The effect of extra-sentential context on Speech to Discrete Unit Translation

B235864

7369 words

Master of Science

Speech and Language Processing

School of Philosophy, Psychology and Language Sciences

University of Edinburgh

2024

Abstract

Contextual information, possibly from previous sentences, is well known to be useful during translation, for resolving ambiguity or translating obscure words. This has been successfully exploited by textual machine translation, and similar success has recently been found in direct speech-to-text translation. More end-to-end speech-to-speech translation is yet to take advantage of these extra-sentential context clues, and it is not clear how well existing methods might apply. This work shows that extra-sentential context can be usefully incorporated into direct speech-to-speech translation, using a method called speech-to-unit translation (S2UT), and a crucial tool for reducing redundancy in the speech input, Adaptive Feature Selection. Results show that S2UT is sensitive to training data choices and parameter choices, but despite that sensitivity, incorporating extra-sentential context yields improvements of up to +5.4 Bleu score over the S2UT baseline.

Acknowledgements

I would like to give thanks to my supervisor Eva Colley, and the rest of the team at CSLT for their guidance and expertise on the topic of speech translation. I would like to offer enormous thanks to the coding help staff at Edinburgh University for helping with some challenging technical issues with FAIRSEQ, and for patiently listening to me explain the details of my project. Finally, I want to thank my family and friends for helping me through this project, and reminding me that there is definitely more to life than Speech to Speech translation.

Table of Contents

Chapter 1

Introduction

It has long been understood by linguists and human translators that utterances rarely exist in a vacuum. For languages with agreement, where a verb must agree with its subject, or pronoun declination must respect its referent, these agreement bonds often extend beyond a single sentence. Even in English, a language with little inflectional morphology we often rely on extra-sentential context to resolve ambgituity, or when using pronouns:

(1) A: I'm in the bank, where are you?
B: *Oh I'm right outside them.
B: Oh I'm right outside it.

Often information needed to translate a pronoun or lexically ambiguous word may be found in previous sentences, or indeed much earlier in a dialogue, as such, translation is a "document-level problem". It has been shown in the well-established field of Machine Translation (MT) that incorporating document-level context into the translation task improves translation quality, and helps to resolve various ambiguities (Maruf et al., 2019). Much time and computational resources have been dedicated to crafting dense representations of words and subwords, along with architectures that can attend over wide contexts, and we are now approaching the stage where machine translation systems can competently attend over information at the document-level.

This is not the case for the relatively burgeoning field of Speech-to-Speech Translation (ST). A sequence of features derived for a word of speech is much longer than the equivalent textual representation, containing information that may not always be relevant to the translation task, such as background noise or speaker characteristics. Recorded speech for a document of text will extend well outside the context window

any current MT model is capable of attending over, presenting a computational bottleneck. If a document-level approach is to be adopted in speech-to-speech translation, new architectures or data manipulation techniques must be considered.

This work introduces several state-of-the-art architectures used for ST, and after providing relevant motivation, adopts Speech-to-Unit translation (S2UT) an end-to-end (E2E) approach to ST that translates directly from source audio into discrete units learnt from the paired target audio (Lee et al., 2021). To solve the computational bottleneck, allowing S2UT to tackle document-level ST, a method called Adaptive Feature Selection (AFS) is borrowed from existing research on Speech-to-text translation (S2T) (Zhang et al., 2020), where speech features are pruned away if they are predicted to be redundant for the downstream task of ST. I implement S2UT following Lee et al. (2021) as a baseline model, then train and incorporate AFS, before finally tackling document-level S2UT, using a concatenative approach pioneered by Tiedemann and Scherrer (2017). The main contributions of this work are the following:

1. Replicating the S2UT model proposed by Lee et al. (2021), and providing an in-depth analysis of the complexity of S2UT.

2. Adapting AFS for use in E2E ST models, with mixed results across sentence and document translation tasks.

3. Implementing concatenative-ST with the S2UT paradigm, giving multiple sources of evidence that S2UT learns to use contextual information to improve translation quality, yielding an improvement of up to +5.4 Bleu Score over the sentence-level S2UT baseline.

Chapter 2

Background

2.1 Traditional approach: cascaded models

Traditionally speech-to-speech translation was handled with a cascaded model. The task was broken into three sub-tasks: Automatic Speech Recognition (ASR) to turn the source speech into source text, MT to translate source text into target text, and Text-To-Speech (TTS) to synthesise target text into target speech. This is simple to understand, and importantly, still allows us to tackle the translation problem at a document level since translation is from text to text, as before. Performing ST in three disjointed tasks also allows us to make use of plentiful training data and strong pre-trained models for each sub-task, and as such, translation quality is often high for cascaded models.

But using separate components for ST has some drawbacks. Cascaded models are often very computationally heavy, since at each step in the pipeline data must be translated in and out of the format that each component requires. Since each component is also trained with its own loss, the model cannot be trained together, so errors occurring early in the pipeline may be propagated forward, which cannot easily be adjusted for by the other components.

Another significant weakness inherent to traditional cascaded models is that they cannot be extended to unwritten languages, since the translation step is performed with text-derived representations. This is an issue that E2E ST has the potential to solve, but to do this, information found in the speech representations must be used for translation directly. Other information found in speech representations can also be used in the translation process, such as paralinguistic information, prosody, emotion, or other speaker characteristics. Currently this information is lost during the ASR step when the source text is discretised into words.

2.2 End-to-End speech translation

Recently, end-to-end (E2E) approaches to ST have been proposed that help to streamline the task, and have been shown to alleviate some of the issues that cascaded models present. Speech to Text translation (S2T) replaces the ASR and MT components with one E2E model that takes source speech as input and predicts words in the target language. Like cascaded models, this target text is then converted to speech by a TTS component. This can remove some error propagation that would otherwise occur between the ASR and MT components. S2T models have also been shown to provide strong translation performance, with some models achieving parity with cascaded models (Weiss et al., 2017). But many drawbacks of the cascaded model approach are inherited by S2T. Computational cost, although less than an equivalent cascaded model, is still very high, with large memory usage, long training times and high inference latency. It has been shown that the TTS component is the bottleneck in both cascaded and S2T systems, taking up over 80% of runtime and contributing to the maximum memory usage (Lee et al., 2021). Having to produce speech from target text also renders S2T unsuitable for translating into unwritten languages, and also eliminates any speech characteristics or paralinguistic information present in the source speech.

A modern alternative proposed by Jia et al. (2019b) is to eliminate the TTS component and directly translate from a source spectrogram into a target spectrogram (S2ST). This is successfully applied to unwritten language translation tasks, and later developments of the original model are also able to preserve paralinguistic information and speaker identity (Jia et al., 2022a). However, there is still a gap in translation quality between S2ST and S2T, evidenced by an over-generation issue. This is likely because the attention mechanism has a much longer target sequence it now needs to map the input to. Ideally we want to preserve the translation quality from S2T, but maintain lower computational costs and unwritten capabilities from E2E approaches.

2.3 Speech-to-Unit translation

Speech-to-Unit Translation (S2UT) is a novel approach to ST that replaces the target sequence with a series of discrete units generated directly from the target audio (Lee et al., 2021). These units can be learnt using unsupervised methods such as HuBERT (Hsu et al., 2021) or wav2vec (Schneider et al., 2019): currently HuBERT is more com-

monly used based on evidence of its superior performance across ASR and language modelling tasks (Yang et al., 2021). HuBERT is trained via masked prediction to encode the target audio with continuous representations at regular intervals. A k-means clustering algorithm can then be applied to those representations to quantise the representations, labelling them with their cluster index into discrete units. Intuitively, these units can be thought of as "pseudo-phonemes", although this description depends on the granularity of the quantisation, i.e. how many clusters the k-means model is trained to create. These units are then used to train a lightweight vocoder, which replaces the TTS component from cascaded models. This is visualised in figure 2.1:

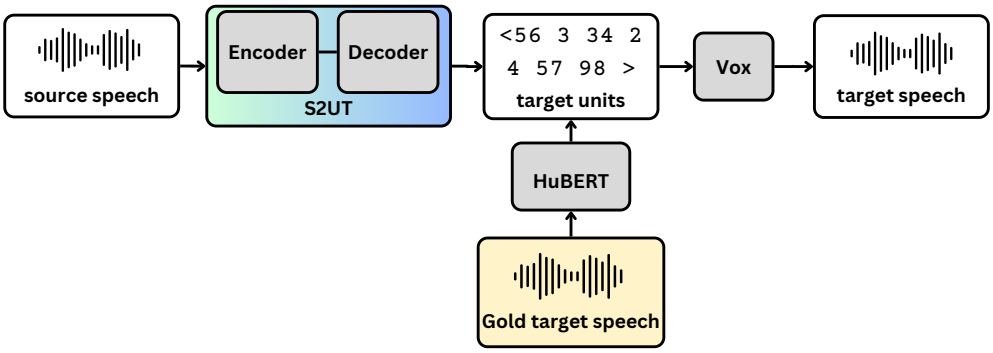

Figure 2.1: S2UT architecture. The gold standard target speech is used to produce the target units used during training.

Lee et al. (2021) show that this approach can be extended to unwritten languages: if the target language is unwritten, multilingual HuBERT can be used to extract target units. But performance can be improved for written languages by supplementing training with multitask learning, where intermediate representations from the S2UT model are extracted and used to predict source or target characters, phonemes or any other discrete linguistic representation. S2UT is shown to dramatically reduce memory usage, computational cost and inference latency compared to S2T or S2ST, while reporting similar translation accuracy. Although paralinguistic information and speaker identity is lost by discretising the target sequence in this way, this may be easier to recover than for S2T, since the target sequence is more granular, and the HuBERT discrete units are shown to be phoneme-like (Wells et al., 2022).

