Phone Features Improve Speech Translation

Elizabeth Salesky[♈] and **Alan W Black**[♄]
[♈]Johns Hopkins University
[♄]Carnegie Mellon University
esalesky@jhu.edu, awb@cs.cmu.edu

Abstract

End-to-end models for speech translation (ST) more tightly couple speech recognition (ASR) and machine translation (MT) than a traditional cascade of separate ASR and MT models, with simpler model architectures and the potential for reduced error propagation. Their performance is often assumed to be superior, though in many conditions this is not yet the case. We compare cascaded and end-to-end models across high, medium, and low-resource conditions, and show that cascades remain stronger baselines. Further, we introduce two methods to incorporate phone features into ST models. We show that these features improve both architectures, closing the gap between end-to-end models and cascades, and outperforming previous academic work – by up to 9 BLEU on our low-resource setting.

1 Introduction

End-to-end models have become the common approach for speech translation (ST), but the performance gap between these models and a cascade of separately trained speech recognition (ASR) and machine translation (MT) remains, particularly in low-resource conditions. Models for low-resource ASR leverage phone[1] information, but this information is not typically leveraged by current sequence-to-sequence ASR or speech translation models. We propose two methods to incorporate phone features into current neural speech translation models. We explore the existing performance gap between end-to-end and cascaded models, and show that incorporating phone features not only closes this gap, but greatly improves the performance and training efficiency of both model architectures, particularly in lower-resource conditions.

The sequences of speech features used as input for ST are ≈10 times longer than the equivalent sequence of characters in e.g. a text-based MT model. This impacts memory usage, the number of model parameters, and training time. Multiple consecutive feature vectors can belong to the same phone, but the exact number depends on the phone and local context. Further, these speech features are continuously valued rather than discrete, such that a given phone will have many different instantiations across a corpus. Neural models learn to associate ranges of similarly valued feature vectors in a data-driven way, impacting performance in lower-resource conditions. Using phoneme-level information provides explicit links about local and global similarities between speech features, allowing models to learn the task at hand more efficiently and yielding greater robustness to lower-resource conditions.

We propose two simple heuristics to integrate phoneme-level information into neural speech translation models: (1) as a more robust intermediate representation in a cascade; and (2) as a concatenated embedding factor. We use the common Fisher Spanish–English dataset to compare with previous work, and simulate high-, mid-, and low-resource conditions to compare model performance across different data conditions. We compare to recent work using phone segmentation for end-to-end speech translation (Salesky et al., 2019), and show that our methods outperform this model by up to 20 BLEU on our lowest-resource condition.[2] Further, our models outperform all previous academic work on this dataset, achieving similar performance trained on 20 hours as a baseline end-to-end model trained on the full 160 hour dataset. Finally, we test model robustness by varying the quality of our phone features, which may indicate which models will better generalize across differently-resourced conditions.[3]

2 Models with Phone Supervision

We add higher-level phone features to low-level speech features to improve our models' robustness across data conditions and training efficiency. We propose two methods to incorporate phone information into cascaded and end-to-end models, depicted in Figure 1. Our *phone cascade* uses phone labels as the machine translation input, in place of the output transcription from a speech recognition model. Our *phone end-to-end* model uses

[1]The term 'phone' refers to segments corresponding to a collection of fine-grained phonetic units, but which may separate allophonic variation: see Jurafsky and Martin (2000).

[2]4-reference BLEU scores are used for this dataset.

