campaign. We regard this inclusion as some sort of model regularization: the extra training data could reduce overfitting and might enable potential cross-lingual transfer.

4.3 Multilingual ST

Table 4 summarizes the ST results. Our base model using 6 layers delivers an average BLEU of 29.61, largely outperforming the official baseline (Salesky et al., 2021) by $\sim$ 17 BLEU and also beating their cascading baseline. In a fair comparison where knowledge data augmentation is not used, our model still obtain an average BLEU of 23.43.

Increasing the ST model depth slightly improves quality (+0.25 average BLEU), while enlarging ST model yields a larger improvement, reaching 1.08. Although it's widely known that large neural model often suffers from overfitting in low-resource tasks, our results suggest that such model still gains quality with proper regularization (AFS, larger dropout, etc).

Our ablation study demonstrates the effectiveness of AFS, ReLA and RMSNorm, although the corresponding quality gains are marginal. In particular, we observe that applying ReLA to both self-attention and cross-attention in the ST decoder helps (Zhang et al., 2021b). AFS improves training efficiency, allowing larger batch size thus fewer gradient accumulation steps (Zhang et al., 2020a). Besides, data augmentation benefits multilingual ST very much, resulting in $\sim$ 6 average BLEU improvement, and the gain on zero-shot directions is even higher, + 7.54 BLEU. Thus, we mainly ascribe our success on zero-shot translation to the inclusion of pseudo parallel corpora – data matter! – which converts the zero-shot problem into a zero-resource one.

Our E2E model also largely outperforms the cascading system (+ 2.48 average BLEU). Notice that our cascading system is sub-optimal, since we

Model	Es-En	Es-Fr	Es-It	Es-Pt	Fr-En	Fr-Es	Fr-Pt	Pt-En	Pt-Es	It-En	It-Es	Avg
Ensemble of 6 E2E models: 4 base model, 1 12-layer model and 1 big model w/ length penalty of 0.9												
	36.2	30.3	32.9	44.5	26.4	29.5	30.1	27.0	34.5	23.0	31.1	**31.41**
Cascading model: base ASR model + 24-layer MT model												
	33.3	26.8	28.6	39.9	23.7	26.9	26.8	23.6	30.0	19.7	26.7	27.82
Single E2E Model: multilingual ST model + 6 layers, big Transformer												
	35.0	29.9	31.9	44.1	25.5	28.8	29.0	26.2	33.3	22.4	30.1	30.56

Table 5: SacreBLEU↑ for our submissions to the IWSLT2021 multilingual ST task.

didn't bias our MT model towards ASR outputs, and the mismatch between gold transcripts and ASR outputs often hurts cascading performance. Recent advances on avoiding such error propagation might deliver better cascading results (Cheng et al., 2018; Zhang et al., 2019b; Cheng et al., 2019; Sperber et al., 2019).

Our final submission is an ensemble of 6 E2E multilingual ST models, which reaches an average BLEU of 31.34. Apart from the ensemble, we also increase the decoding length penalty from 0.6 to 0.9, which performs slightly better.

5 Submission Results

The IWSLT2021 task prepares a held-out test set for the final evaluation. We submitted three systems: one cascading system, one E2E single model (w/ big ST Transformer) and one ensemble model. Results are shown in Table 5: our E2E multilingual ST model outperforms its cascading counterpart, and the ensemble model reaches the best performance. Our submission achieves runner-up results among all participants.

6 Conclusion and Future Work

We describe Edinburgh's end-to-end multilingual speech translation system for the IWSLT2021 multilingual speech translation task. We observe substantial performance improvement using larger-capacity modeling (deep or big modeling) and data augmentation. In spite of the scarcity of the training data, we show that E2E models benefit greatly from multilingual modeling and deliver promising results on zero-shot translation directions (even without data augmentation). Our E2E multilingual ST greatly surpasses its cascading counterpart.

Regarding future study, we argue that exploring the multilingual transfer behavior should be very practical and promising to ST. This work mainly studies transfer across similar languages. How the current model generalizes to distant languages is still an open question. Besides, a general trend for deep learning is to increase the model capacity via deep and/or big modeling. However, deep models for ST seem to be ineffective. Identifying the reason for this and proposing simple solutions would be of high interest.

Acknowledgements

We thank the reviewers for their insightful comments. This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreements 825460 (ELITR). Rico Sennrich acknowledges support of the Swiss National Science Foundation (MUTAMUR; no. 176727).

References

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondřej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, Alexander Waibel, and Changhan Wang. 2020. FINDINGS OF THE IWSLT 2020 EVALUATION CAMPAIGN. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 1–34, Online. Association for Computational Linguistics.

Jimmy Ba, J. Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *ArXiv*, abs/1607.06450.

Qiao Cheng, Meiyuan Fang, Yaqian Han, Jin Huang, and Yitao Duan. 2019. Breaking the data barrier: Towards robust speech translation via adversarial stability training. *CoRR*, abs/1909.11430.

Yong Cheng, Zhaopeng Tu, Fandong Meng, Junjie Zhai, and Yang Liu. 2018. Towards robust neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1766, Melbourne, Australia. Association for Computational Linguistics.

Orhan Firat, Baskaran Sankaran, Yaser Al-onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016. Zero-resource translation with multi-lingual neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 268–277, Austin, Texas. Association for Computational Linguistics.

Marco Gaido, Mauro Cettolo, Matteo Negri, and Marco Turchi. 2021. CTC-based compression for direct speech translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 690–696, Online. Association for Computational Linguistics.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2020. End-to-end speech-translation with knowledge distillation: FBK@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 80–88, Online. Association for Computational Linguistics.

Mattia Antonino Di Gangi, Matteo Negri, and Marco Turchi. 2019. One-to-many multilingual end-to-end speech translation. In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pages 585–592. IEEE.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA. Association for Computing Machinery.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition.

Hirofumi Inaguma, Kevin Duh, Tatsuya Kawahara, and Shinji Watanabe. 2019. Multilingual end-to-end speech translation. In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pages 570–577. IEEE.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

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.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Yuchen Liu, Hao Xiong, Jiajun Zhang, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019. End-to-End Speech Translation with Knowledge Distillation. In *Proc. Interspeech 2019*, pages 1128–1132.

Sharan Narang, Hyung Won Chung, Yi Tay, William Fedus, Thibault Fevry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam Shazeer, Zhenzhong Lan, Yanqi Zhou, Wei Li, Nan Ding, Jake Marcus, Adam Roberts, and Colin Raffel. 2021. Do transformer modifications transfer across implementations and applications?

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.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2019. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, Florence, Italy. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Elizabeth Salesky, Matthias Sperber, and Alan W Black. 2019. Exploring phoneme-level speech representations for end-to-end speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1841, Florence, Italy. Association for Computational Linguistics.

Elizabeth Salesky, Matthew Wiesner, Jacob Bremerman, Roldano Cattoni, Matteo Negri, Marco Turchi, Douglas W. Oard, and Matt Post. 2021. The multilingual tedx corpus for speech recognition and translation.

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.

Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy. Association for Computational Linguistics.

Matthias Sperber, Graham Neubig, Ngoc-Quan Pham, and Alex Waibel. 2019. Self-attentional models for lattice inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1185–1197, Florence, Italy. Association for Computational Linguistics.

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.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy. Association for Computational Linguistics.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021a. Share or not? learning to schedule language-specific capacity for multilingual translation. In *International Conference on Learning Representations*.

Biao Zhang and Rico Sennrich. 2019a. A lightweight recurrent network for sequence modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1538–1548, Florence, Italy. Association for Computational Linguistics.

Biao Zhang and Rico Sennrich. 2019b. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Biao Zhang, Ivan Titov, Barry Haddow, and Rico Sennrich. 2020a. Adaptive feature selection for end-to-end speech translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2533–2544, Online. Association for Computational Linguistics.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2019a. Improving deep transformer with depth-scaled initialization and merged attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 898–909, Hong Kong, China. Association for Computational Linguistics.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2020b. On sparsifying encoder outputs in sequence-to-sequence models.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2021b. Sparse attention with linear units.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020c. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2020d. Neural machine translation with deep attention. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(1):154–163.

Biao Zhang, Deyi Xiong, jinsong su, Qian Lin, and Huiji Zhang. 2018. Simplifying neural machine translation with addition-subtraction twin-gated recurrent networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4273–4283. Association for Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Austin, Texas. Association for Computational Linguistics.

Pei Zhang, Niyu Ge, Boxing Chen, and Kai Fan. 2019b. Lattice transformer for speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6475–6484, Florence, Italy. Association for Computational Linguistics.

Chengqi Zhao, Mingxuan Wang, and Lei Li. 2020. Neurst: Neural speech translation toolkit.

F Beyond Sentence-Level End-to-End Speech Translation: Context Helps

Beyond Sentence-Level End-to-End Speech Translation: Context Helps

Biao Zhang[1] **Ivan Titov**[1,2] **Barry Haddow**[1] **Rico Sennrich**[3,1]

[1]School of Informatics, University of Edinburgh
[2]ILLC, University of Amsterdam
[3]Department of Computational Linguistics, University of Zurich
B.Zhang@ed.ac.uk, {ititov,bhaddow}@inf.ed.ac.uk, sennrich@cl.uzh.ch

Abstract

Document-level contextual information has shown benefits to text-based machine translation, but whether and how context helps end-to-end (E2E) speech translation (ST) is still under-studied. We fill this gap through extensive experiments using a simple concatenation-based context-aware ST model, paired with adaptive feature selection on speech encodings for computational efficiency. We investigate several decoding approaches, and introduce in-model ensemble decoding which jointly performs document- and sentence-level translation using the same model. Our results on the MuST-C benchmark with Transformer demonstrate the effectiveness of context to E2E ST. Compared to sentence-level ST, context-aware ST obtains better translation quality (+0.18-2.61 BLEU), improves pronoun and homophone translation, shows better robustness to (artificial) audio segmentation errors, and reduces latency and flicker to deliver higher quality for simultaneous translation.[1]

1 Introduction

Document-level context often offers extra informative clues that could improve the understanding of individual sentences. Such clues have been proven effective for textual machine translation (MT), particularly in handling translation errors specific to discourse phenomena, such as inaccurate coreference of pronouns (Guillou, 2016) and mistranslation of ambiguous words (Rios et al., 2017). Besides, ensuring consistency in translation is virtually impossible without document-level context as well (Voita et al., 2019). Analogous to MT, speech translation (ST) also suffers from these translation issues, and super-sentential context could in fact be more valuable to ST because 1) homophones and acoustic noise bring additional ambiguity to ST, and 2) a common use case in ST is simultaneous translation, where the system has to output translations of sentence fragments, and may have to predict future input to account for word order differences between the source and target language (Grissom II et al., 2014). Both for ambiguity from the acoustic signal, and operating on small sentence fragments, we hypothesize that access to extra context[2] will be beneficial.

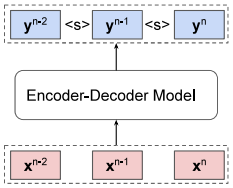


Figure 1: Overview of the concatenation-based context-aware ST. $\mathbf{y}^n$ denotes the n-th target sentence in a document; $\mathbf{x}^n$ denotes the speech encodings extracted from the n-th audio segment. We use dashed gray box to indicate the concatenation operation. "<s>": sentence separator symbol.

Although recent studies on ST have achieved promising results with end-to-end (E2E) models (Anastasopoulos and Chiang, 2018; Di Gangi et al., 2019; Zhang et al., 2020a; Wang et al., 2020; Dong et al., 2020), nevertheless, they mainly focus on sentence-level translation. One practical challenge when scaling up sentence-level E2E ST to the document-level is the encoding of very long audio segments, which can easily hit the computational bottleneck, especially with Transformers (Vaswani et al., 2017). So far, the research question of whether and how contextual information benefits E2E ST has received little attention.

In this paper, we answer this question through extensive experiments by exploring a concatenation-

[1]Source code is available at https://github.com/bzhangGo/zero.

[2]By default, we use *context* to denote both source- and target-side information from previous sentences.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 2566–2578
August 1–6, 2021.

based context-aware ST model. Figure 1 illustrates our model, where neighboring source (target) sequences are chained together into one sequence for joint translation. This paradigm only requires data-level manipulation, thus allowing us to reuse any existing sentence-level E2E ST models. Despite its simplicity, this approach successfully leverages contextual information to improve textual MT (Tiedemann and Scherrer, 2017; Bawden et al., 2018; Lopes et al., 2020), and here we adapt it to ST. As for the computational bottleneck, we shorten the speech encoding sequence via adaptive feature selection (Zhang et al., 2020b,a, AFS), which only retains a small subset of encodings (∼16%) for each audio segment.