To extend S2UT, or any E2E model to the task of document-level ST, we must still address the issue of encoding long audio sequences. If the input to our E2E model is speech, incorporating extra-sentential information presents a memory and computational challenge, requiring architectural or data manipulation strategies to solve. The next chapter introduces one such strategy: Adaptive Feature Selection.

Chapter 3

Adaptive Feature Selection (AFS)

One of the biggest challenges for speech recognition tasks like ASR and ST is that speech information is unevenly distributed throughout the audio signal. Speech sounds and pauses have variable durations, and extracting only useful features is difficult. Traditionally this extraction is done with convolution layers, which reduce the input across the temporal and/or feature dimension by a fixed factor. This is effective for compressing the input sequence, but a more flexible method designed to directly remove redundancy prevents useful information being compressed and diluted with distracting features. Adaptive feature selection (AFS), proposed by Zhang et al. (2020) builds on convolution by learning to selectively mask features if they are found not to be useful for a task. This is accomplished via the sparsity-inducing method $\mathcal{L}_0$DROP.

3.1 $\mathcal{L}_0$DROP

At the heart of $\mathcal{L}_0$DROP is a gating function g_i, sampled from the HardConcrete distribution (Louizos et al., 2017). This distribution is a variation of the Concrete distribution (Gal et al., 2017), but stretched and with hard-sigmoid applied. Under the hyperparameter specification described by Louizos et al. (2017, pp. 5) the HardConcrete distribution assigns roughly half its probability mass to $\{0,1\}$ and half to (0,1).

Each encoded feature $\mathbf{x}_i \in \mathbb{R}^d$ is assigned a scalar value $g_i \in [0,1]$ as follows:

$$\mathcal{L}_0\text{DROP}(x_i) = g_i x_i, \tag{3.1}$$

$$g_i \sim \text{HardConcrete}(\alpha_i, \beta, \varepsilon), \tag{3.2}$$

where β governs the temperature of the HardConcrete distribution and ε is a small

smoothing hyperparemter (Louizos et al., 2017). α_i directly determines the shape of the HardConcrete distribution and this value is predicted by a neural layer:

$$\log \alpha_i = \mathbf{x}_i^T \cdot \mathbf{w} \tag{3.3}$$

This way, L_0DROP can control g_i via α_i, and by applying the penalty function described in equation 3.4, encourages the probability mass of the HardConcrete distribution to move closer to 0:

$$\mathcal{L}_0(X) = \sum_{i=1}^{n} 1 - p(g_i = 0 | \alpha_i, \beta, \varepsilon) \tag{3.4}$$

When g_i is sampled with reparameterisation (Kingma and Welling, 2013) this loss function is fully differentiable. Now we have formalised the mechanism underlying AFS we can use it to train a feature extractor.

3.2 Training AFS

AFS must be trained to extract features useful for our task of ST. But both our S2UT architecture and bilingual data are complex to model, so we instead train AFS on a ASR task, and make the simplifying assumption that useful features for ASR will also be useful for ST. The full training procedure for AFS, as described in Zhang et al. (2020) is illustrated below:

Training AFS

1. Train ASR model with following objective and architecture until convergence:

$$\mathcal{L}^{\text{ASR}} = \eta \mathcal{L}_{\text{MLE}}(Y|X) + \gamma \mathcal{L}^{\text{CTC}}(Y|X) \tag{3.5}$$

$$\mathcal{M}^{\text{ASR}} = \mathcal{D}^{\text{ASR}}(Y, \mathcal{E}^{\text{ASR}}(X)) \tag{3.6}$$

2. Finetune ASR model with AFS for m steps:

$$\mathcal{L}^{\text{AFS}} = \mathcal{L}_{\text{MLE}}(Y|X) + \lambda \mathcal{L}_0(X) \tag{3.7}$$

$$\mathcal{M}^{\text{AFS}} = \mathcal{D}^{\text{ASR}}(Y, F(\mathcal{E}^{\text{ASR}}(X))) \tag{3.8}$$

3. Freeze encoder and AFS together and use as feature extractor before ST model encoder

First, an encoder-decoder ASR model is trained to transcribe audio in the source language for the subsequent ST task. This is trained with maximum likelihood estimate (MLE) loss and Connectionist Temporal Classification (CTC) loss on the encoder output, to help with alignment between the encoder output and the target sequence, and to encourage the encoder to produce semantically useful representations. Once this model has converged, the AFS module is inserted after the encoder, to sparsify its output before passing this to the decoder. This model is then finetuned for an extra m steps. During this step, the CTC loss is removed, to relax the alignment constraint and improve the flexibility of AFS (Zhang et al., 2020). At this stage, the $\mathcal{L}_0$ loss described in equation 3.4 can be incorporated with the MLE loss to form the full AFS loss function for step 2:

$$\mathcal{L}_{\text{AFS}} = \mathcal{L}_{\text{MLE}}(Y|X) + \lambda \mathcal{L}_0(X), \tag{3.9}$$

where λ is an interpolation parameter controlling sparsity – a larger λ encourages more gates to go to 0. L_0DROP can be easily applied to both temporal and feature dimensions, by also learning a feature independent gating model with trainable parameter $\mathbf{W}$. Zhang et al. (2020) found that feature dimension $\mathcal{L}_0$DROP doesn't actually prune away features but rather acts as a weighting factor scaling down less important neuron output.

Finally, in step 3 the decoder is dropped and the frozen encoder and AFS modules now form the feature extractor to be inserted before the encoder of the ST model.

Chapter 4

Adding context: concatenative speech translation

With AFS we can now reduce the dimensionality of the input sufficiently to tackle document-level ST. Here I adopt the technique described by Zhang et al. (2021) of concatenative-ST, where longer paired documents are segmented roughly into sentences. The model then translates each sentence, but extra-sentential context is incorporated by concatenating it with the input for the current sentence. More formally, given a pre-segmented source document $\mathbf{A} = (\mathbf{a}^1 ... \mathbf{a}^N)$, and its corresponding pre-segmented target document $\mathbf{Y} = (\mathbf{y}^1 ... \mathbf{y}^N)$, where a^n and y^n are document segments, and N is the total number of segments in the document, the goal is to maximise the following:

$$\log p(\mathbf{Y}|\mathbf{A}) = \sum_{n=1}^{N} \log p(\mathbf{y}^n|\mathbf{x}^n, \mathcal{C}_x^n, \mathcal{C}_y^n) \tag{4.1}$$

where $\mathbf{x}^n = \text{AFS}(\mathbf{a}^n)$ and $\mathcal{C}_x^n, \mathcal{C}_y^n$ refer to the source and target context segments respectively, i.e. $\mathcal{C}_x^n = \{\mathbf{x}^{n-i}\}_{i=1}^{C}$. After segmenting the audio, AFS is applied to extract features for each source segment. During training when a segment is to be translated, C prefix segment features are concatenated to the current segment features. Likewise, C prefix target sequences are concatenated with the current target sequence for teacher-forcing, separated with "$\langle \mathbf{s} \rangle$" . This is so the current segment can be retrieved, either to compute the loss over the current segment during training, or to retain only the current segment of output during inference, as described in section 4.1.

4.1 Inference Strategies

Zhang et al. (2021) explored different inference strategies to take advantage of concatenative-ST, with a tradeoff between computational cost and translation accuracy. These strategies are visualised in figure 4.1. The first strategy is Chunk-based Decoding (CBD), where source and target documents are split into N non-overlapping chunks, each with C segments. These chunks are translated independently of one another. This is the simplest and most computationally efficient strategy but was found to produce misaligned translations (Zhang et al., 2021, pp. 2568), where fewer or more target sentences are produced than desired.

The next strategy, Sliding-Window-Based Decoding (SWBD), mitigates the misalignment issue by translating each segment, moving the chunk window forward not an entire chunk, but rather a segment at a time, adding only the final segment of output to the final document translation. This is slower than CBD since each segment is translated several times, but Zhang et al. (2021) found SWBD improves the alignment issue, although doesn't solve it completely. They attributed this to a mismatch between prefixes at different time steps, as prefix segments are translated from scratch at each decoding step. For example, after translating segments *w*-*x*-*y* into *W*-*X*-*Y*, the model then translates *x*-*y*-*z* into *X′*-*Y′*-*Z*; there is a mismatch between X,Y and X',Y'. To counter this they introduced SWBD-cons(trained) and In-Model Ensemble-Decoding (IMED). Both of these techniques use any previously produced segments as decoding constraints, after which the model need only produce one segment, so in our example after producing *W*-*X*-*Y* and coming to translate *x*-*y*-*z* the model is forced to produce *X*-*Y*- after which it can produce *Z*.

In IMED, the document-level prediction of the current segment (p^d) from SWBD-Cons is interpolated with a sentence-level prediction (p^s) as follows:

$$p^{\text{IMED}}(y_t^n|C) = \gamma p_\theta^s(y_t^n|y_{<t}^n, x^n) + (1-\gamma)p_\theta^d(y_t^n|C) \tag{4.2}$$

$$C = \{C_x^n, C_y^n, x^n, y_{<t}^n\} \tag{4.3}$$

where γ is the interpolation parameter[1], y_n^t denotes the t-th target unit in the n-th segment and both models are derived from the same model θ. The sentence-level prediction acts as a regulariser on the document-level prediction, encouraging the model not to include material from previous segments in its output for the current segment. IMED

[1]Note that Zhang et al. (2021) they use λ for this parameter, but to avoid confusion with the AFS $\mathcal{L}_0$ interpolation parameter described in section 3.1 I use γ in this study.

is the foremost contribution of Zhang et al. (2021), but in their leading experiment a variant of SWBD performs best across several metrics.