[3]Our code is public: github.com/esalesky/xnmt-devel

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2388–2397

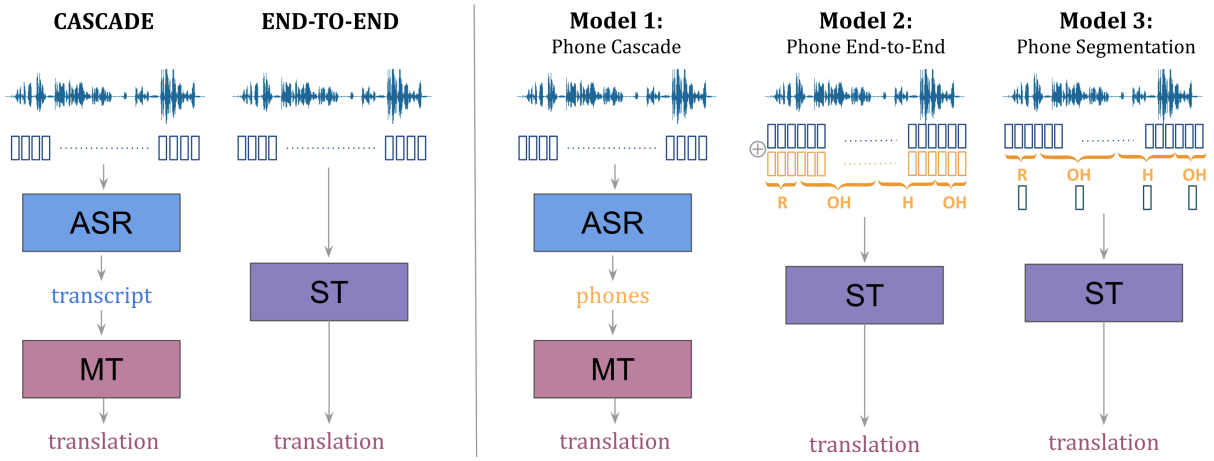

Figure 1: Comparison between traditional cascaded and end-to-end models, and our proposed methods using *phone features* as (1) the intermediate representation in a cascaded model; and (2) a concatenated embedding factor in an end-to-end model. We additionally compare to previous work; (3) where phone segmentation is used for feature vector downsampling in time (Salesky et al., 2019).

phone labels to augment source speech feature vectors in end-to-end models. We call these end-to-end or 'direct' because they utilize a single model with access to the source speech features, though they additionally use phone features generated by an external model. We additionally compare to a recent end-to-end model proposed by Salesky et al. (2019).

Model 1: Phone Cascade. In a cascade, the intermediate representation between ASR and MT is the final output of a speech recognition model, e.g. characters, subwords, or words. Using separate models for ASR and MT means that errors made in ASR are likely to propagate through MT. Common errors include substitution of phonetically similar words, or misspellings due to irregularities in a language's orthography, the latter of which may be addressed by using phone labels in place of ASR output. By not committing to orthographic targets, we believe this model will propagate fewer errors to downstream MT.

Model 2: Phone End-to-End. Our final model uses phone-factored embeddings, where trainable embeddings for phone features are concatenated to typical speech feature vector input. Because phone durations are variable and typically span more than one filterbank feature (or frame), adjacent filterbank features may have the predicted phone label; in the example shown in Figure 1, /R/ spans three frames or filterbank features. We note that this method maintains the same source sequence length as the original speech feature sequence. This method associates similar feature vectors at the corpus level, because all filterbank features with the same phone alignment (e.g. /OH/) will have the same trainable phone embedding concatenated. In MT and NER, concatenating trainable embeddings for linguistic features to words, such as morphemes and phones, has improved models' ability to generalize (Sennrich and Haddow, 2016; Chaudhary et al., 2018). While these works appended finer-grained information to associate words with similar lower-level structure, we use phone embeddings to associate higher-level structure to similar but unique speech feature vectors globally across a corpus.

Model 3: Phone Segmentation. We compare to the method from Salesky et al. (2019) as a strong end-to-end baseline. Here, phone *boundaries* are used to segment and compress speech feature vector sequences. Within each utterance, the feature vectors of consecutive speech frames with the same phone label are averaged to produce one feature vector for translation from a variable number of frames. This significantly reduces source sequence lengths (by ~80%), reducing the number of model parameters and memory. Rather than having a variable number of feature vectors per phone-like unit, each has one representation, more similar in granularity to character-based MT. The averaged feature vectors remain continuously-valued, and are locally summarized: a given phone across the corpus will still have different representations in each instance.

3 Data

We use the Fisher Spanish-English corpus,[4] which consists of parallel speech, transcripts, and translations, enabling comparisons between cascaded and direct models on the same data and allowing us to generate phone supervision using matched data. The dataset contains 160 hours of Spanish telephone speech, split into 138K utterances, which were translated via crowdsourcing by Post et al. (2013). We use the standard dev and test sets, each with ~4k utterances. Because we are particularly interested in how our methods will affect training across differently-resourced conditions, we compare results using randomly selected 40 hour and 20 hour subsets of the data.