We investigate several decoding methods, including chunk-based decoding and sliding-window based decoding. We also study an extension of the latter with the constraint of target prefix, where the prefix denotes the translation of previous context speeches. We find that using these methods sometimes results in *misaligned translations*, particularly when using the constraint. This issue manifests itself in mismatching sentence boundaries and producing over- and/or under-translation, which greatly hurts sentence-based evaluation metrics. To avoid such misalignments, we introduce *in-model ensemble decoding* (IMED) to regularize the document-level translation with its sentence-level counterpart. Note that we use the same context-aware ST model here for both types of translation – that's why we call it *in-model* ensemble.

We adopt Transformer (Vaswani et al., 2017) for experiments with the MuST-C dataset (Di Gangi et al., 2019). We study the impact of context on translation in different settings. Our results demonstrate the effectiveness of contextual modeling. Our main findings are summarized below:

- Incorporating context improves overall translation quality (+0.18-2.61 BLEU) and benefits pronoun translation across different language pairs, resonating with previous findings in textual MT (Miculicich et al., 2018; Huo et al., 2020). In addition, context also improves the translation of homophones.
- ST models with contexts suffer less from (artificial) audio segmentation errors.
- Contextual modeling improves translation quality and reduces latency and flicker for simultaneous translation under re-translation strategy (Arivazhagan et al., 2020a).

2 Related Work

Our work is inspired by pioneer studies on context-aware textual MT. Context beyond the current sentence carries information whose importance for translation cohesion and coherence has long been posited (Hardmeier et al., 2012; Xiong and Zhang, 2013). With the rapid development of neural MT and also available document-level textual datasets, research in this direction gained great popularity. Recent efforts often focus on either advanced contextual neural architecture development (Tiedemann and Scherrer, 2017; Kuang et al., 2018; Miculicich et al., 2018; Zhang et al., 2018, 2020c; Kang et al., 2020; Chen et al., 2020; Ma et al., 2020a; Zheng et al., 2020) and/or improved analysis and evaluation targeted at specific discourse phenomena (Bawden et al., 2018; Läubli et al., 2018; Guillou et al., 2018; Voita et al., 2019; Kim et al., 2019; Cai and Xiong, 2020). We follow this research line, and adapt the concatenation-based contextual model (Tiedemann and Scherrer, 2017; Bawden et al., 2018; Lopes et al., 2020) to ST. Our main interest lies in exploring the impact of context on ST. Developing dedicated contextual models for ST is beyond the scope of this study, which we leave to future work.

Context-aware ST extends the sentence-level ST towards streaming ST which allows models to access unlimited previous audio inputs. Instead of improving contextual modeling, many studies on streaming ST aim at developing better sentence/word segmentation policies to avoid segmentation errors that greatly hurt translation (Matusov et al., 2007; Rangarajan Sridhar et al., 2013; Iranzo-Sánchez et al., 2020; Zhang and Zhang, 2020; Arivazhagan et al., 2020b). Very recently, Ma et al. (2020b) proposed a memory augmented Transformer encoder for streaming ST, where the previous audio features are summarized into a growing continuous memory to improve the model's context awareness. Despite its success, this method ignores the target-side context, which turns out to have significant positive impact on ST in our experiments.

Our study still relies on *oracle* sentence segmentation of the audio. The most related work to ours is (Gaido et al., 2020), which also investigated contextualized translation and showed that context-aware ST is less sensitive to audio segmentation errors. While they exclusively focus on the robustness to segmentation errors, our study investigates the benefits of context-aware E2E ST more broadly.

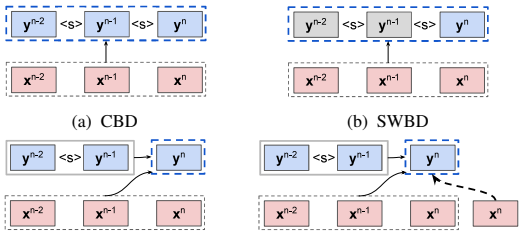


Figure 2: Illustration of different decoding methods: chunk-based decoding (CBD, 2a), sliding-window based decoding without (SWBD, 2b) and with (SWBD-Cons, 2c) the target prefix constraint and the proposed in-model ensemble decoding (IMED, 2d). The dashed blue box denotes model generation; the solid gray box (2c, 2d) indicates the target prefix constraint; sentences in the gray rectangle (2b) are discarded after generation. The dashed arrow in IMED stands for the sentence-level translation.

3 Context-aware ST via Concatenation

We extend the sentence-level ST with document-level context, by modeling up to C previous source/target segments/sentences for translation. Formally, given a pre-segmented audio (source document) $\mathbf{A} = \left(\mathbf{a}^1, \ldots, \mathbf{a}^N\right)$ as well as its paired target document $\mathbf{Y} = \left(\mathbf{y}^1, \ldots, \mathbf{y}^N\right)$, the model is trained to maximize the following likelihood:

$$\log p\left(\mathbf{Y}|\mathbf{A}\right) = \sum_{n=1}^{N} \log p\left(\mathbf{y}^n|\mathbf{x}^n, \mathcal{C}_{\mathbf{y}}^n, \mathcal{C}_{\mathbf{x}}^n\right), \quad (1)$$

where $\mathbf{x}^n = \text{AFS}\left(\mathbf{a}^n\right)$, i.e. the speech encodings extracted via AFS (Zhang et al., 2020a). $\mathbf{a}^n$ and $\mathbf{y}^n$ denote the n-th audio segment and target sentence, respectively. N is the number of segments/sentences in the document. $\mathcal{C}_{\mathbf{x}}^n$ and $\mathcal{C}_{\mathbf{y}}^n$ stand for the source and target context, respectively, i.e. $\{\mathbf{x}^{n-i}\}_{i=1}^{C}$ and $\{\mathbf{y}^{n-i}\}_{i=1}^{C}$.

Adaptive Feature Selection Audio segment is often converted into frame-based features for neural modeling. Different from text, each segment might contain hundreds or even thousands of such features, making contextual modeling computationally difficult. Zhang et al. (2020a) found that most speech encodings emitted by a Transformer-based audio encoder carry little information for translation, and their deletion even improves translation quality. We follow Zhang et al. (2020a) and perform AFS to only extract those informative encodings (∼16%) optimized via sentence-level speech recognition with $\mathcal{L}_0$DROP (Zhang et al., 2020b). This greatly shortens the speech encoding sequence, thus enabling broader context exploration.

Concatenation-based Contextual Modeling We adopt the concatenation method to incorporate the previous context ($\mathcal{C}_{\mathbf{x}}^n$/$\mathcal{C}_{\mathbf{y}}^n$) (Tiedemann and Scherrer, 2017; Bawden et al., 2018) as shown in Figure 1. After obtaining the AFS-based encodings ($\mathbf{x}^n$) for each audio segment, we concatenate those encodings of neighboring segments to form the source input. The same is applied to the target-side sentences, except for a separator symbol "<s>" inserted in-between sentences to distinguish sentence boundaries.[3] Such modeling enables us to use arbitrary encoder-decoder models for context-aware ST, such as the Transformer (Vaswani et al., 2017) used in this paper. Despite no dedicated hierarchical modeling (Miculicich et al., 2018), this paradigm still allows for intra- and inter-sentence attention during encoding and decoding, which explicitly utilizes context for translation and has been proven successful (Lopes et al., 2020).

4 Inference

Concatenation-based contextual modeling allows for different inference strategies with possible trade-offs between simplicity/efficiency and accuracy. We investigate the following inference strategies (see Figure 2):

Chunk-based Decoding (CBD) CBD splits all audio segments in one document into non-overlapping chunks, with each chunk concatenating $C + 1$ segments, as shown in Figure 2a. CBD directly translates each chunk, and then recovers sentence-level translation via the separator symbol "<s>". CBD is the most efficient inference strategy, only encoding/decoding each sentence once, but it might suffer from *misaligned translation*,

[3] Note that we did not add similar boundary information to audio segments, because AFS implicitly captures these signals through independent segment encoding.

producing more or fewer sentences than the input segments. We simply drop the extra generated sentences and replace the missing ones with "<unk>" when computing sentence-based evaluation metrics. Also, CBD introduces an independence assumption between chunks.

Sliding Window-based Decoding (SWBD) SWBD avoids such inter-chunk independence by sequentially translating each audio segment ($\mathbf{x}^n$), together with its corresponding previous source context ($\mathcal{C}^n_{\mathbf{x}}$). We distinguish two variants of SWBD. The first variant, SWBD, translates the concatenated segments and regards the last generated sentence as the translation of the current segment while discarding all other generations (Figure 2b). Note that this might introduce inconsistencies between the output produced at a time step, and the one used as target context in future time steps. By contrast, the second variant, SWBD-Cons, leverages the previously generated (up to C) sentences as a decoding constraint, based on which the model only needs to generate one sentence (Figure 2c).

In-Model Ensemble Decoding (IMED) We observe that SWBD still suffers from *misaligned translation*, where the translation of the current segment might contain information from previous segments. We introduce IMED to alleviate this issue as shown in Figure 2d. IMED extends SWBD-Cons by interpolating the document-level prediction (p^d) with the sentence-level prediction (p^s) as follows:

$$\lambda p^s_\theta\left(\mathbf{y}^n_t|\mathbf{y}^n_{<t},\mathbf{x}^n\right)+(1-\lambda)p^d_\theta\left(\mathbf{y}^n_t|\mathcal{C}\right), \quad (2)$$

where $\mathcal{C}=\{\mathcal{C}^n_{\mathbf{x}},\mathcal{C}^n_{\mathbf{y}},\mathbf{x}^n,\mathbf{y}^n_{<t}\}$, λ is a hyperparameter, $\mathbf{y}^n_t$ denotes the t-th target word in sentence $\mathbf{y}^n$, and both predictions are based on *the same model* θ. Intuitively, the sentence-level translation acts as a regularizer, avoiding the over- or under-translation. Note IMED with $\lambda=0$ corresponds to SWBD-Cons.

5 Experiments

5.1 Setup

We use the MuST-C dataset (Di Gangi et al., 2019) for experiments, which was collected from English TED talks and covers translations from English to 8 different languages, including German (De), Spanish (Es), French (Fr), Italian (It), Dutch (Nl), Portuguese (Pt), Romanian (Ro) and Russian (Ru). MuST-C offers a standard training, development and test set split for each language pair, with each dataset consisting of English audio, English transcriptions and their translations. Each training set contains transcribed speeches of ∼452 hours with ∼252K utterances on average. We report results on tst-COMMON, whose size ranges from 2502 (Es) to 2641 (De) utterances. We perform our major study on MuST-C En-De.

To construct acoustic features, for each audio segment, we extract 40-channel log-Mel filterbanks using overlapping windows of 25 ms and step size of 10 ms. We enrich these features with their first and second-order derivatives, followed by mean subtraction and variance normalization. Following Zhang et al. (2020a), we perform non-overlapping feature stacking to combine the features of three consecutive frames. All the texts are tokenized and truecased (Koehn et al., 2007), with out-of-vocabulary words handled by BPE segmentation (Sennrich et al., 2016), using 16K merging operations.

Model Settings and Evaluation Our context-aware ST follows Transformer base (Vaswani et al., 2017): 6 layers, 8 attention heads, and hidden/feed-forward size 512/2048. We use Adam ($\beta_1=0.9,\beta_2=0.98$) (Kingma and Ba, 2015) for parameter updates with label smoothing of 0.1. We use the same learning rate schedule as Vaswani et al. (2017) and set the warmup step to 4K. We apply dropout to attention weights and residual connections with a rate of 0.2 and 0.5, respectively. By default, we set $C=2$ and $\lambda=0.5$. Following (Zhang et al., 2020a), we apply AFS($\epsilon=-0.1,\beta=2/3$) to both temporal and feature dimensions for feature selection, which prunes out ∼84% speech encodings. We initialize our context-aware ST with the sentence-level Baseline, i.e. ST+AFS, and then finetune the model for 20K steps based on the concatenation method with a batch size of around 40K subwords.[4] We adopt beam search for decoding, with a beam size of 4 and length penalty of 0.6. We average the last 5 checkpoints for evaluation.

We measure general translation quality with tokenized case-sensitive BLEU (Papineni et al., 2002) and also report the detokenized one via *sacreBLEU* (Post, 2018)[5] for cross-paper comparison. We calculate BLEU based on sentences unless oth-

[4]Our experiments show that such initialization eases the learning of long inputs and improves the convergence of context-aware ST.