In this paper I draw on the success of SWBD and explore two inference strategies: SWBD and a new SWBD variant that inherits the sentence-level prediction interpolation from IMED, henceforth named SWBD-IMED. I hypothesise the following:

1. Using a sentence-level prediction as a regulariser for the document-level prediction can be integrated into standard SWBD, without constrained decoding
2. This regulariser will result in a lower Bleu score, at an optimum value of γ.
3. The strength of IMED can be attributed only to the sentence-level prediction interpolation, not to the constrained prefix decoding.

After implementing S2UT with AFS, I answer these hypotheses and discuss the outcomes in section 5.3.3.

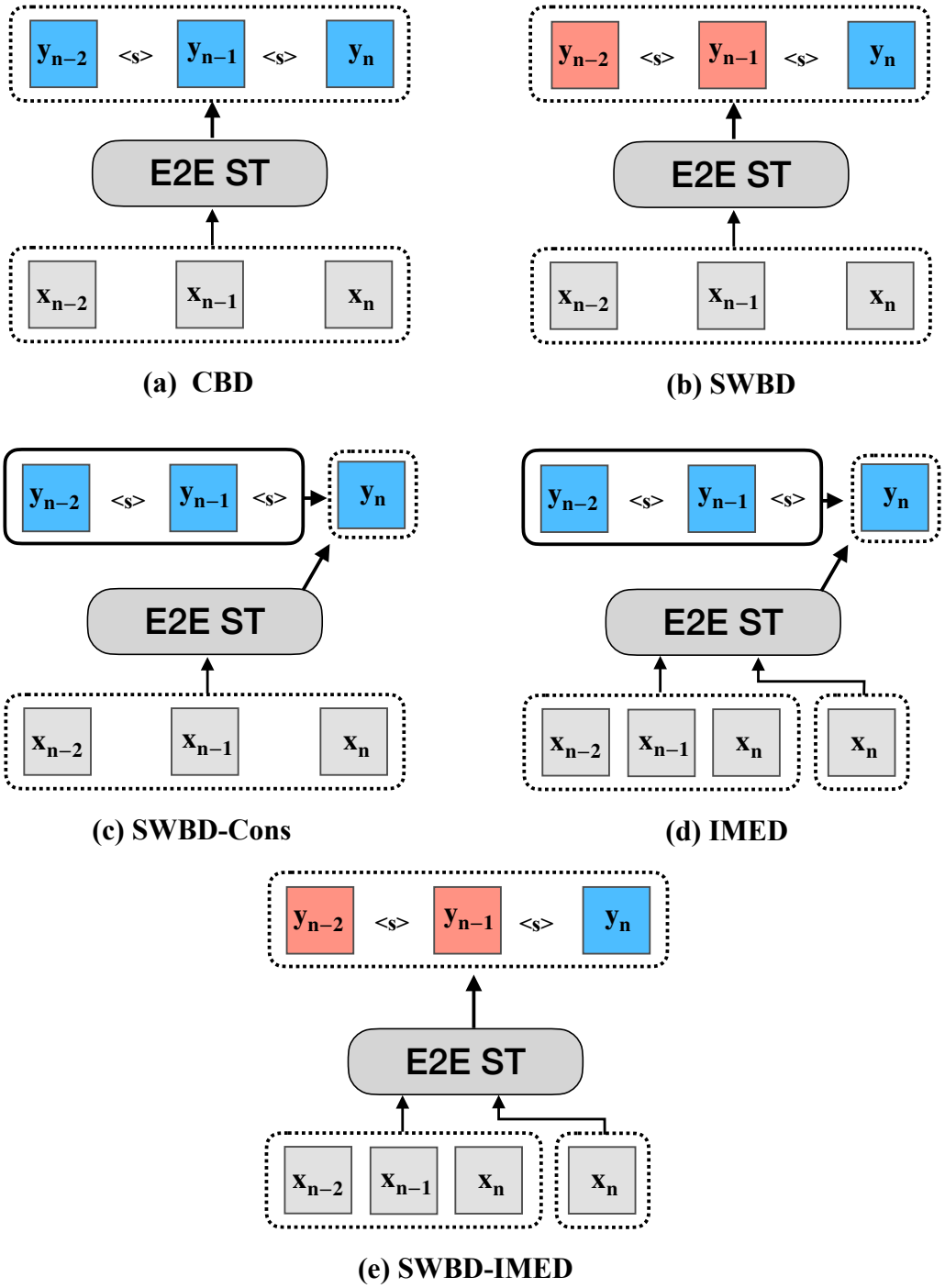

Figure 4.1: (a) Chunk Based Decoding (b) Sliding Window Based Decoding (c) Sliding Window Based Decoding - Constrained (d) In Model Ensemble Decoding (e) SWBD with IMED. $\mathbf{x_n}$ and $\mathbf{y_n}$ are segments of input and output respectively, where one segment is roughly a sentence. Dashed boxes show the model's inputs and outputs for the current decoding step. Blue boxes show output segments added to the final document translation. Red boxes show dropped output segments. Thick lined blocks show decoding constraints.

Chapter 5

Experiments

5.1 Baseline

I first implemented a baseline S2UT model following Lee et al. (2021). This was done mainly using FAIRSEQ (Ott et al., 2019) with some auxiliary functions for preparing the multitask data as described in section 5.1.2.

5.1.1 Data

I experimented with two datasets when implementing the baseline, the Fisher Spanish-English dataset (Post et al., 2014) and CVSS Spanish-English (Jia et al., 2022b). Fisher is a dataset of Spanish telephone conversations, paired with English transcriptions, and was the dataset used in Lee et al. (2021). I closely followed the same pre-processing steps (Lee et al., 2021, pp. 4): I used a pre-trained Fastpitch-2 model (Łańcucki, 2021; nvidia, 2024) to synthesise the target text into gold-standard target audio. I then used Whisper medium (Radford et al., 2022) to transcribe the gold audio, and compared the transcriptions against the gold-standard text, removing any samples with a Word Error Rate (WER) greater than 80. All audio was then down-sampled to 16Khz.

CVSS is a subset of the CoVoST 2 (Wang et al., 2020) dataset, with speech in multiple languages with paired text translations: I again use Spanish to English. In CVSS that target text is synthesised with a strong proprietary TTS model. Again all audio was down-sampled to 16Khz. Both datasets were then split into training, validation and test sets according to their own specifications, as shown below:

Fisher					
	Train	Dev	Dev2	Test	Total
Samples	138,789	3977	3959	3641	150,366
Hours	171:32	4:35	4:42	4:28	185:19
Avg duration (s)	4.49	4.15	4.27	4.42	4.44

Fisher post-processing					
	Train	Dev	Dev2	Test	Total
Samples	114,232	3131	3155	2980	123,498
Hours	160:45	4:14	4:18	4:12	173:31
Avg duration (s)	5.07	4.87	4.91	5.07	5.06

CVSS					
	Train	Dev	Dev2	Test	Total
Samples	45,690	11,030	N/A	12,036	68,756
Hours	70:05	18:38	N/A	20:59	109:43
Avg duration (s)	5.52	6.08	N/A	6.28	5.74

Table 5.1.1: Number of samples, total hours and average duration per sample for Fisher and CVSS datasets. The data statistics for Fisher post- processing are closely aligned with those found in Lee et al. (2021, pp. 4).

5.1.2 Model

The architecture of the baseline model follows the description in Lee et al. (2021, pp. 4), visualised in figure 5.1. Both the quantisation model used to extract the discrete units from the gold standard speech, and the vocoder trained to predict English speech from those units were pre-trained (Lee, 2024). I implement the best model variant in Lee et al. (2021) and use "reduced units" where after quantisation, any consecutive repeating units are collapsed into a single unit, resulting in a string of unique units.

During training 80 dimensional Mel-filterbank features are extracted from the source speech every 10ms, with cepstral mean and variance normalisation and SpecAugment (Park et al., 2019) applied. These features were then passed to an encoder comprising a subsampling module (2 convolution layers, kernel size = 5, # channels = 1024) and 12 transformer layers (embedding size = 256, feed-forward network (FFN) embedding

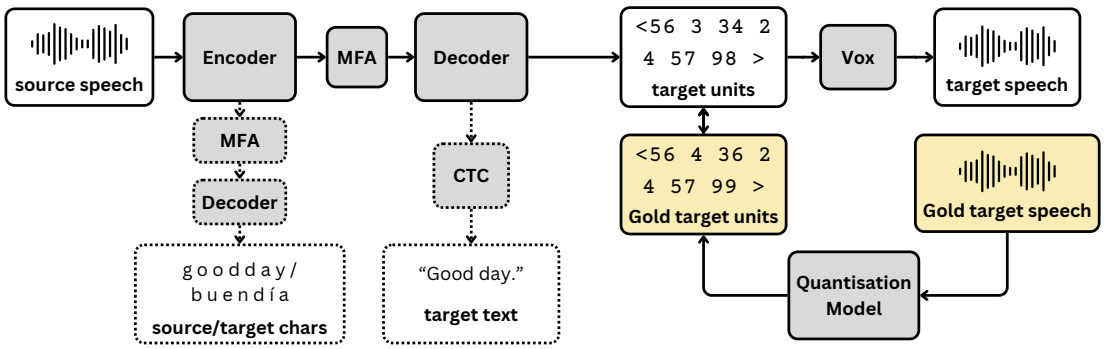

Figure 5.1: Baseline model architecture. Dashed boxes indicate tasks performed during training but not at inference. Note that unlike in Lee et al. (2021), target text is not produced at inference time.

size = 2048, # attention heads = 4). The decoder had 6 transformer layers (embedding size = 256, FFN embedding size = 2048, # attention heads = 8).