[4] joshua.incubator.apache.org/data/fisher-callhome-corpus

4 Generating Phone Supervision

To generate phoneme-level labels for sequences of speech features, we generate frame-level alignments using a trained speech recognizer. Specifically, we extract 40-dimensional Mel filterbank features with per-speaker mean and variance normalization using Kaldi (Povey et al., 2011). We train an HMM/GMM system on the full Fisher Spanish dataset with the Kaldi recipe (Povey et al., 2011), using the Spanish CALLHOME Lexicon (LDC96L16), and compute per-frame phone alignments with the triphone model (tri3a) with LDA+MLLT features. This yields 50 phone labels, including silence (<sil>), noise, and laughter.

Producing phone alignments uses supervision from a transcript, which inherently does not exist at inference time. While phones can be extracted from Kaldi lattices at inference time, we found that our HMM/GMM model was not our best performing ASR model on this dataset – by greater than 10 WER. To leverage our better-performing neural ASR models for phone generation, we create essentially a '2-pass' alignment procedure: first, generating a transcript, and second, using this transcript to force align phones. Table 1 shows the mapping between phone quality and the ASR models used for phone feature generation. This procedure enables us to both improve phone alignment quality and also match training and inference procedures for phone generation for our translation models. In Section 8, we compare the impact of phone alignment quality on our translation models utilizing phone features, and show higher quality phone features can improve downstream results by >10 BLEU.

Alignment Quality	WER	ASR Supervision
Gold	–	*Gold transcript*
High	23.2	Salesky et al. (2019)
Med	30.4	Seq2Seq ASR
Low	35.5	Kaldi HMM/GMM

Table 1: Mapping between phone quality and the ASR models used for alignment generation, with the models' WER on Fisher Spanish test.

Producing phone features in this way uses the same data (source speech and transcripts) as the ASR task in a cascade, and auxiliary ASR tasks from multi-task end-to-end models, but as we show, to far greater effect. Further, auxiliary tasks as used in previous work rely on three-way parallel data, while it is possible to generate effective phoneme-level supervision using a recognizer trained on other corpora or languages (Salesky et al., 2019), though we do not do this here.

5 Model & Training Procedure

As in previous academic work on this corpus (Bansal et al., 2018; Sperber et al., 2019; Salesky et al., 2019), we use a sequence-to-sequence architecture inspired by Weiss et al. (2017) modified to train within lower resources; specifically, each model converges within ≈5 days on one GPU. We build encoder-decoder models with attention in `xnmt` (Neubig et al., 2018) with 512 hidden units. Our pyramidal encoder uses 3-layer BiLSTMs with linear network-in-network (NiN) projections and batch normalization between layers (Sperber et al., 2019; Zhang et al., 2017). The NiN projections are used to downsample by a factor of 2 between layers, resulting in the same total 4× downsampling in time as the additional convolutional layers from Weiss et al. (2017); Bansal et al. (2019): They give us the benefit of added depth with fewer additional parameters. We use single layer MLP attention (Bahdanau et al., 2015) with 128 units and 1 decoder layer as opposed to 3 or 4 in previous work – we did not see consistent benefits from additional depth.

In line with previous work on this dataset, all experiments preprocess target text by lowercasing and removing punctuation aside from apostrophes. We use 40-dimensional Mel filterbank features as previous work did not see significant difference with higher-dimensional features (Salesky et al., 2019). We use 1k BPE units for translation text, shown in Salesky et al. (2019) to have both better performance and training efficiency than characters (Weiss et al., 2017; Sperber et al., 2019) or words (Bansal et al., 2018). For both text and phones, we use 64-dimensional embeddings.

For the MT component in cascaded speech translation models, we compared using the pyramidal speech architecture above (3 encoder, 1 decoder layers) to the traditional BiLSTM text model (2 layers each for encoder and decoder). Using the pyramidal architecture resulted in the same performance as the BiLSTM model when translating BPE transcriptions from ASR, but gave us consistent improvements of up to 1.5 BLEU when instead translating phone sequences; we posit this is because phone sequences are longer than BPE equivalents. Accordingly, we use the same model architecture for all our ASR, MT, and ST models.