[5]signature: BLEU+c.mixed+#.1+s.exp+tok.13a+v.1.3.6

ID	Model	BLEU	APT
1	Baseline (ST+AFS)	22.38 (27.40)	60.77
2	Ours + CBD	22.72 (27.95)	62.31
3	Ours + SWBD	22.70 (28.02)	**62.83**
4	Ours + SWBD-Cons	22.11 (27.98)	60.94
5	Ours + IMED	**22.86** (**28.03**)	62.56
6	1 + 20K-step finetuning	22.02 (27.00)	61.58
7	5 + $\lambda = 1.0$	22.42 (27.62)	61.96
8	1 + *lp* = 1.0	22.71 (27.77)	61.89
9	3 + *lp* = 1.0	**22.97** (**28.29**)	**63.51**
10	5 + *lp* = 1.0	22.94 (28.11)	62.76
11	3 w/o $\mathcal{C}_{\mathbf{y}}^{n}$	21.12 (26.17)	59.51
12	5 w/o $\mathcal{C}_{\mathbf{y}}^{n}$	20.72 (25.43)	58.18
13	3 w/o Baseline Initial.	21.75 (27.15)	62.29
14	5 w/o Baseline Initial.	21.97 (27.20)	62.08

Table 1: Case-sensitive tokenized BLEU and APT for different models and settings on MuST-C En-De test set. Numbers in bracket denote *document*-based BLEU. *lp*: the length penalty for beam search decoding. "w/o $\mathcal{C}_{\mathbf{y}}^{n}$": models that are trained without target-side context. Best results are highlighted in bold. Note $C = 2$, $\lambda = 0.5$ and *lp* = 0.6 by default.

erwise specified. We use APT (Miculicich Werlen and Popescu-Belis, 2017), the accuracy of pronoun translation, as an approximate proxy for document-level evaluation. Word alignment required by APT is automatically extracted via *fast align* (Dyer et al., 2013) with the strategy "*grow-diag-final-and*".

5.2 Results on MuST-C En-De

Does context improve translation? Yes, but the decoding method matters for context-aware ST. Table 1 summarizes the results. Our model with IMED outperforms Baseline by +0.48 BLEU (significant at $p < 0.05$)[6] and +1.79 APT (1→5), clearly showing the benefits from contextual modeling. Although SWBD-Cons yields worse sentence-based BLEU (-0.27, 1→4), it still beats Baseline in document-based BLEU (+0.58) and pronoun translation (+0.17 APT). The reason behind this inferior BLEU partially lies in misaligned translation (see Table 8 in Appendix for example). We observe that SWBD-Cons sometimes segments its output in a way that is misaligned to the reference segmentation. This also hurts CBD, where CBD produces mismatched sentences for around 1.8% cases. This is only a problem if we rely on the sentence-level alignment for BLEU, but not when we measure document-based BLEU (in brackets), where translations in one document are concatenated into a sequence for BLEU calculation. Overall, SWBD and IMED are more stable and perform the best, and SWBD surpasses Baseline by 2.06 APT (1→3). We will proceed with using IMED and SWBD for more reliable results with APT and later analysis.

Since we finetune our model based on the pre-trained Baseline, directly comparing with Baseline might be unfair. To offset its influence, we continue to train Baseline for the same 20K steps, following the settings in Section 5.1. Results show that this extra training (1→6) slightly deteriorates BLEU (-0.36) and only explains part of the improvement in APT (+0.81). Therefore, the gain brought by SWBD and IMED does not come from longer training. However, we do observe that initializing from the sentence-level Baseline benefits context-aware ST, compared to directly training context-aware ST from the AFS model (13→3, 14→4).

Apart from faster convergence and higher quality, another benefit of this finetuning is that the trained context-aware ST still carries the ability to translate individual sentences. Table 1 shows that using context-aware ST for sentence-level translation (1→7) yields similar BLEU to Baseline (+0.04) but surprisingly much better pronoun translation (+1.19), although it still underperforms SWBD and IMED. The fact that we can perform sentence-level ST using the same context-aware ST model indicates that it can be useful for ensembling, as confirmed by the effectiveness of IMED.

Upon closer inspection, we find that context-aware ST prefers to produce longer translations than Baseline. To control for the effects of output length on BLEU differences, we experiment with larger length penalty (*lp*: 0.6→1.0) to beam search. Results in Table 1 show that biasing the decoding greatly improves sentence-level ST (1→8), achieving performance on par with context-aware ST (when *lp* is 0.6) in terms of BLEU with similar translation lengths but still falling short of pronoun translation (-0.94 APT, 8→3). In addition, we observe that context-aware ST also benefits from decoding with larger length penalty, beating all sentence-level ST models (3→9, 5→10). Particularly, SWBD with *lp* of 1.0 delivers the best BLEU of 22.97 and APT of 63.51 (3→9). Note we adopt *lp* of 0.6 for the following experiments.

Does target-side context matter for context-aware ST? Yes, it matters a lot. By default, we utilize both source- and target-side context for contextual modeling. Removing the target-side part (also at training), as shown in Table 1 (11, 12), sub-

[6] We perform significance test using *bootstrap-hypothesis-difference-significance.pl* in *moses* (Koehn et al., 2007).

Model	BLEU	APT
SWBD	22.70	62.83
SWBD + Random $\mathcal{C}_{\mathbf{x}}^{n}$	22.31	61.16
IMED	22.86	62.56
IMED + Random $\mathcal{C}_{\mathbf{x}}^{n}$	21.83	59.95
IMED + Random $\mathcal{C}_{\mathbf{y}}^{n}$	21.99	60.01
IMED + Random $\mathcal{C}_{\mathbf{y}}^{n}$ & $\mathcal{C}_{\mathbf{y}}^{n}$	21.76	59.67

Table 2: Case-sensitive tokenized BLEU and APT for context-aware ST with random source/target context on MuST-C En-De test set. We report average performance over three runs with different random seeds. $C = 2, \lambda = 0.5$. Incorrect context hurts our model.

stantially weakens translation quality, even leading to worse performance than Baseline. Apart from offering direct target-side translation clues, we argue that the target-side context also enforces the context-aware ST to utilize the source-side context for translation, thus benefiting its training. This observation echoes with several previous studies on textual translation (Bawden et al., 2018; Huo et al., 2020; Lopes et al., 2020).

Does the model learn to utilize context? Yes. We answer this question by studying the impact of incorrect context on our model. We replace the correct source context with some random audio segments from the same document, and randomly select the target context from previous translations during decoding. Intuitively, the performance of our model should be intact if it ignores the context. Note that we trained our model with correct contexts but test it with random contexts here.

Results in Table 2 show that the randomized context, either source- or target-side, hurts the performance of our model in both BLEU and APT, similar to the findings in (Voita et al., 2018), and the translation of pronouns suffers more ($>$ -1.6 APT). Compared to SWBD, the incorrect context has more negative impact on IMED, resulting in worse performance than Baseline (Table 1), although IMED also uses sentence-level translation. We ascribe this to the target prefix constraint in IMED which makes translation errors at early decoding much easier to propagate. We observe that the incorrect target context acts similarly to its source counterpart under IMED, albeit its selection scope is much smaller (only limited to the translated segments), and combining both contexts leads to a slight but consistent performance degradation. These results demonstrate that our model indeed learns to use contextual information for translation.

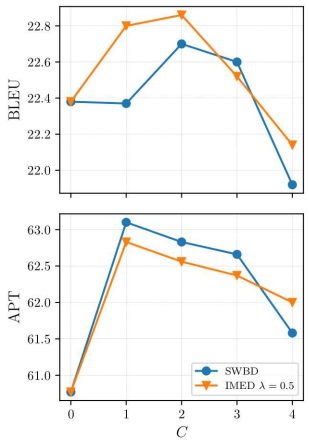


Figure 3: Case-sensitive tokenized BLEU (top) and APT (bottom) as a function of context size C on MuST-C En-De test set.

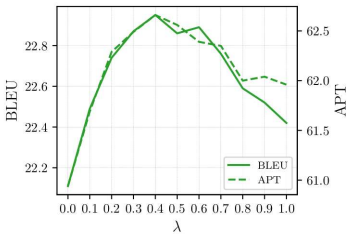


Figure 4: Case-sensitive tokenized BLEU (left y-axis) and APT (right y-axis) on MuST-C En-De test set when varying λ for IMED. Solid and dashed curves are for BLEU and APT, respectively. $C = 2$.

How much context sentences should we use? Although adding extra context provides more information, it makes learning harder: neural models often struggle with long sequences. Figure 3 shows the impact of context size on translation. We find that our models do not benefit from context size beyond 2 previous segments. Figure 3 also shows that the overall trend of the impact of C on BLEU and APT is similar for different decoding methods. Increasing C to 1 delivers the best APT, while context-aware ST achieves its best BLEU at $C = 2$. We use $C = 2$ for the following experiments.

Impact of λ on IMED. IMED heavily relies on the hyperparameter λ (Eq. 2) to control its preference between sentence-level and document-level decoding. Figure 4 shows its impact on translation

Model	ACC_{hp}
Baseline (ST+AFS)	48.93
Ours + SWBD	**49.90**
Ours + IMED	49.66
Ours + IMED $\lambda = 1.0$	48.77

Table 3: Translation accuracy of homophones (ACC_{hp}) on MuST-C En-De test set. $C = 2, \lambda = 0.5$.

Model	Random	Gold
Baseline (ST+AFS)	20.40	27.40
Ours + SWBD	21.83	28.02
Ours + IMED	**22.03**	**28.03**

Table 4: *Document*-level case-sensitive tokenized BLEU for different models on MuST-C En-De test set with erroneous audio segmentation. We report average BLEU over three runs; each run uses a different random seed to simulate segmentation errors. $C = 2, \lambda = 0.5$. *Random/Gold*: document-based BLEU when the random/gold segments are used.

quality, which clearly reveals a trade-off. The performance of IMED (BLEU and APT) reaches its peak at $\lambda = 0.4$, and decreases when λ becomes either smaller or larger. The optimal value of λ for IMED might vary greatly across different language pairs. It also shows some difference across evaluation sets (see Figure 7 in Appendix). In the following experiments, we will apply equal weighting ($\lambda = 0.5$), a common choice for model ensembles and not substantially worse than the optimum on this dataset.

Impact of context on homophone translation. Homophones (words that sound the same but hold different meanings, such as "*I*" vs. "*eye*" and "*would*" vs. "*wood*") and other acoustically similar words increase the learning difficulty of ST models compared to textual MT. To allow for automatic quantitative evaluation, we extract words from the MuST-C test set transcriptions which share the same phonemes with *Montreal Forced Aligner* (McAuliffe et al., 2017). We collect all homophones and evaluate their translation accuracy (ACC_{hp}) in the same way as APT.

Table 3 shows that context-aware ST outperforms Baseline by > 0.73 ACC_{hp}, where SWBD performs slightly better than IMED. After removing the document-level decoding, IMED ($\lambda = 1.0$) performance drops greatly, even underperforming Baseline. While we see some improvements to homophone translations, they are in the same relative range as general improvements from context. Anecdotal examples from manual inspection (see Table 7 in Appendix) indicate that context may at times help disambiguate acoustically similar forms, but that (near-)homophones still remain a salient source of translation errors.

Context improves the robustness of ST models to audio segmentation errors. In MuST-C, the audio is already well-segmented, with each segment corresponding to a short transcript. Nevertheless, natural audio, streaming speeches in particular, has no such segment boundaries, and how to partition audio itself is an active research area (Rangarajan Sridhar et al., 2013; Zhang and Zhang, 2020). Since ST models are often trained with gold segments, they inevitably suffer from segmentation errors at inference when the gold ones are unavailable.

The bottleneck mainly comes from the incompleteness of each segment, which, we argue, contextual information could alleviate. We simulate segmentation errors by randomly re-segmenting the audio in MuST-C En-De test set based on the given segment number. Especially, given an audio with N gold segments, we randomly re-segment it into N disjoint pieces, where each piece usually has different boundaries against its gold counterpart.[7] We evaluate different ST models with document-based BLEU.

Table 4 summarizes the results. Segmentation noise deteriorates translation quality for all ST models to a large degree ($>$ -6 BLEU). Compared to sentence-level ST, context-aware ST is less sensitive to those errors. In particular, our model with IMED yields a document-based BLEU of 22.03, substantially outperforming Baseline (by 1.63 BLEU). Our results also confirm the findings of Gaido et al. (2020).

Context benefits simultaneous translation. Simultaneous translation requires that we start decoding before receiving the whole audio input to minimize latency; operating on such short units increases ambiguity, and the model may be forced to predict future input to account for word order differences, which we hypothesize is easier with access to super-sentential context. We focus on segment-

[7]Note we intentionally keep the same segment number, N, in the simulated noisy segmentation, because this offers us a fair setup to analyze the impact of segmentation errors on the final translation when compared to the gold segmentation. This avoids the potential influence resulting from mismatched segment number. We leave the study of the model's robustness to genuine segmentation noises to future work.