Lee et al. (2021) perform multitask learning to encourage the model to converge and experiment with predicting source and target characters and phonemes. I use source and target characters, following their findings that this configuration provides the best performance. A separate attention module (# heads = 4) and decoder (embedding size = 256, FFN embedding size = 2048, # layers = 2) are attached to the sixth and eighth layers of the encoder to predict source and target characters respectively. CTC decoding is also used to jointly predict text output during training: the CTC decoder is conditioned on the third layer of the discrete unit decoder. The loss weights for source character, target character, and CTC target text decoding were 8.0, 8.0 and 1.6 respectively.

The model was trained for 400K steps using Adam to schedule the learning rate ($\beta_1 = 0.9$, $\beta_2 = 0.98$, $\varepsilon = 10^{-8}$). Label smoothing of 0.2 and an inverse square root learning rate decay schedule with 10k warmup steps were also used. All other hyperparameters followed the specification provided by Lee (2024).

5.1.3 Results

The most common metric used for evaluation of E2E ST models is ASR-Bleu, where the output speech is transcribed using an ASR model, and a Bleu score (Papineni et al., 2002) is computed over the transcription. Since the ASR model introduces the potential for error in the pipeline, this score can be thought of as a lower bound on translation quality.

Baseline Experiment			
ID	System	ASR-Bleu ↑	COMET ↑
0	META Gold	90.5	N/A
1	META Fisher	39.9	N/A
2	Gold Fishe	84.27	90.45
3	Fisher pre	12.93	60.90
4	Fisher post	13.58	**63.13**
5	CVSS	**16.59**	59.02

Table 5.1.2: Results for the baseline model trained on two datasets, CVSS and Fisher post-processing. Gold refers to the evaluation of the gold-standard discrete units after being synthesised and transcribed. Equivalent results from Lee et al. (2021) are also given for comparison, starting with "META".

During inference all discrete unit sequences produced by the model for the Fisher and CVSS test sets were passed through the pre-trained vocoder to produce output speech. This was transcribed using Whisper-medium and Bleu was calculated using SACREBLEU[1] (Post, 2018). Lee et al. (2021) used 4 references when computing Bleu score; due to time limitations only 1 batch of references is used in this work. It is unclear how significant of an impact this may have had on the final scores. Finally, I also computed COMET scores (Rei et al., 2020), a neural metric designed to emulate human translation scores. This was largely a sanity check as Bleu score does not always provide an accurate metric for individual text translation quality (Reiter, 2018). The results for the baseline experiment are given in table 5.1.2.

It is clear from these results that neither the model trained on CVSS nor the model trained on the original Fisher dataset achieved a performance comparable to what was reported in Lee et al. (2021). It is worth highlighting the difference between the two gold-standard results: this is likely due to a stronger ASR model used to compute ASR-Bleu, since both the quantisation model used to produce the target discrete units and the vocoder used to synthesise them were pre-trained checkpoints from Lee (2024). The performance difference in the evaluation pipeline only partially explains the lower ASR-Bleu score on the Fisher test set (1 vs 4). The cause is unlikely to be data pre-processing differences, since due to a weaker ASR model more samples were removed than by Lee et al. (2021) (see table 5.1.1). Furthermore there was only a small perfor-

[1] signature: nrefs:1—case:lc—eff:no—tok:13a—smooth:exp—version:2.4.2.

mance gain from performing this pre-processing (3 vs 4), so a small change in number of samples is unlikely to have a large effect. Careful investigation of the training data, including multitask data did not reveal any formatting issues. The architectures (1 vs 3-5) were the same, although the hardware used to train the models was likely different[2]. Although not described in Lee et al. (2021) there may be some other hyperparameters or a seed values may have differed. This drop in baseline performance and its knock-on effects are discussed further in section 6.

It is not fully clear from table 5.1.2 whether CVSS or Fisher produced the stronger model. The CVSS test set is much larger than the Fisher test set, contained longer sentences with more varied material: upon closer inspection many of the high Bleu and COMET scores for 3 and 4 came from one word utterances. This fact, paired with the higher Bleu score led me to proceed with 5 for future experiments, henceforth referred to as *CVSS*.

5.2 Adding AFS

Before tackling document-level ST, we should first examine how applying AFS to the input affects model performance. Here I train an AFS feature extractor and subsequently re-train *CVSS*, but adapted to accept AFS features.

5.2.1 Model

For the first stage of AFS training, an encoder-decoder model is trained on an ASR task, with Cross Entropy (CE) loss, and CTC loss conditioned on the encoder output. The architecture for this task follows the description in Zhang et al. (2020, pp. 2536) and was largely implemented using the standard FAIRSEQ `s2t_transformer` architecture. The encoder and decoder both have 6 layers (embedding size = 512, # attention head = 8, FFN embedding dimension = 2048), and the encoder has the same subsampling module described in section 5.1.2. The loss weight for CTC was set to 0.3, and Adam was used to schedule learning rate updates ($\beta_1 = 0.9$, $\beta_2 = 0.98$). This model was trained for 300 epochs.

This model is then fine-tuned for an additional 40 epochs with the AFS module inserted between the encoder and decoder, filtering the encoder output before passing it to the decoder. The AFS module has FFN layer that maps from the encoder embedding

[2] 3-5 trained on 4 Tesla V100-SXM2-16GB GPUs

size to 1, used for computing g_i for each feature. During this phase the CE loss is interpolated with the $\mathcal{L}_0$ loss according to the parameter λ. This loss is also regularised, ramping up linearly for a 10K updates. I experiment with different values for λ, and discuss its impact in section 5.2.3.

Finally the encoder and AFS modules are frozen, and replace the sub-sampling module in *Baseline*. I shall refer to the frozen encoder and AFS modules as AFS going forward. *CVSS* is then fine-tuned with AFS, using the same parameters as described in section 5.1.2.

5.2.2 Data

AFS was trained using Voxpopuli data (Wang et al., 2021), Spanish audio and paired transcriptions from European Parliment event recordings. I split this dataset into two halves and used the first half during ASR pre-training, and the second half during AFS fine-tuning, henceforth Voxpopuli 1 and 2 respectively. The statistics and splits for these two sets are given in table 5.2.1. For the S2UT + AFS training I use CVSS.

Voxpopuli 1 - ASR pre-training				
	Train	Dev	Test	Total
Samples	25,460	816	764	27,040
Hours	75:58	2:37	2:26	81:02
Avg duration (s)	10.74	11.54	11.47	10.79

Voxpopuli 2 - AFS fine-tuning				
	Train	Dev	Test	Total
Samples	25,462	816	765	27,043
Hours	76:01	2:36	2:26	81:04
Avg duration (s)	10.75	11.47	11.45	10.79

Table 5.2.1: Number of samples, total hours and average duration per sample for the Voxpopuli dataset splits used during AFS training

5.2.3 What does AFS remove?

Introducing AFS presents a challenge: we want to remove as much redundant input as possible without losing useful features. After the ASR pre-training task I fine-tune with AFS, and experiment with different λ values to try and find the best balance

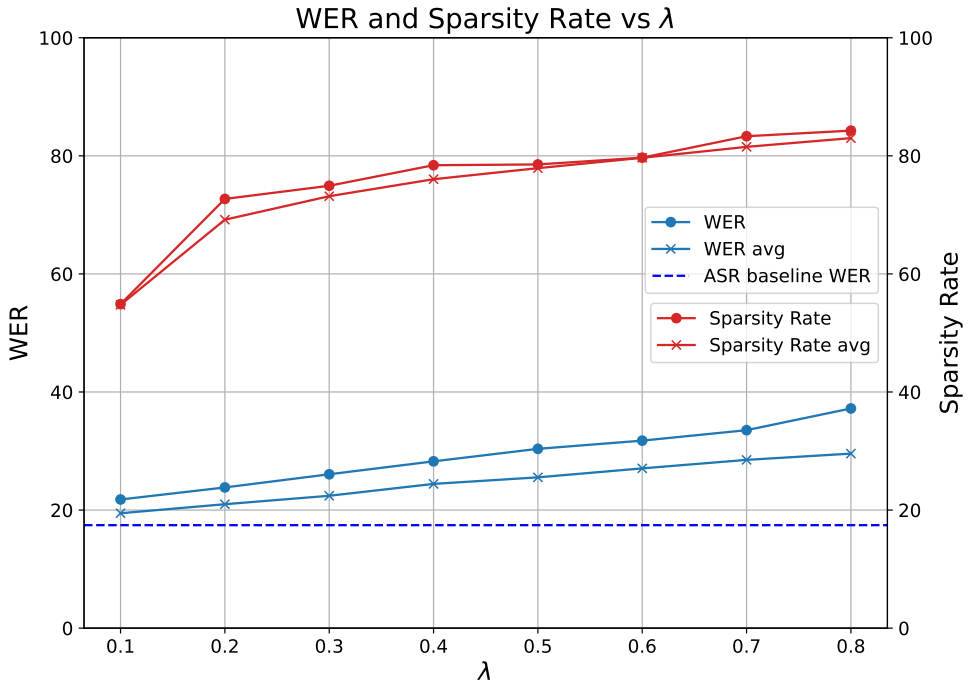

Figure 5.2: WER and temporal sparsity rate for AFS for different λ values. The dashed line shows the WER for the ASR pre-trained model fine-tuned without AFS. Results are given for the final checkpoint and for the average of the final 5 checkpoints, for each variant.

between ASR performance and sparsity rate, the percentage of features pruned away by AFS. For simplicity I record temporal sparsity rate only. Zhang et al. (2020) showed that while feature sparsity remains 0 for all λ values, the value for the feature gate g_f^i, which acts as a weighting mechanism on the features, is correlated with temporal sparsity rate. I perform the same sweep over λ values as performed by Zhang et al. (2020, pp.2537), in the range [0.1, 0.8] with a step of 0.1. I train these AFS variants on Voxpopuli 2, for 40 epochs, and average the weights of the final 5 checkpoints for evaluation. For comparison I perform the same fine-tuning regime with the ASR pre-trained model without AFS. For evaluation I compute the WER and average sparsity rates for each model on the Voxpopuli 2 test set.