We use layer dropout with $p = 0.2$ and target embedding dropout with $p = 0.1$ (Gal and Ghahramani, 2016). We apply label smoothing with $p = 0.1$ (Szegedy et al., 2016) and fix the target embedding norm to 1 (Nguyen and Chiang, 2018). For inference, we use beam of size 15 and length normalization with exponent 1.5. We set the batch size dynamically depending on the input sequence length with average batch size was 36. We use Adam (Kingma and Ba, 2015) with initial learning rate 0.0003, decayed by 0.5 when validation BLEU did not improve for 10 epochs initially and subsequently 5 epochs. We do not use L2 weight decay or Gaussian noise, and use a single model replica. We use input feeding (Luong et al., 2015), and exclude utterances longer than 1500 frames in training for memory.

6 Prior Work: Cascaded vs End-to-End Models on Fisher Spanish-English

The large body of research on the Fisher Spanish-English dataset, including both cascaded and end-to-end models, makes it a good benchmark to compare these architectures. Not all previous work has compared across multiple resource settings or compared to cascaded models, which we address in this section. We summarize best previous results on this dataset on high, medium, and low-resource conditions in Table 2.

Best Results. The cascade of traditional HMM/DNN ASR and Joshua MT models from Kumar et al. (2014) set a competitive baseline on the full dataset (40.4 test BLEU) which no subsequent academic models have been able to match until this work; subsequent exploration of end-to-end models has produced notable relative improvements but the best end-to-end academic number (Salesky et al., 2019) remains 1.6 BLEU behind this traditional cascade.

Industry models from Weiss et al. (2017) achieved exceptional performance with very deep end-to-end models on the full dataset (47.3 test BLEU), exceeding a cascade for the first time. They additionally show results with an updated cascade using neural models, improving over Kumar et al. (2014). Their results have been previously unmet by the rest of the community. This is likely in part due to the computational resources required to fully explore training schedules and hyperparameters with models of their depth. While their ASR models took ~4 days to converge, their ST models took another 2 weeks, compared to the lighter-weight models of recent academic work which converged in <5 days (Sperber et al., 2019; Salesky et al., 2019; Bansal et al., 2019).

This dataset is challenging: improving ASR WER from 35 (Post et al.) to 23 (Kumar et al.) only resulted in 4 BLEU ST improvement: see *Components* in Table 2. We believe this to be in part because the multi-reference scoring masks some model differences, and the conversational phenomena (like disfluencies) are challenging.

Lower-Resource. While deep end-to-end models have become competitive at higher-resource conditions, previous work on this dataset has showed they are not as data-efficient as cascades under lower-resource conditions. While some works have tested multiple resource conditions, only Sperber et al. (2019) compared against cascades across multiple conditions. Their end-to-end baseline outperformed their cascades on the full dataset, but not under lower-resource conditions, while their end-to-end but multi-stage attention-passing model is more data-efficient than previous models and shows the best previous results under lower-resource condition. Sperber et al. do not report results without auxiliary ASR, MT, and autoencoding tasks, which they state add up to 2 BLEU.

Additional Data. Stoian et al. (2020); Bansal et al. (2019); Sperber et al. (2019) investigate speech translation performance using additional corpora through transfer learning from ASR and auxiliary MT tasks. The ability to leverage non-parallel corpora was previously a strength of cascades and had not been explored with end-to-end models. We do not use additional data here, but show these numbers as context for our results with phone supervision, and refer readers to Sperber et al. for discussion of cascaded and end-to-end models' capacity to make use of more data.

Parameter Tuning. We find cascaded model performance can be impacted significantly by model settings such as beam size and choice of ASR target preprocessing. While Weiss et al. (2017); Sperber et al. (2019) use character targets for ASR, we use BPE, which gave us an average increase of 2 BLEU. Further, we note that search space in decoding has significant impact on cascaded model performance. In cascaded models, errors produced by ASR can be unrecoverable, as the MT component has access only to ASR output. While Sperber et al. (2019) use a beam of size 1 for the ASR component of their cascade to compare with their two-stage end-to-