Metric	Model	De	Es	Fr	It	Nl	Pt	Ro	Ru
BLEU↑	Baseline (ST+AFS)	22.38	27.04	33.43	23.35	25.05	26.55	21.87	14.92
	Ours + SWBD	22.70	27.12	34.23	23.46	25.84	26.63	23.70	15.53
	Ours + IMED	**22.86**	**27.50**	**34.28**	**23.53**	**26.12**	**27.37**	**24.48**	**15.95**
SacreBLEU↑	Baseline (ST+AFS)	22.4	26.9	31.6	23.0	24.9	26.3	21.0	14.7
	Ours + SWBD	22.7	27.0	32.4	23.0	25.7	26.4	22.8	15.4
	Ours + IMED	**22.9**	**27.3**	**32.5**	**23.1**	**26.0**	**27.1**	**23.6**	**15.8**
APT↑	Baseline (ST+AFS)	60.77	32.87	63.67	34.74	61.00	34.79	38.28	40.61
	Ours + SWBD	**62.83**	33.01	64.58	**35.20**	61.69	35.56	40.30	41.74
	Ours + IMED	62.56	**33.60**	**64.66**	**35.20**	**61.75**	**36.50**	**40.92**	**42.32**
ACC_{hp}↑	Baseline (ST+AFS)	48.93	43.85	56.96	**41.08**	50.73	43.64	47.07	30.80
	Ours + SWBD	**49.90**	43.73	57.30	40.04	51.48	44.03	47.66	**32.67**
	Ours + IMED	49.66	**44.66**	**57.76**	40.62	**52.07**	**45.42**	**48.49**	32.56

Table 5: Results on MuST-C for 8 language pairs. We set $C = 2, \lambda = 0.5$. Numbers in bold are the best results.

Model	BLEU↑	DAL↓	NE↓
Baseline (ST+AFS)	21.02	3.97	1.72
Ours + SWBD	21.86	3.82	1.95
Ours + SWBD-Cons	21.98	**3.75**	**1.59**
Ours + IMED	**22.55**	3.91	1.64

Table 6: Simultaneous translation results (BLEU, DAL and NE) for different models on MuST-C En-De test set. $C = 2, \lambda = 0.5$.

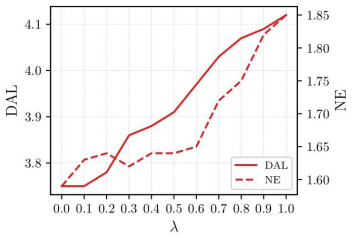


Figure 5: DAL (left y-axis) and NE (right y-axis) as a function of λ for IMED on MuST-C En-De test set in simultaneous translation setting. Solid and dashed curves are for DAL and NE, respectively. $C = 2$. $\lambda \to 0.0$: document-level decoding; $\lambda \to 1.0$: sentence-level decoding.

level E2E simultaneous translation, and adopt the re-translation method (Niehues et al., 2016; Arivazhagan et al., 2020b,a) where we translate the source input segment from scratch after every 1 second. For training, we finetune each model for extra 20K steps with a 1:1 mix of full-segment and prefix pairs, following Arivazhagan et al. (2020a). We construct the prefix pairs by uniformly selecting an audio prefix length and then proportionally deciding the target prefix length based on the sentence length. Note that the context inputs in our model are still full segments/sentences. We adopt tokenized BLEU, differentiable average lagging (DAL), and normalized erasure (NE) to evaluate the translation quality, latency and stability, respectively, following Arivazhagan et al. (2020a). Note DAL and NE are measured based on words.

Results in Table 6 show that context-aware ST improves translation quality (> +0.84 BLEU) and reduces translation latency (> -0.06 DAL) regardless of the decoding method. It also enhances translation stability when the target prefix constraint is applied (> -0.08 NE, SWBD-Cons & IMED). SWBD performs worse in NE, because it allows changes in the translation of context which increases instability. Overall, context provides extra information to the translation model, before the E2E ST models see the whole input, which benefits simultaneous translation.

Figure 5 further illustrates how context impacts simultaneous translation. With the increase of sentence-level decoding ($\lambda \to 1.0$), IMED produces higher DAL and NE, i.e. worse quality. We ascribe the reduction of latency and stability in our model to the inclusion of contextual information.

5.3 Results on Other Language Pairs

Table 5 summarizes the results for all 8 translation pairs covered by MuST-C. Overall, our model obtains improvements over most metrics and language pairs, despite their different language characteristics. Out of 8 languages, our model performs relatively worse on Es and It with smaller BLEU gains and even negative results in ACC_{hp}. By contrast, our model yields the largest improvement on Ro. In particular, our model with IMED achieves a detokenized BLEU of 23.6 on En-Ro, surpassing the state-of-the-art result 22.2 (Zhao et al., 2020) reported so far.

6 Conclusion and Future Work

Our experiments confirm the effectiveness of context-aware modeling for end-to-end speech translation. With concatenation-based contextual modeling and appropriate decoding method, we observe positive impact of context on translation. Context-aware ST improves general translation quality in BLEU, and also helps pronoun and homophone translation. ST models become less sensitive to (artificial) audio segmentation errors with context. In addition, context also improves simultaneous translation by reducing latency and erasure. We observe overall positive results over different languages and evaluation metrics on the MuST-C corpus.

In the future, we will investigate more dedicated neural architectures to handle long-form speech input. While we relied on a dataset with sentence segmentation in this work, we are interested in removing the reliance on segmentation at inference time to implement the full-fledged streaming translation scenario.

Acknowledgements

We thank the reviewers for their insightful comments. This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreements 825460 (ELITR). Rico Sennrich acknowledges support of the Swiss National Science Foundation (MUTAMUR; no. 176727).

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 82–91, New Orleans, Louisiana. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020a. Re-translation versus streaming for simultaneous translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 220–227, Online. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Isabelle Te, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020b. Re-translation strategies for long form, simultaneous, spoken language translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7919–7923. IEEE.

Rachel Bawden, Rico Sennrich, Alexandra Birch, and Barry Haddow. 2018. Evaluating discourse phenomena in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1304–1313, New Orleans, Louisiana. Association for Computational Linguistics.

Xinyi Cai and Deyi Xiong. 2020. A test suite for evaluating discourse phenomena in document-level neural machine translation. In *Proceedings of the Second International Workshop of Discourse Processing*, pages 13–17, Suzhou, China. Association for Computational Linguistics.

Junxuan Chen, Xiang Li, Jiarui Zhang, Chulun Zhou, Jianwei Cui, Bin Wang, and Jinsong Su. 2020. Modeling discourse structure for document-level neural machine translation. In *Proceedings of the First Workshop on Automatic Simultaneous Translation*, pages 30–36, Seattle, Washington. Association for Computational Linguistics.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. MuST-C: a Multilingual Speech Translation Corpus. 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 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting Transformer to End-to-End Spoken Language Translation. In *Proc. Interspeech 2019*, pages 1133–1137.

Qianqian Dong, Mingxuan Wang, Hao Zhou, Shuang Xu, Bo Xu, and Lei Li. 2020. Sdst: Successive decoding for speech-to-text translation. *arXiv preprint arXiv:2009.09737*.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, Mauro Cettolo, and Marco Turchi. 2020. Contextualized Translation of Automatically Segmented Speech. In *Proc. Interspeech 2020*, pages 1471–1475.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until

the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352, Doha, Qatar. Association for Computational Linguistics.

Liane Guillou, Christian Hardmeier, Ekaterina Lapshinova-Koltunski, and Sharid Loáiciga. 2018. A pronoun test suite evaluation of the English–German MT systems at WMT 2018. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 570–577, Belgium, Brussels. Association for Computational Linguistics.

Liane Kirsten Guillou. 2016. *Incorporating pronoun function into statistical machine translation*. Ph.D. thesis, University of Edinburgh.

Christian Hardmeier, Joakim Nivre, and Jörg Tiedemann. 2012. Document-wide decoding for phrase-based statistical machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1179–1190, Jeju Island, Korea. Association for Computational Linguistics.

Jingjing Huo, Christian Herold, Yingbo Gao, Leonard Dahlmann, Shahram Khadivi, and Hermann Ney. 2020. Diving deep into context-aware neural machine translation. In *Proceedings of the Fifth Conference on Machine Translation*, pages 604–616, Online. Association for Computational Linguistics.

Javier Iranzo-Sánchez, Adrià Giménez Pastor, Joan Albert Silvestre-Cerdà, Pau Baquero-Arnal, Jorge Civera Saiz, and Alfons Juan. 2020. Direct segmentation models for streaming speech translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2599–2611, Online. Association for Computational Linguistics.

Xiaomian Kang, Yang Zhao, Jiajun Zhang, and Chengqing Zong. 2020. Dynamic context selection for document-level neural machine translation via reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2242–2254, Online. Association for Computational Linguistics.

Yunsu Kim, Duc Thanh Tran, and Hermann Ney. 2019. When and why is document-level context useful in neural machine translation? In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 24–34, Hong Kong, China. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Shaohui Kuang, Deyi Xiong, Weihua Luo, and Guodong Zhou. 2018. Modeling coherence for neural machine translation with dynamic and topic caches. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 596–606, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4796, Brussels, Belgium. Association for Computational Linguistics.

António Lopes, M. Amin Farajian, Rachel Bawden, Michael Zhang, and André F. T. Martins. 2020. Document-level neural MT: A systematic comparison. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 225–234, Lisboa, Portugal. European Association for Machine Translation.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020a. A simple and effective unified encoder for document-level machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3505–3511, Online. Association for Computational Linguistics.

Xutai Ma, Yongqiang Wang, Mohammad Javad Dousti, Philipp Koehn, and Juan Pino. 2020b. Streaming simultaneous speech translation with augmented memory transformer. *arXiv preprint arXiv:2011.00033*.

Evgeny Matusov, Dustin Hillard, Mathew Magimai-Doss, Dilek Hakkani-Tür, Mari Ostendorf, and Hermann Ney. 2007. Improving speech translation with automatic boundary prediction. In *Eighth Annual Conference of the International Speech Communication Association*.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pages 498–502.

Lesly Miculicich, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *Proceedings of the 2018 Conference*

on Empirical Methods in Natural Language Processing, pages 2947–2954, Brussels, Belgium. Association for Computational Linguistics.

Lesly Miculicich Werlen and Andrei Popescu-Belis. 2017. Validation of an automatic metric for the accuracy of pronoun translation (APT). In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 17–25, Copenhagen, Denmark. Association for Computational Linguistics.

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic transcription for low-latency speech translation. In *Interspeech 2016*, pages 2513–2517.

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.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 230–238, Atlanta, Georgia. Association for Computational Linguistics.

Annette Rios, Laura Mascarell, and Rico Sennrich. 2017. Improving word sense disambiguation in neural machine translation with sense embeddings. In *Proceedings of the Second Conference on Machine Translation*, pages 11–19, Copenhagen, Denmark. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. 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.

Jörg Tiedemann and Yves Scherrer. 2017. Neural machine translation with extended context. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 82–92, Copenhagen, Denmark. Association for Computational Linguistics.

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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Chengyi Wang, Yu Wu, Shujie Liu, Ming Zhou, and Zhenglu Yang. 2020. Curriculum pre-training for end-to-end speech translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3728–3738, Online. Association for Computational Linguistics.

Deyi Xiong and Min Zhang. 2013. A topic-based coherence model for statistical machine translation. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, page 977–983. AAAI Press.

Biao Zhang, Ivan Titov, Barry Haddow, and Rico Sennrich. 2020a. Adaptive feature selection for end-to-end speech translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2533–2544, Online. Association for Computational Linguistics.

Biao Zhang, Ivan Titov, and Rico Sennrich. 2020b. On sparsifying encoder outputs in sequence-to-sequence models. *arXiv preprint arXiv:2004.11854*.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018. Improving the transformer translation model with document-level context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542, Brussels, Belgium. Association for Computational Linguistics.

Pei Zhang, Boxing Chen, Niyu Ge, and Kai Fan. 2020c. Long-short term masking transformer: A simple but effective baseline for document-level neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1081–1087, Online. Association for Computational Linguistics.

Ruiqing Zhang and Chuanqiang Zhang. 2020. Dynamic sentence boundary detection for simultaneous translation. In *Proceedings of the First Workshop*

on Automatic Simultaneous Translation, pages 1–9, Seattle, Washington. Association for Computational Linguistics.

Chengqi Zhao, Mingxuan Wang, and Lei Li. 2020. Neurst: Neural speech translation toolkit. *arXiv preprint arXiv:2012.10018*.