The trade-off between ASR performance and sparsity rate is illustrated clearly in figure 5.2. It is worth noting however that for $\lambda > 0.1$ the sparsity rate saturates around 80%. This is a similar to the sparsity rate found by Zhang et al. (2020, pp. 2537), although they experience saturation around $\lambda > 0.5$. They applied AFS to a S2T task; since the target sequence for S2UT is more granular I hypothesise that it will be more

difficult to remove features and still accurately predict the target units, so a sparsity rate lower than their chosen value of 85% will be required to achieve similar performance for S2UT. For this reason, I use AFS trained with $\lambda = 0.1$ and $\lambda = 0.3$ for future experiments.

Figure 5.3 illustrates which features are being kept by AFS, using some example samples from Voxpopuli 2. Quieter parts are rarely kept, and short portions from most words are kept, providing support that AFS removes intuitively redundant features from the audio signal. Comparing the features kept by AFS for the three λ values, both $\lambda = 0.3$ and $\lambda = 0.1$ keep more consecutive features: in order to cover the same duration of audio but keep sparsity rate high, AFS with $\lambda = 0.8$ must leave many more gaps between features. For a target sequence of words of subwords this may not cause too many issues, since there are still plenty of features remaining to inform the choice of target. However for a target sequence of phoneme-like discrete units, leaving this many gaps is likely damaging. I discuss this at greater length in section 6.3. For $\lambda = 0.1$ and $\lambda = 0.3$ we see the models selecting larger chunks from most words, with some function words like "las" ("the") being ignored. Interestingly $\lambda = 0.3$ drops some content words, "seguir mirando" ("continue looking"), which $\lambda = 0.1$ doesn't. Although it is hard to show empirically that there is a wider trend for higher λ value AFS to drop more content words, this may be a partial explanation for the drop in WER from the $\lambda = 0.3$ model to the $\lambda = 0.1$ model.

5.2.4 Results

In this section I tackle the following research question: Does replacing the subsampling module from the *CVSS* S2UT model with AFS improve translation accuracy? To answer this, I train an S2UT model, whose architecture and training regime is identical to *CVSS* as described in sections 5.1.1–5.1.2, except the subsample module is replaced by AFS. Two variants of this architecture are trained, one using AFS trained with $\lambda = 0.1$ and another with AFS trained with $\lambda = 0.3$.

Theses model are then evaluated on the CVSS test set, and resulting ASR-Bleu and COMET scores added to table 5.1.1, given here as table 5.2.2. Unfortunately we see a drop in performance for both models using AFS (6-7), with a significant drop for 7. These findings seem to contradict the results found by Zhang et al. (2020, pp. 2537), where applying AFS is shown to improve translation accuracy. The negative correlation between sparsity rate and performance indicate that for the task of S2UT

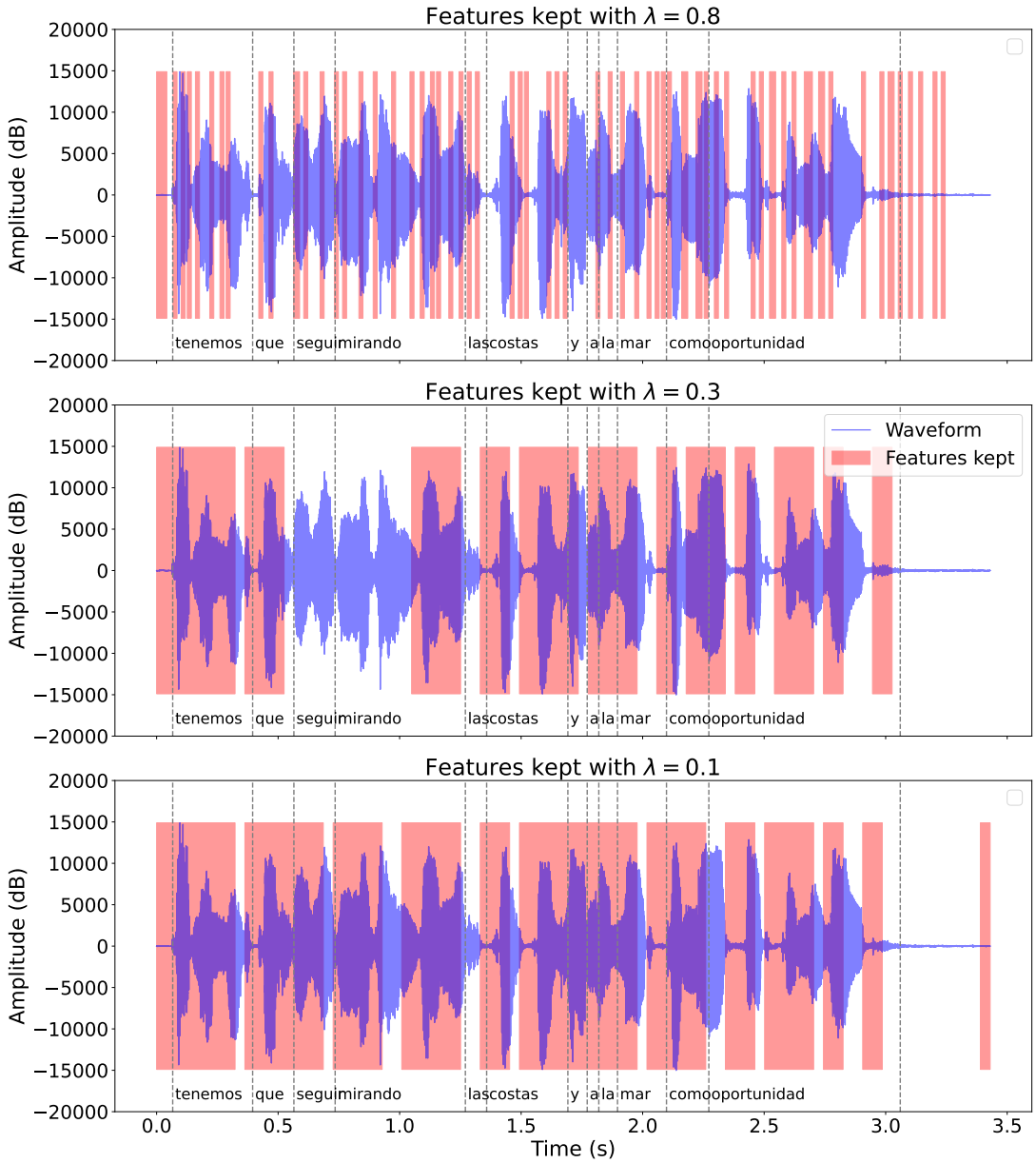

Figure 5.3: Sample audio, with overlay to show features selected by AFS, trained with $\lambda = 0.1$, $\lambda = 0.3$, and $\lambda = 0.8$

Baseline Experiment			
ID	System	ASR-Bleu ↑	COMET ↑
0	META Gold	90.5	N/A
1	META Fisher	39.9	N/A
2	Gold	84.27	90.45
3	Fisher pre	12.93	60.90
4	Fisher post	13.58	**63.13**
5	CVSS	**16.59**	59.02
6	CVSS + AFS ($\lambda = 0.1$)	12.67	53.71
7	CVSS + AFS ($\lambda = 0.3$)	7.40	46.40

Table 5.2.2: Results for the baseline models and from compared with CVSS with AFS as the subsampling module.

there is far less room for feature reduction than for S2T. As such, I progress only with *CVSS* + AFS $\lambda = 0.1$, which from here is referred to as *CVSS* + *AFS*. I discuss the viability of using AFS for S2UT in section 6.2 on page 31.

5.3 Adding extra-sentential context

The main focus of this work is to investigate whether incorporating extra-sentential information improves translation quality, taking a step closer to document-level translation in E2E ST. Here, I introduce a new translation task created using FAIRSEQ, named `Doc_Speech_to_Speech`, where I implement the concatenative-ST approach described in chapter 4 and use it to train *CVSS* + *AFS*.

5.3.1 Data

To perform concatenative-ST as described by Zhang et al. (2021), we must use data whose samples are longer audio documents, and segment them roughly into sentences. In these experiments I use Europarl-ST (Iranzo-Sánchez et al., 2020), a corpus of paired speech and translated texts from European parliamentary speeches. I follow the Europarl-ST instructions on how to segment these speeches into shorter utterances. I then apply the same pre-proessing as was used for the Fisher dataset, by synthesising the target text, applying ASR, and removing any samples whose transcriptions received

Europarl - pre-segmentation				
	Train	Dev	Test	Total
Samples	727	202	206	1135
Hours	21:34	5:39	5:19	32:33
Avg duration (s)	106.86	100.72	93.05	103.26

Europarl - post-segmentation				
	Train	Dev	Test	Total
Samples	7384	1935	1807	11,126
Hours	20:40	5:22	5:03	31:05
Avg duration (s)	10.08	9.99	10.07	10.06

Table 5.3.1: Number of samples, total hours and average duration per sample for Europarl-ST before and after both segmentation and pre-processing.

a WER score greater than 80. The resulting data statistics pre- and post-segmentation and processing are provided in table 5.3.1.