Zaixiang Zheng, Xiang Yue, Shujian Huang, Jiajun Chen, and Alexandra Birch. 2020. Towards making the most of context in neural machine translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3983–3989. International Joint Conferences on Artificial Intelligence Organization. Main track.

A Impact of C and λ on Dev Set

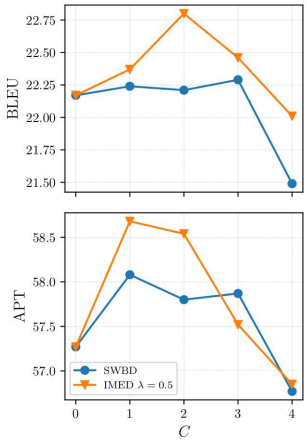


Figure 6: Case-sensitive tokenized BLEU (top) and APT (bottom) as a function of context size C on MuST-C En-De **dev set**.

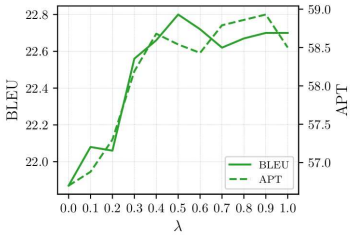


Figure 7: Case-sensitive tokenized BLEU (left y-axis) and APT (right y-axis) on MuST-C En-De **dev set** when varying λ for IMED. Solid and dashed curves are for BLEU and APT, respectively. $C = 2$.

Results in Figure 6 and 7 show that the optimal value of C and λ also differs across evaluation sets. Overall, setting $C = 2$ and $\lambda = 0.5$ offers us decent performance. Note again, we selected these configurations for generality and simplicity rather than its being optimal.

B Case Study on Homophone Translation

C Examples for Misaligned Translation

Context	I remember my first fire.
Source	I was the second volunteer on the **scene**, so there was a pretty good chance I was going to get in.
Reference	Ich war der zweite Freiwillige an der **Brandstelle**, ich hatte also recht gute Chancen hinein zu können.
Baseline	Ich war der zweite Freiwillige auf der **CNU**, also war ich ziemlich gut darin.
Ours + SWBD	Ich war der zweite Freiwillige auf der **CNN**, also gab es eine ziemlich gute Chance, dass ich sie bekommen würde.
Ours + SWBD-Cons	Ich war der zweite Freiwillige auf dem **CNN**, also gab es eine ziemlich gute Chance, dass ich sie bekommen würde.
Ours + IMED	Ich war der zweite Freiwillige auf dem **CNN**, also war ich ziemlich gut darin, dass ich ihn kriegen würde.
Context	The Human Genome Project started in 1990, and it took 13 years.
Source	It **cost** 2.7 billion dollars.
Reference	Es **kostete** 2,7 Milliarden Dollar.
Baseline	Es **kostet** 2,7 Milliarden Dollar. *(EN: costs)*
Ours + SWBD	Es **kostete** 2,7 Milliarden Dollar.
Ours + SWBD-Cons	Es **kostete** 2,7 Milliarden Dollar.
Ours + IMED	Es **kostet** 2,7 Milliarden Dollar. *(EN: costs)*

Table 7: Examples of translation errors due to confusion with near-homophones (bold) from the MuST-C En-De test set.

(1)	Source	She asked the monk, "Why is it that her hand is so warm and the rest of her is so cold?" "Because you have been holding it since this morning," **he said. "You have not let it go."**
	Reference	Sie fragte den Mönch: "Wieso ist ihre Hand so warm und der Rest von ihr ist so kalt?" "Weil Sie sie seit heute morgen halten", **sagte er. "Sie haben sie nicht losgelassen."**
	Translation	Sie fragte den Monat: "Warum ist ihre Hand so warm?" Und der Rest von ihr ist so kalt, weil ihr seit diesem Morgen das hält.
(2)	Source	If there is a sinew in our family, it runs through the women.
	Reference	Wenn es in unserer Familie ein Band gibt, dann verläuft es durch die Frauen.
	Translation	**Er sagte: "Sie haben es nicht geschafft, loszulassen."**

Table 8: Example of misaligned translation for SWBD-Cons from the MuST-C En-De test set. The translation for the second segment (2) actually aligns with the first one (1), as highlighted in bold.

G Revisiting End-to-End Speech-to-Text Translation From Scratch

Revisiting End-to-End Speech-to-Text Translation From Scratch

Anonymous Authors[1]

Abstract

End-to-end (E2E) speech-to-text translation (ST) often depends on pretraining its encoder and/or decoder on speech recognition or text translation tasks, without which translation performance drops substantially. However, whether such pretraining is a necessity for E2E ST has rarely been studied in the literature. In this paper, we revisit this question and explore the extent to which the quality gap between models with and without pretraining can be narrowed. We reexamine several techniques proven beneficial to ST previously, and offer a set of best practices that biases a Transformer-based E2E ST system toward training from scratch. Besides, we propose parameterized distance penalty to facilitate the modeling of locality in the self-attention model for speech. On four benchmarks covering 23 languages, our experiments show that, without any pretraining, the proposed system reaches and even outperforms previous studies adopting pretraining, although the gap remains in (extremely) low-resource settings. Finally, we discuss neural acoustic feature modeling, where a neural model is designed to extract acoustic features from raw speech signals directly, with the goal to simplify inductive biases and add freedom to the model in describing speech. For the first time, we demonstrate its feasibility and show encouraging results on ST tasks. Source code will be released upon acceptance.

1. Introduction

End-to-end (E2E) speech-to-text translation (ST) is the task of translating a source-language audio directly to a foreign text without any intermediate outputs (Duong et al., 2016; Bérard et al., 2016), which has gained increasing popularity and obtained great success recently (Sung et al., 2019; Salesky et al., 2019; Zhang et al., 2020; Chen et al., 2020; Han et al., 2021; Zheng et al., 2021; Anastasopoulos et al., 2021). Different from the traditional cascading method which decomposes ST into two sub-tasks – automatic speech recognition (ASR) for transcription and machine translation (MT) for translation, E2E ST jointly handles them in a single, large neural network. This endows E2E ST with special advantages on reducing translation latency and bypassing transcription mistakes made by ASR models, making it theoretically attractive.

However, directly modeling speech-to-text mapping is non-trivial. The translation alignment between speech and text is no longer subject to the monotonic assumption. Also, the high variation of speech increases the modeling difficulty. Therefore, rather than training E2E ST models from scratch, researchers often resort to pipeline-based training with auxiliary tasks, which first pretrains the speech encoder on ASR data and/or the text decoder on MT data followed by a finetuning on ST data. Such pretraining was reported to greatly improve translation quality (Di Gangi et al., 2019; Wang et al., 2019a; Zhang et al., 2020; Xu et al., 2021), and has become the *de-facto* standard in recent ST studies and toolkits (Inaguma et al., 2020; Wang et al., 2020a; Zhao et al., 2021; Zheng et al., 2021). Despite its success, nevertheless, whether the pretraining is a necessity for E2E ST and how far we can go without it are still open questions.

In this paper, we aim at exploring the extent to which the quality gap between ST models with and without pretraining can be narrowed, and also when the pretraining really matters. We argue that the inferior performance of ST from scratch is mainly a result of the dominance of pretraining, and consequent lack of focus on optimizing E2E ST models trained from scratch. To test this hypothesis, we investigate methods to bias a Transformer-based E2E ST model (Vaswani et al., 2017) towards training from scratch. We summarize a set of best practices for our setup by revisiting several existing techniques that have been proven useful to ST previously. We further introduce two proposals to add freedom to Transformer to model speech with the hope of gaining translation quality: 1) a parameterized distance penalty that facilitates self-attention to capture local dependencies of speech; and 2) neural acoustic feature modeling providing a trainable alternative to the heuristic rule-based acoustic feature extraction.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

To examine the generality of our methods, we conducted (bilingual) experiments on four speech translation benchmarks, including MuST-C, Covost2, LibriSpeech, and Kosp2e, which cover 23 languages of different families with training data sizes. Experimental results show that the necessity of pretraining has been over-estimated in prior work, and integrating techniques to improve E2E ST without using any pretraining is feasible and promising. Our main findings are listed below:

- With proper adaptation, E2E ST trained on speech-translation pairs alone from scratch can match or even surpass its pretrained counterpart.
- Pretraining still matters, mainly in (extremely) low-resource regime and when large-scale extra ASR or MT corpora are available.
- We present a set of best practices for E2E ST from scratch, including smaller vocabulary size, wider feed-forward layer, deep speech encoder with the post-LN (layer normalization) structure, Connectionist Temporal Classification (CTC)-based regularization using translation as the target, and a novel parameterized distance penalty.
- We demonstrate that dropping heuristic rule-based acoustic features is feasible, and that neural acoustic features can be learned in an end-to-end ST framework.

2. Why Revisiting ST *From Scratch*?

In our view, there are several reasons making E2E ST from scratch intriguing.

First of all, our study does not preclude pretraining (or more generally, multi-task learning) for ST. We believe that leveraging knowledge from auxiliary tasks via pretraining to improve ST is a remarkable research direction. But rather, our study contributes to a better understanding of the genuine role of pretraining in E2E ST. Re-assessing the importance of pretraining is a useful signal to inform future research projects and practical deployments of ST.

Secondly, focusing on ST from scratch has an even higher relevance in settings where ASR/MT data is scarce. By only requiring speech-translation training pairs, ST from scratch reduces data requirements and the associated costs. This is especially important for the estimated 3000 languages in the world that have no written form at all, for which it would be impractical to collect large amounts of phonetically transcribed data.

Thirdly, removing pretraining eases model analysis and simplifies the training pipeline, which also offers a testbed to identify inductive biases that support ST with better data efficiency. Pretraining often takes extra training time and computing resources. As pretraining itself affects the final results, it becomes more difficult to figure out the source of the improved performance when new algorithms or architectures are incorporated. In contrast, ST from scratch simplifies model development, and lets us efficiently re-examine recently proposed techniques for ST, and explore novel techniques. This allows us to build strong models for future research to build on or compare to.

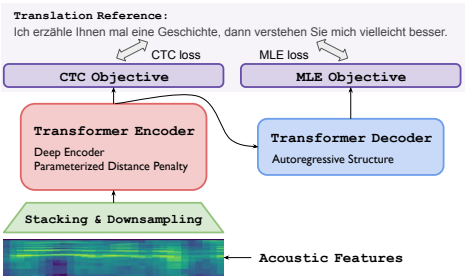


Figure 1: Overview of the proposed ST system. The example is for En-De translation. During inference, the CTC layer is dropped and only the autoregressive decoder is used.

3. Methods for ST From Scratch

We argue that the inferior performance of ST from scratch as reported in the literature is due to a lack of system adaptation with respect to training and modeling. In this section, starting with a brief overview of our baseline system, we discuss several potential directions that could strengthen E2E ST without pretraining. The overall framework of the proposed system is shown in Figure 1.

3.1. Baseline

Our baseline follows the encoder-decoder paradigm (Bahdanau et al., 2015) and uses Transformer (Vaswani et al., 2017) as its backbone. Except for (speech, translation) pairs denoted as (X, Y), respectively, we assume that there is no access to other data at training for ST from scratch.

The encoder stacks N_{enc} identical layers, each of which has a multi-head self-attention sublayer and a feed-forward sublayer. To enhance its short-range dependence modeling, we apply the logarithmic distance penalty (Di Gangi et al., 2019) to each head of its self-attention:

$$\text{Head}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{head}}} - \pi\left(\mathbf{D}\right)\right)\mathbf{V}, \quad (1)$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{|X| \times d_{head}}$ are the query, key and value inputs, respectively. d_{head} is the attention head dimension. $|\cdot|$ denotes sequence length. $\mathbf{D} \in \mathbb{R}^{|X| \times |X|}$ stores the

position distance, i.e. $D_{i,j} = |i-j|+1$, and $\pi(\cdot) = \log(\cdot)$.

Analogous to the encoder, the decoder stacks N_{dec} identical layers. We reuse the standard Transformer decoder for our baseline, and optimize all model parameters using the traditional maximum likelihood objective (MLE), or $\mathcal{L}^{\text{MLE}}$.

3.2. Hyperparameter Tuning

Hyperparameters often highly affect ST from scratch, but exhaustively searching for optimal settings is impractical. Instead, we take inspiration from past studies and re-examine several configurations that have been proven beneficial to ST with pretraining. We hypothesize that such configurations also have a high chance to generalize to ST from scratch. For example, since ST is generally a low-resource task, using smaller vocabulary (Inaguma et al., 2020), larger dropout rate (Sennrich & Zhang, 2019), reduced attention heads and model dimension (Inaguma et al., 2020; Zhao et al., 2021) might help to avoid overfitting. We also test different settings for acoustic feature extraction, deep encoder (Zhang et al., 2019) and wide feed-forward layer (Inaguma et al., 2020), apart from tuning the length penalty at inference (Wu et al., 2016).

3.3. CTC-based Regularization

CTC, or Connectionist Temporal Classification, is a latent alignment objective that models probabilistic distribution by marginalizing over all valid mappings between the input and output sequence (Graves et al., 2006). Under a strong conditional independence assumption, it can be computed efficiently and tractably via dynamic programming. We refer readers to Graves et al. (2006) for more algorithmic details. So far, CTC has been applied to non-autoregressive MT and ST successfully (Libovický & Helcl, 2018; Chuang et al., 2021) and is well supported by popular computational frameworks.

In this paper, we regard CTC as a regularizer and stack it onto the encoder for *ST modeling* as shown in Figure 1. The overall training objective becomes as below:

$$\mathcal{L}(X,Y) = (1-\lambda)\mathcal{L}^{\text{MLE}}(Y|X) + \lambda\mathcal{L}^{\text{CTC}}(Y|X), \quad (2)$$

where λ is a hyperparameter controlling the degree of the regularization. Chuang et al. (2021) showed that CTC improves the reordering tendency of the self-attention in non-autoregressive ST, although it assumes monotonicity. We expect that such reordering could reduce the learning difficulty of ST and ease the decoder's job, delivering better translation quality. One problem of applying CTC to ST is that the input speech sequence might be shorter than its translation sequence, which violates CTC's presumption. We simply ignore these samples during training. Note that the CTC layer will be abandoned after training.

3.4. Parameterized Distance Penalty

The distance penalty in Eq. 1 penalizes attention logits logarithmically with distance based on a *hard-coded* function, reaching a certain degree of balance in modeling local and global dependencies. However, such a function lacks flexibility and inevitably suffers from insufficient capacity when characterizing data-specific locality. To solve this problem, we propose parameterized distance penalty (PDP) which includes a learnable parameter for each distance. PDP is inspired by the relative position representation (Shaw et al., 2018; Raffel et al., 2020) and is formulated as below:

$$\pi^{\text{PDP}}(\mathbf{D}) = \log(\mathbf{D}) f(\mathbf{D}), \quad (3)$$

$$f(D_{i,j}) = \begin{cases} \mathbf{w}_{D_{i,j}}, & \text{if } D_{i,j} < R \\ \mathbf{w}_R, & \text{otherwise} \end{cases} \quad (4)$$

where $\mathbf{w} \in \mathbb{R}^R$ is a trainable vector, R is a hyperparameter, and $\mathbf{w}_i$ denotes its i-th element. PDP is easily parallelizable, adding little computational overhead. We initialize each $\mathbf{w}_i$ to 1 so that PDP starts from $\pi(\cdot)$ and then gradually adjusts itself during training. Besides, $\mathbf{w}$ is attention head-specific, i.e. each head has its own parameterization. By doing so, we enable different heads capturing varying degree of locality, which further increases modeling freedom.

4. Experimental Setup

Dataset We work on four benchmarks covering different domains and 23 languages from diverse language families.

MuST-C MuST-C is extracted from TED talks (Di Gangi et al., 2019), offering translations from English (En) to 8 languages: German (De), Spanish (Es), French (Fr), Italian (It), Dutch (Nl), Portuguese (Pt), Romanian (Ro) and Russian (Ru). The training sets of each language are at a similar scale, roughly 452 hours with 252K utterances on average.

LibriSpeech En-Fr The Augmented LibriSpeech dataset is collected by aligning e-books in French with English utterances of LibriSpeech (Kocabiyikoglu et al., 2018). We only use the 100 hours clean training set and its augmented references offered by Google Translate for training, totalling 94K utterances.

Kosp2e Ko-En Kosp2e is constructed from a mix of four domains (textbook, news, AI agent and diary) for Korean-to-English (Ko-En) speech translation (Cho et al., 2021). The training set has about 190 hours with 106K utterances.

CoVoST CoVoST (version 2) is a large-scale multilingual ST corpus collected from Common Voice (Ardila et al., 2020), providing translations from En to 15 languages

Table 1: Ablation results on MuST-C En-De test set. *#Params*: number of model parameters. *BLEU*: higher is better, SacreBLEU. Numbers in bold denote top scores.

ID	System	#Params	BLEU↑
1	Baseline	51M	18.1
Tune beam search, dropout and batch size			
2	1 + adjust length penalty at inference	51M	18.8
3	1 + higher dropout (0.2→0.4)	51M	17.4
4	1 + apply dropout to raw waveform signals (rate 0.1)	51M	14.6
5	1 + reduce batch size by half	51M	17.6
Tune model dimension and depth			
6	2 + reduce model dimension and attention heads ($H: 8 \rightarrow 4, d_{model}: 512 \rightarrow 256$)	20M	19.0
7	6 + enlarge feed-forward layer ($d_{ff}: 2048 \rightarrow 4096$)	33M	19.3
8	6 + enlarge encoder depth with DS-Init ($N_{enc}: 6 \rightarrow 12$)	28M	20.4
9	8 + enlarge feed-forward layer ($N_{enc} = 12, d_{ff}: 2048 \rightarrow 4096$)	47M	21.1
10	2 + enlarge encoder depth with DS-Init ($N_{enc}: 6 \rightarrow 12$)	70M	20.3
Add parameterized distance penalty (PDP)			
11	2 + PDP ($R = 512$)	51M	19.5
12	11 + initialize $\mathbf{w}$ in PDP randomly	51M	18.3
13	11 + use 80-dimensional log mel-scale filterbank ($F: 40 \rightarrow 80$)	51M	19.3
14	11 + remove delta and delta-delta features ($d_{speech}: 120 \rightarrow 40$)	50M	18.8
Tune vocabulary size and LN			
15	9 + PDP	47M	21.8
16	15 + small BPE vocabulary ($V: 16K \rightarrow 8K$)	46M	21.8
17	16 + change post-LN to pre-LN	46M	20.6
Final system: add CTC			
18	16 + CTC regularization ($\lambda = 0.3$) (also, the proposed system)	48M	**22.7**
19	for comparison: 16 + ASR pretraining	46M	**22.9**
20	for comparison: 1 + ASR pretraining	51M	20.7

– Arabic (Ar), Catalan (Ca), Welsh (Cy), De, Estonian (Et), Persian (Fa), Indonesian (Id), Japanese (Ja), Latvian (Lv), Mongolian (Mn), Slovenian (Sl), Swedish (Sv), Tamil (Ta), Turkish (Tr), Chinese (Zh) – and from 21 languages to En, including the 15 target languages as well as Es, Fr, It, Nl, Pt and Ru (Wang et al., 2020b). The training set for En→Xx translation is of similar scale, roughly 427 hours with 289K utterances. In contrast, the training data size for Xx→En translation varies greatly, from about 1.2 hours/1.2K utterances (Id) to 263 hours/207K utterances (Fr). We mainly work on Fr, De, Es, Ca, It, Ru, and Zh for Xx→En.

For each benchmark, we use the official train/dev/test split for experiments. We convert all audios to a sampling rate of 16KHz and truncate segments to 3000 frames. We extract 40-dimensional log mel-scale filterbank features ($F = 40$) with a step size of 10ms and window size of 25ms, which are then expanded with their delta and delta-delta features followed by mean subtraction and variance normalization, resulting in the final 120-dimensional acoustic features ($d_{speech} = 120$). We tokenize and truecase all texts via Moses (Zh and Ja excluded) (Koehn et al., 2007), and handle infrequent words via subword models (Sennrich et al., 2016; Kudo & Richardson, 2018) with a vocabulary size of 16K ($V = 16K$).

Model Setting On top of the acoustic input, we concatenate three consecutive frames without overlapping as a way of downsampling (Zhang et al., 2020), as in Figure 1. We then add a linear layer to get the encoder input of dimension d_{model}. We use the sinusoidal encoding to distinguish different positions, and employ the post-LN (layer normalization) structure for Transformer (Vaswani et al., 2017).

Regarding Baseline, we set $d_{model} = 512$, $d_{head} = 64$, the number of attention head $H = 8$, the feed-forward layer size $d_{ff} = 2048$ and $N_{enc} = N_{dec} = 6$. Note $d_{model} = H \cdot d_{head}$. By default, we set $R = 512$ and $\lambda = 0.3$.

We employ Adam (Kingma & Ba, 2015, $\beta_1 = 0.9, \beta_2 = 0.98$) for parameter update using adaptive learning rate schedule as in (Vaswani et al., 2017) with a warmup step of 4K and label smoothing of 0.1. Dropout of rate 0.2 is applied to residual connections and ReLU activations. We organize training samples of around 20K target subwords into one batch, and train models up to 50K steps.

Evaluation We average the best 10 checkpoints according to dev set performance for evaluation. For decoding, we adopt beam search, where we set the beam size and length

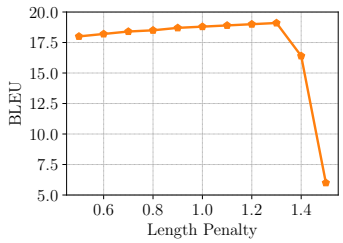


Figure 2: Dev SacreBLEU scores as a function of length penalty ($0.5 \rightarrow 1.5$) for Baseline on MuST-C En-De. Trade-off exists.

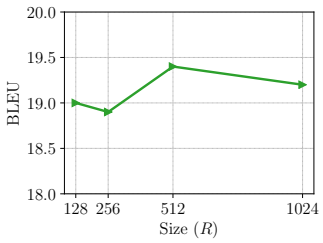


Figure 3: Dev SacreBLEU on MuST-C En-De when changing R in PDP for system 11. Setting $R = 512$ yields the best result.

penalty to 8 and 0.6, respectively. We will examine the impact of the length penalty on translation later. Unless otherwise stated, we measure translation quality with detokenized case-sensitive BLEU (Papineni et al., 2002) offered by SacreBLEU (Post, 2018).[1] Note that we did not perform any filtering to the test set at evaluation time.

5. Results and Analysis

We test different hyperparameters and our proposals mainly on MuST-C En-De. Table 1 summarizes the results.

Apart from architecture, length penalty in beam search also matters. Length penalty is used to bias beam search generating longer or shorter outputs, which often largely affects translation quality as shown in Figure 2.[2] Tuning this setting alone results in +0.7 BLEU gains ($1 \rightarrow 2$).

Applying more dropout and smaller batch size helps little. Dropout is a popular regularizer to avoid overfitting. We tried using larger dropout rate and adding dropout to raw waveforms, but ended up with significantly slower convergence and worse performance ($1 \rightarrow 3, 4$). Also, reducing training batch deteriorates ST ($1 \rightarrow 5$).

Deepening speech encoder, widening feed-forward layer, and reducing model dimension benefit ST. Halving model dimension greatly reduces the number of model parameters but still retains translation quality ($2 \rightarrow 6$). Enlarging the encoder depth (from 6 to 12) and the feed-forward dimension (from 2048 to 4096) leads to substantial quality improvement, +2.1 BLEU ($6 \rightarrow 9$). After varying dimensions, we could achieve a BLEU score of **21.1**. Note, we employed the depth-scaled initialization to smooth model gradients for deep Transformer (Zhang et al., 2019, DS-Init) and set $\alpha = 0.5$. Besides, deep speech encoder improves ST from scratch with the Baseline dimensions ($2 \rightarrow 10$).

The proposed parameterized distance penalty improves ST. The hyperparameter R in Eq. 3 affects the flexibility of PDP in modeling local context. Figure 3 shows its impact on ST. In general, setting $R = 512$ achieves good performance. Note, its optimal setting might (and is likely to) be dataset-dependent.

Applying PDP to ST gains BLEU ($2 \rightarrow 11$) and is complementary to model dimension manipulation ($9 \rightarrow 15$), reaching a test BLEU score of 21.8. We also tested the effectiveness of initializing all $\mathbf{w}_i$ to 1. Using the vanilla random initialization instead delivers inferior quality, -1.2 BLEU ($11 \rightarrow 12$).

Inadequate acoustic feature extraction hurts ST. In previous ST systems (Inaguma et al., 2020; Zhao et al., 2021), acoustic feature extraction often uses 80-dimensional filterbanks without delta and delta-delta features. We checked this in our setup. Using more filterbanks does not help much ($11 \rightarrow 13$), and delta features benefit ST a lot ($11 \rightarrow 14$).