5.3.2 Experimental Setup

The model architecture used in this experiment remains largely unchanged from the description in section 5.2.1 on page 17. However the model is now adapted to accept a new argument for context size C. When translating segment $\mathbf{a}^n$ from document $\mathbf{A}$, segments $\{\mathbf{a}^{n-i}\}_{i=1}^{C}$ from document $\mathbf{A}$ are found and their features are prepended to the features for $\mathbf{a}^n$. The same is done with the relevant target sequences used for teacher-forcing, but these are separated with a "$\langle \mathbf{s} \rangle$" token to facilitate retrieval of the current segment.

During inference I experiment with SWBD and SWBD-IMED, where SWBD is essentially equivalent to SWBD-IMED with $\gamma = 0$, since this eliminates the sentence-level prediction unique to SWBD-IMED. In both approaches the model produces $C+1$ segments of output from $C+1$ segments of input, and only the final segment is retained. Since generated prefix segments are discarded in all conditions, this invites two approaches to training, where the loss is computed over the full output sequence dubbed "extended-loss", or over only the final segment of output. The idea is that the latter would encourage the model to focus entirely on the generation on the current segment, reducing the complexity of the learning task. However, this may prevent the

model from learning how to properly utilise contextual information found in the prefix segments, since it never has to produce accurate output for the first prefix segments segments. To fully explore the usefulness of extra-sentential context, I also propose an additional, more relaxed inference task, "ground-truth-prefix" where we assume the model generated perfect output for the first C segments; in other words, the model is first forced to produce the ground truth target units for the first C segments, then continues to generate as normal.

CVSS + AFS=0.1 is fine-tuned on Europarl-ST dataset for an additional 300 epochs under different configurations of the task parameters mentioned above, including without extra-sentential context ($C = 0$) for comparison. Additionally I train a document level model (C=1) without AFS, although to deal with the computational bottleneck, batch size must be dramatically reduced. All other training parameters remain unchanged from the description provided in section 5.1.2, page 14. At inference I explore SWBD-IMED, with γ values in [0,1] with step size 0.1. I also compare inference results with and without "ground-truth-prefix", using $\gamma = 0.5$ as a default.

5.3.3 Results

I evaluate the model variants mentioned above on the Europarl-ST test set, using ASR-Bleu, COMET as before. I also add document-ASR-Bleu, which is calculated by combining the hypotheses and targets for each segment of a document, then computing the Bleu score over the document. The results in table 5.3.2 provide evidence that incorporating extra-sentential context is useful in S2UT. Although no document-level model performed strongly without the ground-truth-prefix constraint, with this constraint both document-level models with AFS (5-6) outperformed the baseline by a considerable margin of up to +5.4 ASR-Bleu and +5.1 Doc ASR-Bleu. This suggests that the S2UT architecture used in this work is not strong enough to accurately produce a target unit sequence longer than one segment, but when given the first segment of output it is capable of using this with the previous source material to improve translation quality. There is also strong evidence that AFS is crucial to this improvement; the same model but trained without AFS (6 vs 7) does not converge, and is unable to learn anything from an input longer than one segment. In fact the Bleu score of 1.01 is generous, the model simply produces the same output regardless of the input. Document-level models trained with extended-loss also performed better than their counterparts with loss computed only over the current segment. This suggests that teaching the model

Document-level ST experiment				
ID	System	ASR-Bleu ↑	Doc ASR-Bleu ↑	COMET ↑
0	Europarl-ST Gold	71.05	73.38	84.90
1	Baseline (C=0)	8.18	10.63	**52.11**
2	Baseline + AFS (C=0)	3.90	6.05	48.34
3	Doc	1.77	3.22	42.40
4	Doc (w/ e-l)	3.40	5.95	44.48
5	Doc (w/ g-t-p)	12.30	14.20	48.4
6	Doc (w/ g-t-p, w/ e-l)	**13.58**	**15.73**	49.70
7	6 (w/o AFS)	1.01	2.11	43.93

Table 5.3.2: Results from the document-level experiment. ASR-Bleu is computed across segments, Doc ASR-Bleu is computed across documents. Europarl-ST Gold is the evaluation of the gold-standard target units. "g-t-p" refers to the inference task ground-truth-prefix, C is number of prefix source and target segments concatenated to current segment input and target. "e-l" stands for extended-loss, and refers to models trained with a loss computed over the full target sequence. For models where $C > 0$, by default I use SWBD-IMED with $\gamma = 0.5$

explicitly how to model the prefix segments teaches the models information useful to translation in general.

It is worth noting, that all these results are considerably lower than for CVSS and Fisher datasets. This likely has to do with sample duration: comparing the data statistics tables for CVSS and Fisher on page 14 with the table for Europarl-ST 23 shows that the Europarl-ST samples are roughly twice the duration. Due to the longer target sequences, scaling up sample duration incurs a large drop in performance for S2UT. This also impacts the synthesis and evaluation parts of the pipeline, evidenced by the lower Gold scores compared with CVSS and Fisher experiments.

Figure 5.4 shows the results of experimentation with γ values for SWBD-IMED, where γ closer to 1 indicates more weight given to the sentence-level prediction, and closer to 0 indicates indicates more weight for the document-prediction. In both cases, with and without ground-truth-prefix the difference in performance is slight (< 0.3 Bleu, < 0.4 COMET). This could simply be as a result of overall weak model performance, where a clearer trend may be exhibited by a stronger model. For ground-truth-prefix ASR-Bleu seems to correlate slightly negatively with γ, suggesting that the

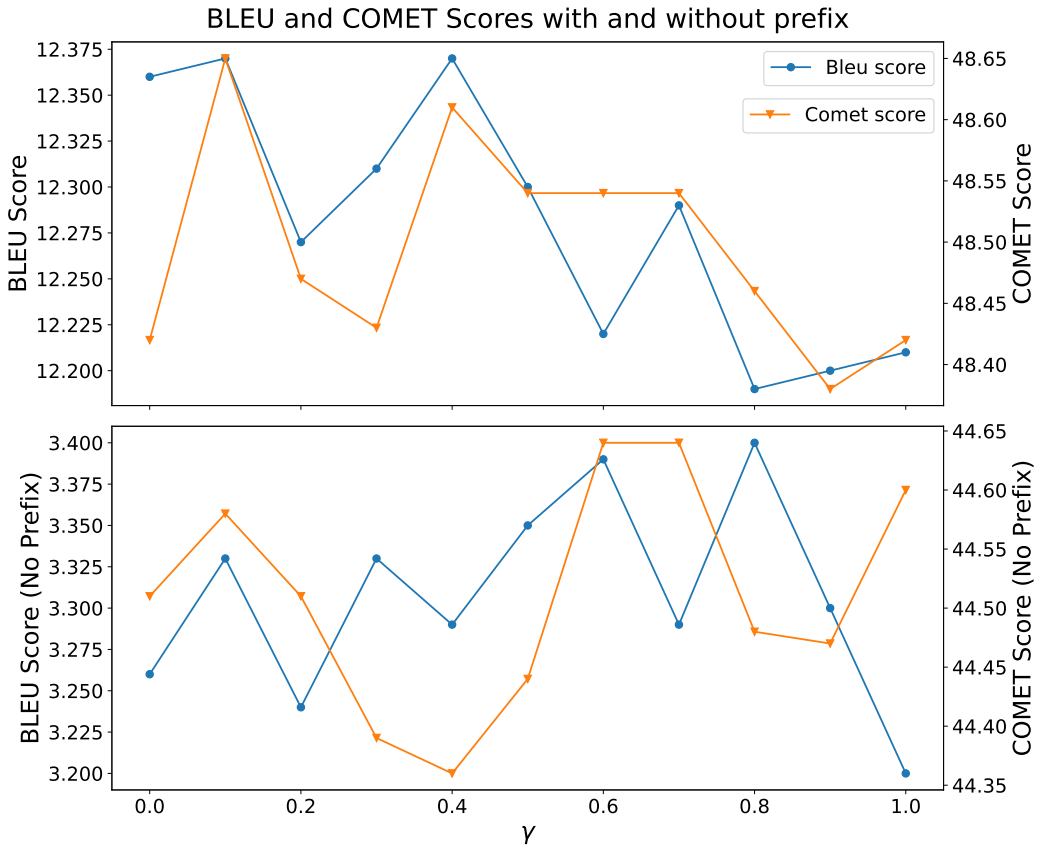

Figure 5.4: Bleu and Comet scores plotted for different values of γ. In both cases model 6 was used (see table 5.3.2), with and without ground-truth-prefix in top and bottom respectively

sentence-level is detracting from the overall prediction, while there is little evidence of a trend in COMET scores. Without ground-truth-prefix the opposite occurs, until a peak at $\gamma = 0.8$. This difference can be explained by the prefix information generated by the model: if it is of poor quality as may be the case for the no ground-truth-prefix task then weighting the sentence-level prediction benefits overall translation quality. The best γ value for SBWD-IMED is likely very dependent on data and architectural choices, but this experiment seems to give supporting evidence that sentence-level prediction interpolation can be adapted directly to SWBD without the need for the SWBD-cons decoding constraint, and provides a slight benefit when previous segment translation is sub-optimal.