Reducing vocabulary size affects En-De translation little. Previous studies also suggest to use smaller vocabularies in low-resource settings (Karita et al., 2019; Sennrich & Zhang, 2019). Reducing vocabulary size by half yields little impact on En-De translation ($15 \rightarrow 16$). We adopt smaller vocabularies due to three reasons: 1) it reduces the number of parameters; 2) we observed that it has much greater influence on other languages; and 3) CTC with smaller vocabulary is more computationally efficient.

Post-LN vs. Pre-LN Another way to train deep Transformer is to use the pre-LN structure (Wang et al., 2019b). It has been shown that the post-LN, once successfully optimized, often outperforms its pre-LN counterpart (Zhang et al., 2019). We reconfirmed this observation, and found that the post-LN ST with DS-Init shows clear superiority in performance, +1.2 BLEU ($17 \rightarrow 16$).

[1] Signature: *BLEU+c.md+#ref.1+s.exp+tok.13a+v.1.4.14*

[2] Note that its impact is dataset-dependent. On CoVoST, BLEU changes little when varying it.

Table 2: Results of different systems on MuST-C tst-COMMON. *Avg*: average score over different languages. †: systems that might perform filtering to the test set, so comparison could be unfair. ‡: systems using large-scale external ASR and/or MT data.

System	Aux. Data		De	Es	Fr	It	Nl	Pt	Ro	Ru	Avg
	ASR	MT									
Adapted Transformer (Di Gangi et al., 2019)	✓		17.3	20.8	26.9	16.8	18.8	20.1	16.5	10.5	18.5
ESPnet-ST (Inaguma et al., 2020)†	✓	✓	22.9	28.0	32.8	23.8	27.4	28.0	21.9	15.8	**25.1**
AFS (Zhang et al., 2020)	✓		22.4	26.9	31.6	23.0	24.9	26.3	21.0	14.7	23.9
Contextual Modeling (Zhang et al., 2021)	✓		22.9	27.3	32.5	23.1	26.0	27.1	23.6	15.8	24.8
Fairseq-ST (Wang et al., 2020a)†	✓		22.7	27.2	32.9	22.7	27.3	28.1	21.9	15.3	24.8
NeurST (Zhao et al., 2021)	✓		22.8	27.4	33.3	22.9	27.2	28.7	22.2	15.1	24.9
E2E-ST-JT (Du et al., 2021)†	✓		23.1	27.5	32.8	23.6	27.8	28.7	22.1	14.9	**25.1**
Chimera (Han et al., 2021)‡	✓	✓	27.1	30.6	35.6	25.0	29.2	30.2	24.0	17.4	***27.4***
our system			22.7	28.1	33.4	23.2	26.9	28.3	22.6	15.4	**25.1**
our system + neural acoustic feature modeling			23.0	28.0	33.5	23.5	27.1	28.2	23.0	15.6	**25.2**

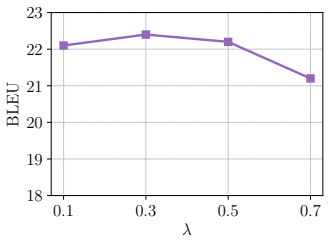


Figure 4: Dev SacreBLEU as a function of λ on MuST-C En-De for system 18. We set $\lambda = 0.3$ in our experiments.

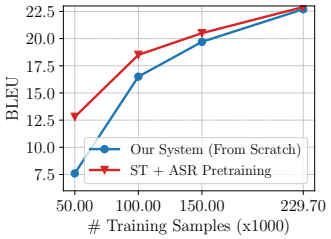


Figure 5: Impact of the amount of training data on MuST-C En-De translation. Results are for test SacreBLEU.

CTC greatly improves ST from scratch. Finally, we integrate the CTC regularization into our best system. The hyperparameter λ in Eq. 2 controls the trade-off between two different objectives. Figure 4 shows that λ directly affects ST, and setting $\lambda = 0.3$ achieves the best result. Under this setting, CTC benefits ST with another significant quality gain, +0.9 BLEU, reaching a test BLEU score of **22.7** (18).

From Baseline to system 18, we improve ST by 4.6 BLEU. Note that this system also outperforms the baseline with ASR pretraining (20), and that the gap between our best system trained from scratch and its pretrained counterpart has become very narrow (18 vs. 19). For all follow-up experiments, we use system 18 as our proposed system.

Pretraining matters in low-resource regime. Pretraining might not be a necessity when rich training data is given, but it matters as the amount of training data decreases. Figure 5 demonstrates this. ASR pretraining helps low-resource ST.

Results On Other Languages Putting all together, we obtain a set of best practices, involving $N_{enc} = 12, N_{dec} = 6, d_{model} = 256, H = 4, d_{ff} = 4096, V = 8K$, using PDP with $R = 512$ and applying CTC with $\lambda = 0.3$. We then keep this configuration and train models for other language pairs. Tables 2-5 list the results.

Our revisiting of ST from scratch shows that its performance gap to ST with pretraining has generally been overestimated in the literature. This gap can be largely reduced and even fully closed after biasing E2E ST towards training from scratch. Our system achieves an average BLEU of 25.1 and 17.3 on MuST-C and CoVoST En→Xx, respectively, which surpasses many popular neural systems, such as the ones supported by Fairseq (Wang et al., 2020a) and NeurST (Zhao et al., 2021). Similarly, our system achieves very promising performance on LibriSpeech En-Fr and Kosp2e Ko-En, delivering 18.9 and 5.8 BLEU, respectively. Note Cho et al. (2021) employed extra large-scale ASR data for pretraining, which is merely 0.1 BLEU higher than ours. While this is beyond the scope of our work, our results suggest that it is worthwhile to revisit large-scale pretraining based on our stronger baseline, which will lead to either new state-of-the-art results or a re-evaluation of the effectiveness of large-scale pretraining.

Table 3: Results of different systems for En→Xx and Xx→En on CoVoST. We report character-level BLEU for Chinese and Japanese following Wang et al. (2020b). Languages underlined have training data fewer than 100K samples.

System	Aux. Data		Xx→En							
	ASR	MT	Fr	De	Es	Ca	It	Ru	Zh	Avg
ST from scratch (Wang et al., 2020b)			24.3	8.4	12.0	14.4	0.2	1.2	1.4	8.8
ST + ASR Pretraining (Wang et al., 2020b)	✓		26.3	17.1	23.0	18.8	11.3	14.8	5.8	**16.7**
our system			26.9	14.1	15.7	17.2	2.4	3.6	2.0	11.7

En→Xx															
Ar	Ca	Cy	De	Et	Fa	Id	Ja	Lv	Mn	Sl	Sv	Ta	Tr	Zh	Avg
8.7	20.2	22.2	13.6	11.1	11.5	18.9	26.9	11.5	6.6	11.5	20.1	9.9	8.9	20.6	14.8
12.1	21.8	23.9	16.3	13.2	13.1	20.4	29.6	13.0	9.2	16.0	21.8	10.9	10.0	25.4	17.1
12.3	22.9	24.5	17.5	13.6	12.7	21.4	28.8	13.6	9.9	15.2	22.9	10.8	10.3	23.3	**17.3**

Table 4: Results of different systems on LibriSpeech En-Fr test set. For comparison to previous work, we report both case-insensitive tokenized BLEU (tok) and SacreBLEU.

System	Aux. Data		BLEU	
	ASR	MT	tok	Sacre
ST + KD (Liu et al., 2019)		✓	17.02	
TCEN (Wang et al., 2019a)	✓	✓	17.05	
AFS (Zhang et al., 2020)	✓		18.56	
LUT (Dong et al., 2021)	✓	✓	18.34	
Chimera (Han et al., 2021)‡	✓	✓		***19.4***
our system			**18.90**	16.5

Table 5: Results of different systems on Kosp2e Ko-En test set.

System	Aux. Data		BLEU
	ASR	MT	
ST from scratch (Cho et al., 2021)			2.6
ST + pretraining (Cho et al., 2021)‡	✓		**5.9**
our system			**5.8**

Our results also show that pretraining matters mainly in two aspects: 1) low-resource scenarios, where our system still lags far behind pretraining-enhanced ST, -5.0 BLEU on CoVoST Xx→En in Table 3; and 2) large-scale ASR and/or MT data is available, where pretraining or joint modeling can largely improve ST, +2.3 BLEU on MuST-C in Table 2 yielded by Chimera (Han et al., 2021).

Notice that our system should be regarded as a lower-bound for ST from scratch, since many outstanding optimization techniques for E2E ST, e.g. SpecAugment (Park et al., 2019), are not considered here due to resource limitations. In addition, we did not aggressively optimize our system towards very low-resource scenarios, so there should still be room for quality improvement on CoVoSt Xx→En. Also note that comparison to ST models powered by ESPnet (Inaguma et al., 2020) and Fairseq (Wang et al., 2020a) might not be fair because both toolkits perform data filtering to the test set, although SacreBLEU is also used.

6. Neural Acoustic Feature Modeling

A general trend in deep learning is to replace handcrafted features with neural networks to let the model automatically capture or learn the underlying pattern behind data. In E2E ST, one heuristic is the adoption of log mel-scale filterbanks for acoustic modeling. Despite its success, filterbank-based modeling prevents us from accessing full acoustic details and its transformation might suffer from information loss (Lam et al., 2021), making it sub-optimal for ST. Inspired by recent speech studies on modeling raw waveforms (Lam et al., 2021), we propose neural acoustic feature modeling (NAFM) to remove such heuristic and increase the freedom of E2E ST in describing speech.

The extraction of filterbanks often involves a sequence of two specifically designed linear transformations. To simulate such structure, we employ two feed-forward neural blocks for NAFM as follows:

$$\mathbf{x}^{(1)} = \mathrm{LN}\left(\mathrm{FFN}\left(\mathbf{x}^{(0)}\right) + \mathbf{x}^{(0)}\right), \quad (5)$$

$$\mathbf{x}^{(2)} = \mathrm{LN}\left(\mathrm{FFN}\left(\mathbf{x}^{(1)}\right) + \mathbf{x}^{(1)}\right), \quad (6)$$

where $\mathbf{x}^{(0)} \in \mathbb{R}^{d_{speech}}$ is the raw speech frame, and $\mathrm{FFN}(\cdot)$ is the feed-forward layer as in Transformer (Vaswani et al., 2017) with $d_{ff} = 4096$. We expect that, by adding trainable parameters tuned with translation losses, NAFM could induce ST-oriented acoustic features that improves ST.

However, directly using $\mathbf{x}^{(2)}$ as an alternative to the filterbank features $\mathbf{x}^{f}$ results in poor convergence. We argue that filterbanks offer helpful inductive biases to ST, and propose to leverage such information to regularize NAFM. Formally,

Table 6: Results of applying NAFM to ST on MuST-C En-De.

System	# Params	BLEU
our system	48M	22.7
our system + NAFM	54M	**23.0**
our system + two FFN blocks alone	54M	22.7

we add the following L_2 objective into training:

$$\mathcal{L}^{\text{NAFM}}(X,Y) = \mathcal{L}(X,Y) + \gamma \frac{1}{|X|}(\|\mathbf{X}^{(2)} - \mathbf{X}^{f}\|^2), \quad (7)$$

where γ is a hyperparameter and set to 0.05 in experiments.

Results in Table 6 show that training E2E ST from scratch on raw waveforms is feasible. NAFM improves ST by 0.3 BLEU on MuST-C En-De, and such improvement is not a trivial result of simply adding parameters. The last row of Table 2 shows the effectiveness of NAFM on other languages. Overall, the performance of NAFM matches and even outperforms its filterbank-based counterpart across different languages. Although NAFM does not deliver significant gains, we believe that optimizing ST with raw waveforms has great potential and deserves more effort.

7. Related Work

Methods to improve E2E ST are many. Apart from developing novel model architectures (Di Gangi et al., 2019; Karita et al., 2019; Zhang et al., 2020), one promising way is to leverage knowledge transfer from auxiliary tasks. Multilingual or cross-lingual ST improves translation by adding translation supervisions from other languages (Inaguma et al., 2019; Bansal et al., 2019; Liu et al., 2019). Multi-task learning benefits ST by jointly modeling ASR and ST tasks within a single model (Anastasopoulos & Chiang, 2018; Zheng et al., 2021; Dong et al., 2021). Pretraining methods, including large-scale self-supervised pretraining (Schneider et al., 2019) and ASR/MT-based supervised pretraining, offer a warm-up initialization for E2E ST to improve its data efficiency (Le et al., 2021; Salesky et al., 2019; Xu et al., 2021). However, all these studies assume that (bilingual) ST from scratch is poor, while spending little effort on optimizing it. We challenge this assumption and demonstrate that ST from scratch can also yield decent performance.

We adopt the CTC objective as a regularizer to improve E2E ST. CTC was proposed for ASR tasks to handle the latent alignment between speech and transcript (Graves et al., 2006), which has been widely used to train ASR models, including ASR pretraining for ST. It also contributes to non-autoregressive translation. Libovický & Helcl (2018) and Saharia et al. (2020) applied the CTC loss to non-autoregressive MT and obtained improved translation performance. Gu & Kong (2021) observed that CTC is essential to achieve fully or one-step non-autoregressive MT. In addition, Chuang et al. (2021) showed that CTC enhances the reordering behavior of non-autoregressive ST. Different from these studies, we apply CTC to improve autoregressive ST, although Haviv et al. (2021) showed that CTC helps autoregressive MT little.

There are several pioneering studies trying to relax the heuristics in acoustic features to improve speech representation. Sainath et al. (2013) and Seki et al. (2017) explored a neural filter bank layers as an alternative to the hand-engineered filterbanks. Hoshen et al. (2015) proposed a convolutional neural acoustic model that operates directly on raw waveforms, aiming at capturing the fine-grained time structure. Lam et al. (2021) further proposed a globally attentive locally recurrent network, gaining quality and robustness for ASR. These studies mainly focus on ASR. To the best of our knowledge, applying NAFM to ST has never been investigated before, and we demonstrated its feasibility.

8. Conclusion and Discussion

Is pretraining a necessity to E2E ST? We answer this question by reexamining several techniques and devising two novel proposals, namely parameterized distance penalty (PDP) and neural acoustic feature modeling (NAFM), for ST from scratch. Via extensive experiments, we present a set of best practices for ST from scratch, including smaller vocabulary, deep post-LN encoder, wider feed-forward layer, ST-based CTC regularization and PDP. We show that ST models trained from scratch, when properly optimized, can match and even outperform previous work relying on pretraining, thus challenging its necessity.

Our study does not preclude pretraining for ST. Instead, we provide an improved understanding of its role on E2E ST. Our results show that pretraining matters mainly in two settings: (extremely) low-resource setup and scenarios where large-scale external ASR and MT data is available. The performance gap in such settings remains. From our perspective, how to leverage other types of data to improve pretraining for ST is a promising yet challenging research topic. We invite researchers to build upon our models to re-examine the importance of pretraining in various settings.

In addition, we examined and demonstrated the feasibility of performing E2E ST on raw waveforms through NAFM. Although we did not obtain consistent and substantial quality gains, NAFM still has the potential of fully leveraging all acoustic signals and yielding improved acoustic features for ST, achieving better results with more suitable architectures.

References

Anastasopoulos, A. and Chiang, D. Tied multitask learning for neural speech translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 82–91, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1008. URL https://www.aclweb.org/anthology/N18-1008.

Anastasopoulos, A., Bojar, O., Bremerman, J., Cattoni, R., Elbayad, M., Federico, M., Ma, X., Nakamura, S., Negri, M., Niehues, J., Pino, J., Salesky, E., Stüker, S., Sudoh, K., Turchi, M., Waibel, A., Wang, C., and Wiesner, M. FINDINGS OF THE IWSLT 2021 EVALUATION CAMPAIGN. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pp. 1–29, Bangkok, Thailand (online), August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.iwslt-1.1. URL https://aclanthology.org/2021.iwslt-1.1.

Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F., and Weber, G. Common voice: A massively-multilingual speech corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 4218–4222, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.520.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Bansal, S., Kamper, H., Livescu, K., Lopez, A., and Goldwater, S. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. 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)*, pp. 58–68, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1006. URL https://www.aclweb.org/anthology/N19-1006.

Bérard, A., Pietquin, O., Servan, C., and Besacier, L. Listen and translate: A proof of concept for end-to-end speech-to-text translation. In *NIPS Workshop on End-to-end Learning for Speech and Audio Processing*, Barcelona, Spain, 2016.

Chen, J., Ma, M., Zheng, R., and Huang, L. Mam: Masked acoustic modeling for end-to-end speech-to-text translation. *arXiv preprint arXiv:2010.11445*, 2020.

Cho, W. I., Kim, S. M., Cho, H., and Kim, N. S. kosp2e: Korean Speech to English Translation Corpus. In *Proc. Interspeech 2021*, pp. 3705–3709, 2021. doi: 10.21437/Interspeech.2021-1040.

Chuang, S.-P., Chuang, Y.-S., Chang, C.-C., and Lee, H.-y. Investigating the reordering capability in CTC-based non-autoregressive end-to-end speech translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1068–1077, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.92. URL https://aclanthology.org/2021.findings-acl.92.

Di Gangi, M. A., Cattoni, R., Bentivogli, L., Negri, M., and Turchi, M. MuST-C: a Multilingual Speech Translation Corpus. 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)*, pp. 2012–2017, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1202. URL https://www.aclweb.org/anthology/N19-1202.

Di Gangi, M. A., Negri, M., and Turchi, M. Adapting Transformer to End-to-End Spoken Language Translation. In *Proc. Interspeech 2019*, pp. 1133–1137, 2019. doi: 10.21437/Interspeech.2019-3045. URL http://dx.doi.org/10.21437/Interspeech.2019-3045.

Dong, Q., Ye, R., Wang, M., Zhou, H., Xu, S., Xu, B., and Li, L. Listen, understand and translate: Triple supervision decouples end-to-end speech-to-text translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12749–12759, May 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17509.

Du, Y., Zhang, Z., Wang, W., Chen, B., Xie, J., and Xu, T. Regularizing end-to-end speech translation with triangular decomposition agreement. *arXiv preprint arXiv:2112.10991*, 2021.

Duong, L., Anastasopoulos, A., Chiang, D., Bird, S., and Cohn, T. An attentional model for speech translation without transcription. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 949–959, San Diego, California, June 2016. Association for Computational Lin-

guistics. doi: 10.18653/v1/N16-1109. URL https://www.aclweb.org/anthology/N16-1109.

Graves, A., Fernández, S., and Gomez, F. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *In Proceedings of the International Conference on Machine Learning, ICML 2006*, pp. 369–376, 2006.

Gu, J. and Kong, X. Fully non-autoregressive neural machine translation: Tricks of the trade. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 120–133, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.11. URL https://aclanthology.org/2021.findings-acl.11.

Han, C., Wang, M., Ji, H., and Li, L. Learning shared semantic space for speech-to-text translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2214–2225, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.195. URL https://aclanthology.org/2021.findings-acl.195.

Haviv, A., Vassertail, L., and Levy, O. Can latent alignments improve autoregressive machine translation? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2637–2641, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.209. URL https://aclanthology.org/2021.naacl-main.209.

Hoshen, Y., Weiss, R. J., and Wilson, K. W. Speech acoustic modeling from raw multichannel waveforms. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4624–4628, 2015. doi: 10.1109/ICASSP.2015.7178847.

Inaguma, H., Duh, K., Kawahara, T., and Watanabe, S. Multilingual end-to-end speech translation. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 570–577. IEEE, 2019.

Inaguma, H., Kiyono, S., Duh, K., Karita, S., Yalta, N., Hayashi, T., and Watanabe, S. ESPnet-ST: All-in-one speech translation toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 302–311, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.34. URL https://aclanthology.org/2020.acl-demos.34.

Karita, S., Chen, N., Hayashi, T., Hori, T., Inaguma, H., Jiang, Z., Someki, M., Soplin, N. E. Y., Yamamoto, R., Wang, X., Watanabe, S., Yoshimura, T., and Zhang, W. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 449–456, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kocabiyikoglu, A. C., Besacier, L., and Kraif, O. Augmenting librispeech with French translations: A multimodal corpus for direct speech translation evaluation. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://www.aclweb.org/anthology/L18-1001.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pp. 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P07-2045.

Kudo, T. and Richardson, J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012.

Lam, M. W., Wang, J., Weng, C., Su, D., and Yu, D. Raw Waveform Encoder with Multi-Scale Globally Attentive Locally Recurrent Networks for End-to-End Speech Recognition. In *Proc. Interspeech 2021*, pp. 316–320, 2021. doi: 10.21437/Interspeech.2021-2084.

Le, H., Pino, J., Wang, C., Gu, J., Schwab, D., and Besacier, L. Lightweight adapter tuning for multilingual speech translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 817–824, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-short.103. URL https://aclanthology.org/2021.acl-short.103.

Libovický, J. and Helcl, J. End-to-end non-autoregressive neural machine translation with connectionist tempo-

ral classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3016–3021, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1336. URL https://aclanthology.org/D18-1336.

Liu, Y., Xiong, H., Zhang, J., He, Z., Wu, H., Wang, H., and Zong, C. End-to-End Speech Translation with Knowledge Distillation. In *Proc. Interspeech 2019*, pp. 1128–1132, 2019. doi: 10.21437/Interspeech.2019-2582. URL http://dx.doi.org/10.21437/Interspeech.2019-2582.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. Specaugment: A simple data augmentation method for automatic speech recognition. 2019.

Post, M. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6319.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Saharia, C., Chan, W., Saxena, S., and Norouzi, M. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1098–1108, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.83. URL https://aclanthology.org/2020.emnlp-main.83.

Sainath, T. N., Kingsbury, B., Mohamed, A.-r., and Ramabhadran, B. Learning filter banks within a deep neural network framework. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 297–302, 2013. doi: 10.1109/ASRU.2013.6707746.

Salesky, E., Sperber, M., and Black, A. W. Exploring phoneme-level speech representations for end-to-end speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1835–1841, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1179. URL https://www.aclweb.org/anthology/P19-1179.

Schneider, S., Baevski, A., Collobert, R., and Auli, M. wav2vec: Unsupervised pre-training for speech recognition. Apr 2019. doi: http://doi.org/10.21437/Interspeech.2019-1873. URL https://arxiv.org/abs/1904.05862.

Seki, H., Yamamoto, K., and Nakagawa, S. A deep neural network integrated with filterbank learning for speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5480–5484, 2017. doi: 10.1109/ICASSP.2017.7953204.

Sennrich, R. and Zhang, B. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 211–221, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1021. URL https://aclanthology.org/P19-1021.

Sennrich, R., Haddow, B., and Birch, A. 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)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://www.aclweb.org/anthology/P16-1162.

Shaw, P., Uszkoreit, J., and Vaswani, A. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 464–468, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2074. URL https://aclanthology.org/N18-2074.

Sung, T.-W., Liu, J.-Y., Lee, H.-y., and Lee, L.-s. Towards end-to-end speech-to-text translation with two-pass decoding. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7175–7179. IEEE, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio,

S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Wang, C., Wu, Y., Liu, S., Yang, Z., and Zhou, M. Bridging the gap between pre-training and fine-tuning for end-to-end speech translation. *arXiv preprint arXiv:1909.07575*, 2019a.

Wang, C., Tang, Y., Ma, X., Wu, A., Okhonko, D., and Pino, J. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 33–39, Suzhou, China, December 2020a. Association for Computational Linguistics. URL https://aclanthology.org/2020.aacl-demo.6.

Wang, C., Wu, A., and Pino, J. Covost 2 and massively multilingual speech-to-text translation. *arXiv preprint arXiv:2007.10310*, 2020b.

Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D. F., and Chao, L. S. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1810–1822, Florence, Italy, July 2019b. Association for Computational Linguistics. doi: 10.18653/v1/P19-1176. URL https://aclanthology.org/P19-1176.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Łukasz Kaiser, Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL http://arxiv.org/abs/1609.08144.

Xu, C., Hu, B., Li, Y., Zhang, Y., Huang, S., Ju, Q., Xiao, T., and Zhu, J. Stacked acoustic-and-textual encoding: Integrating the pre-trained models into speech translation encoders. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 2619–2630, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.204. URL https://aclanthology.org/2021.acl-long.204.

Zhang, B., Titov, I., and Sennrich, R. Improving deep transformer with depth-scaled initialization and merged attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 898–909, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1083. URL https://aclanthology.org/D19-1083.

Zhang, B., Titov, I., Haddow, B., and Sennrich, R. Adaptive feature selection for end-to-end speech translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 2533–2544, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.230. URL https://aclanthology.org/2020.findings-emnlp.230.

Zhang, B., Titov, I., Haddow, B., and Sennrich, R. Beyond sentence-level end-to-end speech translation: Context helps. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 2566–2578, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.200. URL https://aclanthology.org/2021.acl-long.200.

Zhao, C., Wang, M., Dong, Q., Ye, R., and Li, L. NeurST: Neural speech translation toolkit. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pp. 55–62, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-demo.7. URL https://aclanthology.org/2021.acl-demo.7.

Zheng, R., Chen, J., Ma, M., and Huang, L. Fused acoustic and text encoding for multimodal bilingual pretraining and speech translation. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12736–12746. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/zheng21a.html.