Ablation Study: Random Context				
ID	System	ASR-Bleu ↑	Doc ASR-Bleu ↑	COMET ↑
0	Doc (w/ g-t-p, w/ e-l)	13.58	15.73	49.70
1	0 + Random $\mathcal{C}_x^n$	2.48	4.44	43.07
2	0 + Random $\mathcal{C}_y^n$	3.36	5.93	44.43
3	0 + Random $\mathcal{C}_x^n$, Random $\mathcal{C}_y^n$	2.7	4.55	43.18

Table 5.4.1: Ablation study replacing source and/or target side context with a segment from a randomly sampled document from Europarl-ST dataset. Model 0 is equivalent to model 5 from table 5.3.2 and has the following parameters: (C=1, w/ g-t-p, w/ extended-loss). $\mathcal{C}_x^n$ refers to source context and $\mathcal{C}_y^n$ refers to target context

5.4 To what degree is extra-sentential context relied upon?

The results clearly show that the model benefits from a wider window of source and target information, but it is difficult to discern how much, and in what way the document-level models are utilising information in the prefix source and target segments. To further verify the effectiveness of extra-sentential context, I perform an ablation study inspired by Zhang et al. (2021, pp. 2571). Instead of prepending the prefix segment of source and/or target features to the current segment, I prepend a segment from a random document from Europarl-ST. If the model does not use the prefix information then this change should not harm performance. The results from this ablation experiment in table 5.4.1 show a clear drop in performance in all scenarios. Interestingly, and contrary to the findings of Zhang et al. (2021), random source context is more damaging than both random target, and random source + random target contexts, although with Bleu scores this low, these differences could be due to chance.

To visualise the use of extra-sentential context by the model, cross-attention scores from models 5-6 on some samples from the Europarl-ST test set are given in figure 5.5. To see if the model is relying on source information from previous segments, we look to the upper left of the diagrams; indeed, we see that in these examples the model is using previous source features for the current segment predictions. These examples suggest strongly that training with extended-loss teaches the model to more effectively make use of previous contextual information, evidenced by the higher attention scores left of the source segment boundary. Furthermore, the extended-loss model often better learnt a temporal relationship between the input and output, which we can see from the diagonal pattern in attention scores (bottom). Some other interesting behaviours

are also highlighted by these diagrams. The vertical lines of high attention scores indicate that the same features are found to be useful for many target units, and the large spaces between these lines suggest there may still be considerable redundancy along the temporal dimension. These findings are discussed further in section 6.

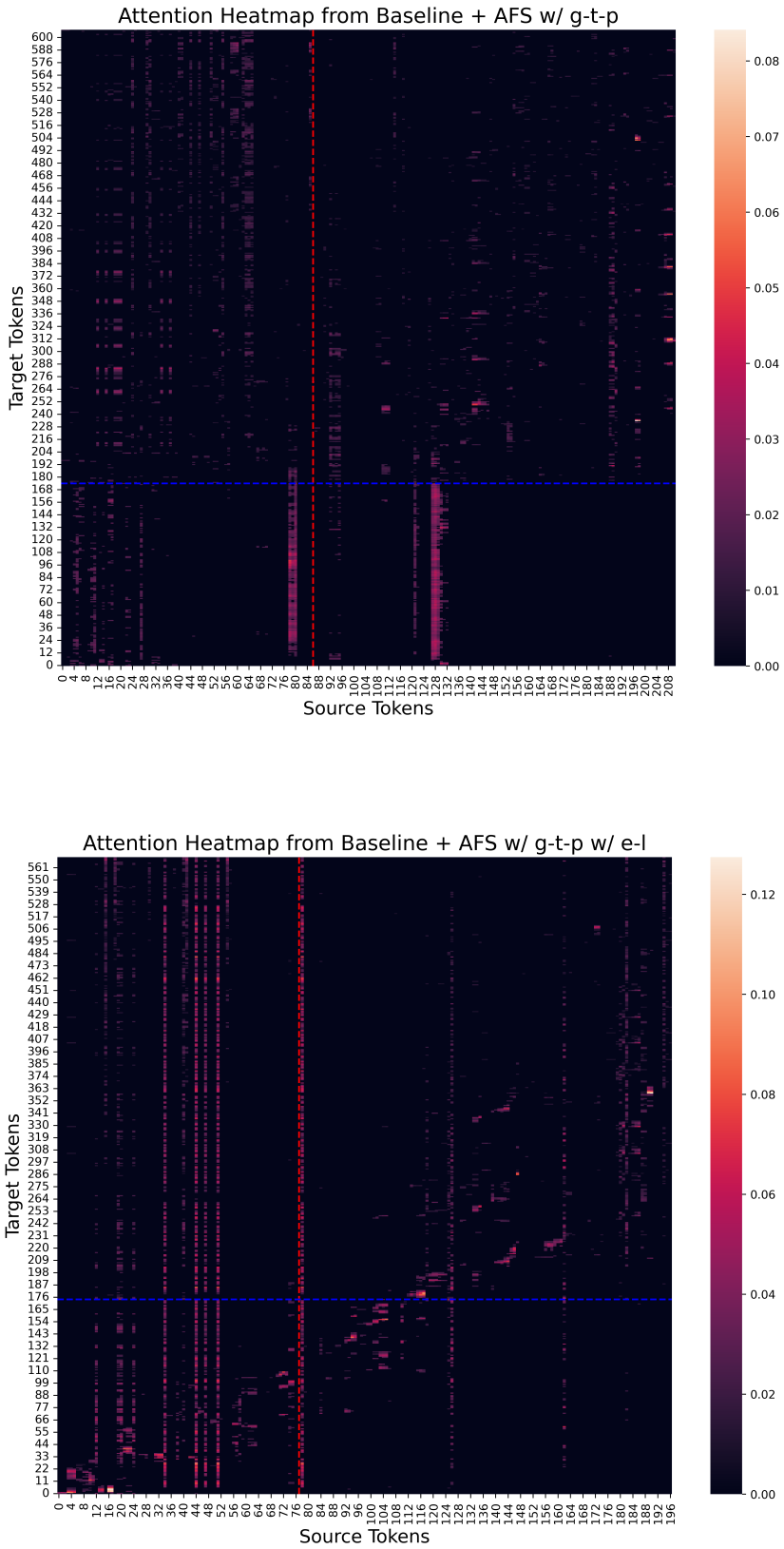

Figure 5.5: Heatmaps of cross-attention scores between S2UT encoder output and decoder previous output. Both heatmaps are for the same sample from Europarl-ST, from models 4 (top) and 5 (bottom) from table 5.3.2. To improve readability, only the 10 highest source scores are given for each target token. The red dashed line indicates the original source segment boundary, scaled to its relevant position in the encoder output. The blue dashed line indicates the target segment boundary.

Chapter 6

Discussion

6.1 The challenge of discrete unit translation

The poor performance of the S2UT *Baseline* model, as discussed in section 5.1, was surprising. Thorough error checking did not reveal anything to suggest this poor performance was due to the data preparation, synthesis or evaluation parts of the pipeline, so the most likely failure point is the S2UT translation model itself. This fact suggests the nature of the S2UT task is highly sensitive to training parameter, architectural and possibly data choices. This is reinforced by the dramatic changes in performance when making slight changes to the architecture, such as between AFS variants, or changing datasets to Europarl-ST. It is very common in the field of deep neural networks that small perturbations to training data, or internal network parameters can have large knock-on effects for downstream prediction tasks (Shu and Zhu, 2019), but this only partly explains such exaggerated changes in performance from between the discussed models.

Modelling a sequence of discrete units in another language from speech audio is clearly very difficult, even compared to a task like S2T. A target sequence of words or sub-words is both shorter than the equivalent discrete unit sequence, and is likely distributed in a more semantically meaningful way, allowing the model to better learn which parts of the input are useful for which parts of the output. The attention heatmaps on page 29 suggest the models are not always learning a linear relationship between features and target units, for although we wouldn't expect a perfect correlation along the temporal dimension, we also shouldn't expect that certain features are used the most across all target units, even across segments.

The difficulty of the S2UT task is not unknown; other notable works have trained

S2UT models on CVSS and achieved similar ASR-Bleu scores (Inaguma et al., 2022, pp. 5), and other approaches have been proposed to ease the learning task. These include pre-training the decoder on unit-mBART (Popuri et al., 2022; Liu et al., 2020), applying ASR pre-training on the encoder (Stoian et al., 2020) and data augmentation (Jia et al., 2019a). Despite performance lower than expected, it is still clear that leveraging extra-sentential contextual information is possible, and has proven very useful even in the challenging domain of S2UT.

6.2 The viability of Adaptive Feature Selection

The ratio of speech features to target speech units is much lower than for target subwords as were used in the original works on AFS (Zhang et al., 2020, 2021). This work provides evidence that at this lower ratio, it becomes much more difficult to remove features without quickly losing the ability to predict these short phoneme-like units. Zhang et al. (2020) found an optimum λ value of 0.5 when using AFS on ST: this work shows a negative correlation between sparsity rate and sentence-level ST performance, with $\lambda = 0.1$ proving the best. Indeed, this may simply be a parameter issue, and a suitable parameter combination of λ, $\mathcal{L}_0$ warm-up rate and number of training steps may be found heuristically with enough time and resources. Searching for an ideal $\lambda \in [0, 0.1]$ may prove more fruitful. The attention heatmaps (page 29) provide conflicting evidence: the vertical lines of attention scores and large spaces between them indicate that there is a lot of redundancy in the input. Possibly the assumption that useful features for ASR will be useful for S2UT does not hold, and a different AFS pre-training task or different loss combination may more successfully remove features redundant to the S2UT task.

Nonetheless, AFS proved necessary to tackle document-level S2UT. Without AFS the model was unable to converge, and produced a similar output regardless of the input. This may have been caused by having to reduce the batch size significantly, but this computational bottleneck is intrinsic to the document-level ST problem, one that AFS alleviates greatly.

6.3 Future work

More work is required to bring E2E ST methods closer to tackling document-level translation. Approaches such as AFS have proven extremely successful for S2T, and

the performance gap between S2T models and traditional cascaded models is rapidly closing. E2E ST paradigms like S2UT need to try and mirror the developments in S2T if they are also to close this performance gap. Two prominent challenges are the granularity of, and the lack of semantic information in the target sequence, both of which make it much more difficult to reduce the temporal dimensionality of the input, fundamental to document-level ST. One solution to this might be to perform Byte Pair Encoding (Sennrich et al., 2015) over the target discrete unit sequences, combining commonly co-occuring units together, thereby shortening the sequence. This would increase the total vocabulary size, but on average the ratio of speech features to target unit would increase, potentially allowing for greater AFS sparsity levels.

Another approach might be to incorporate attentive pooling after the S2UT encoder (Santos et al., 2016). By performing forced alignment with the source text to find word or sub-word boundaries, the source audio could be segmented with "$\langle \mathbf{s} \rangle$" tokens, which an attentive pooling layer could use to pool all encoder representations between "$\langle \mathbf{s} \rangle$" into a single representation. These representations could then be targeted with multitask training to predict source or target sub-word units, similar to how multitask learning is currently used to predict source and target characters (Lee et al., 2021). Finally, advancements in mixed-modality training (Fang et al., 2022), suggest that through joint training on text and speech data, E2E ST models can maximise mutual information between the speech and textual representations.This allows the models to leverage the large supply of textual MT data, and helps make the speech feature space richer in semantic information Fang et al. (2022). There is no reason why this success could not be directly applied in E2E ST methods like S2UT.

Chapter 7

Conclusion

This work has built upon decades of research in the field of machine translation, aiming to bring breakthroughs and improvements from textual machine translation to the relatively new field of E2E ST. Specifically, the benefits from incorporating extra-sentential information during translation are shown to be numerous in several works on textual machine translation (Post and Junczys-Dowmunt, 2023; Voita et al., 2019; Gonzales et al., 2017) and more recently on S2T (Zhang et al., 2021). The main contribution from this work has been to extend some of these benefits to a more direct, end-to-end ST paradigm, S2UT Lee et al. (2021). Additionally, this work has added evidence supporting the necessity of methods like AFS for the task of document-level ST: without reducing redundancy in the input speech features, it becomes very difficult, if not impossible, to pass S2UT input longer than a sentence.

The results from this work are mixed. Firstly it is evident that more experimentation is necessary if AFS is to be adapted to S2UT. The longer, more granular target unit sequence does not allow for the same degree of sparsity in the input, and even at much lower sparsity levels the AFS model still performed worse than the baseline on the sentence-level translation task, contrary to the findings of Zhang et al. (2020). On the other hand, AFS was crucial to the success of the document-level models: without AFS the model could not converge when given input and target sequences longer than a sentence.

As for document-level information, results clearly showed that when provided correct extra-sentential information the models learnt to use it to improve translation quality. However, with this current architecture and training regime, using inferred previous target context severely damages translation quality: only when the target prefix information is correct can the model retain functionality. Future work with S2UT

should focus on finding new ways to shorten the target sequence so AFS can be more successfully applied, or stabilising the learning process against small perturbations to parameters, data or architecture, which can currently dramatically hurt model performance.

Bibliography

Fang, Q., Ye, R., Li, L., Feng, Y., and Wang, M. (2022). Stemm: Self-learning with speech-text manifold mixup for speech translation. *arXiv preprint arXiv:2203.10426*.

Gal, Y., Hron, J., and Kendall, A. (2017). Concrete dropout. *Advances in neural information processing systems*, 30.

Gonzales, A. R., Mascarell, L., and Sennrich, R. (2017). Improving word sense disambiguation in neural machine translation with sense embeddings. In *Proceedings of the Second Conference on Machine Translation*, pages 11–19.

Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. (2021). Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460.

Inaguma, H., Popuri, S., Kulikov, I., Chen, P.-J., Wang, C., Chung, Y.-A., Tang, Y., Lee, A., Watanabe, S., and Pino, J. (2022). Unity: Two-pass direct speech-to-speech translation with discrete units. *arXiv preprint arXiv:2212.08055*.

Iranzo-Sánchez, J., Silvestre-Cerda, J. A., Jorge, J., Roselló, N., Giménez, A., Sanchis, A., Civera, J., and Juan, A. (2020). Europarl-st: A multilingual corpus for speech translation of parliamentary debates. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8229–8233. IEEE.

Jia, Y., Johnson, M., Macherey, W., Weiss, R. J., Cao, Y., Chiu, C.-C., Ari, N., Laurenzo, S., and Wu, Y. (2019a). Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7180–7184. IEEE.

Jia, Y., Ramanovich, M. T., Remez, T., and Pomerantz, R. (2022a). Translatotron 2: High-quality direct speech-to-speech translation with voice preservation. In *International Conference on Machine Learning*, pages 10120–10134. PMLR.

Jia, Y., Tadmor Ramanovich, M., Wang, Q., and Zen, H. (2022b). CVSS corpus and massively multilingual speech-to-speech translation. In *Proceedings of Language Resources and Evaluation Conference (LREC)*, pages 6691–6703.

Jia, Y., Weiss, R. J., Biadsy, F., Macherey, W., Johnson, M., Chen, Z., and Wu, Y. (2019b). Direct speech-to-speech translation with a sequence-to-sequence model. *arXiv preprint arXiv:1904.06037*.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Łańcucki, A. (2021). Fastpitch: Parallel text-to-speech with pitch prediction. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6588–6592. IEEE.

Lee, A. (2024). direct_s2st_discrete_units.md. https://github.com/facebookresearch/fairseq/blob/main/examples/speech_to_speech/docs/direct_s2st_discrete_units.md. Accessed: 2024-08-14.

Lee, A., Chen, P.-J., Wang, C., Gu, J., Popuri, S., Ma, X., Polyak, A., Adi, Y., He, Q., Tang, Y., et al. (2021). Direct speech-to-speech translation with discrete units. *arXiv preprint arXiv:2107.05604*.

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Louizos, C., Welling, M., and Kingma, D. P. (2017). Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*.

Maruf, S., Saleh, F., and Haffari, G. (2019). A survey on document-level machine translation: Methods and evaluation. *arXiv preprint arXiv:1912.08494*, 5.

nvidia (2024). fastpitch-2. https://huggingface.co/nvidia/tts_en_fastpitch. Accessed: 2024-08-13.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (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.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Popuri, S., Chen, P.-J., Wang, C., Pino, J., Adi, Y., Gu, J., Hsu, W.-N., and Lee, A. (2022). Enhanced direct speech-to-speech translation using self-supervised pre-training and data augmentation. *arXiv preprint arXiv:2204.02967*.

Post, M. (2018). A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Post, M. and Junczys-Dowmunt, M. (2023). Escaping the sentence-level paradigm in machine translation. *arXiv preprint arXiv:2304.12959*.

Post, M., Kumar, G., Lopez, A., Karakos, D., Callison-Burch, C., and Khudanpur, S. (2014). Fisher and callhome spanish–english speech translation. *LDC2014T23. Web Download. Philadelphia: Linguistic Data Consortium*.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2022). Robust speech recognition via large-scale weak supervision.

Rei, R., Stewart, C., Farinha, A. C., and Lavie, A. (2020). Comet: A neural framework for mt evaluation. *arXiv preprint arXiv:2009.09025*.

Reiter, E. (2018). A structured review of the validity of bleu. *Computational Linguistics*, 44(3):393–401.

Santos, C. d., Tan, M., Xiang, B., and Zhou, B. (2016). Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Schneider, S., Baevski, A., Collobert, R., and Auli, M. (2019). wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Shu, H. and Zhu, H. (2019). Sensitivity analysis of deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4943–4950.

Stoian, M. C., Bansal, S., and Goldwater, S. (2020). Analyzing asr pretraining for low-resource speech-to-text translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7909–7913. IEEE.

Tiedemann, J. and Scherrer, Y. (2017). Neural machine translation with extended context. *arXiv preprint arXiv:1708.05943*.

Voita, E., Sennrich, R., and Titov, I. (2019). When a good translation is wrong in context: Context-aware machine translation improves on deixis, ellipsis, and lexical cohesion. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1198–1212, Florence, Italy. Association for Computational Linguistics.

Wang, C., Pino, J., Wu, A., and Gu, J. (2020). CoVoST: A diverse multilingual speech-to-text translation corpus. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4197–4203, Marseille, France. European Language Resources Association.

Wang, C., Riviere, M., Lee, A., Wu, A., Talnikar, C., Haziza, D., Williamson, M., Pino, J., and Dupoux, E. (2021). VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. 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 993–1003, Online. Association for Computational Linguistics.

Weiss, R. J., Chorowski, J., Jaitly, N., Wu, Y., and Chen, Z. (2017). Sequence-to-sequence models can directly translate foreign speech. *arXiv preprint arXiv:1703.08581*.

Wells, D., Tang, H., and Richmond, K. (2022). Phonetic analysis of self-supervised representations of english speech. In *23rd Annual Conference of the International Speech Communication Association, INTERSPEECH 2022*, pages 3583–3587. ISCA.

Yang, S.-w., Chi, P.-H., Chuang, Y.-S., Lai, C.-I. J., Lakhotia, K., Lin, Y. Y., Liu, A. T., Shi, J., Chang, X., Lin, G.-T., et al. (2021). Superb: Speech processing universal performance benchmark. *arXiv preprint arXiv:2105.01051*.

Zhang, B., Titov, I., Haddow, B., and Sennrich, R. (2020). Adaptive feature selection for end-to-end speech translation. *arXiv preprint arXiv:2010.08518*.

Zhang, B., Titov, I., Haddow, B., and Sennrich, R. (2021). 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.