		HIGH (160hr)		**MID** (40hr)		**LOW** (20hr)		*Components*	
Model	**Source**	dev	test	dev	test	dev	test	*ASR* ↓	*MT* ↑
Cascaded	Weiss et al. (2017)	45.1	45.5	–	–	–	–	*23.2*	*57.9*
	Kumar et al. (2014)	–	**40.4**†	–	–	–	–	*25.3*	*62.9*
	Sperber et al. (2019)	–	32.5	–	16.8	–	6.6	*40.9*	*58.1*
End-to-End	Weiss et al. (2017)	46.5	**47.3**∗	–	–	–	–		
	Salesky et al. (2019)	37.6	38.8	21.0	19.8	11.1	10.0		
	Sperber et al. (2019)	–	36.7	–	**31.9**	–	**22.8**		
	Stoian et al. (2020)	34.1	34.6	–	–	10.3	10.2		
+ *Add'l Data*	Sperber et al. (2019)	–	38.8	–	–	–	–		
	Stoian et al. (2020)	37.9	37.8	–	–	20.1	20.2		

Table 2: End-to-end vs cascaded speech translation model performance in BLEU↑ on Fisher Spanish-English data from the literature. (†) denotes the best previous academic result on the full dataset, (∗) the best from industry. Component models for cascades reported on test on full dataset: ASR reported in WER↓ and MT in BLEU↑.

end models, we find that using equal beam sizes of 15 for both ASR and MT improves cascaded performance with the same model by 4-8 BLEU; combining these two parameter changes makes the same cascaded model a much more competitive baseline (compare lines 3 in both Table 2 and Table 3). In contrast, widening beam size to yield an equivalent search space for end-to-end models has diminishing returns after a certain point; we did not see further benefits with a larger beam (> 15).

Our Baselines. We report best numbers from previous work in Table 2 for comparison (which may use multi-task training), but use single-task models in our work. We report our baseline results in Table 3. On the full dataset, our baseline cascade improves slightly over Kumar et al. (2014) with 41.0 compared to 40.4 on test, a mark most recent work has not matched primarily due to model choices noted above, with component ASR performance of WER 30.4 and 58.6 BLEU for MT. Our end-to-end baseline is comparable to the baselines in Salesky et al. (2019); Sperber et al. (2019); Stoian et al. (2020). This suggests we have competitive baselines for both end-to-end and cascaded models.

7 Results Using Phone Features

We compare our two ways to leverage phone features to our cascaded and end-to-end baselines across three resource conditions. Table 3 shows our results; following previous work, all BLEU scores are multi-reference. Average single reference scores may be found in Appendix A. All models using phone supervision outperform the end-to-end baseline on all three resource conditions, while our proposed models also exceed the cascaded baseline and previous work at lower-resource conditions.

Phone features. Salesky et al. (2019) performs most similarly to the end-to-end baseline, but nonetheless represents an average relative improvement of 13% across the three data sizes with a significant reduction in training time. Our phone featured models use not just the phone segmentation, but also the phone labels, and perform significantly better. Our **phone end-to-end** model not only shows less of a decrease in performance across

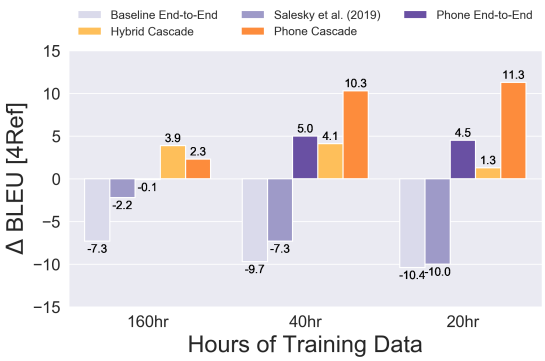

Figure 2: Performance of all models relative to '*Baseline Cascade*' (Δ = 0) across our 3 resource conditions. **Cascaded** models in **orange**, **end-to-end** models in **purple**. Our proposed models yield improvements across all three conditions, with a widening margin under low-resource conditions for the phone cascade.

resource conditions than Salesky et al. (2019), but further improves by 4 BLEU over the baseline cascade on our two lower-resource conditions. This suggests augmenting embeddings with discrete phone features is more effective than improved downsampling. The **phone cascade** performs still better, with marked improvements across all conditions over all other models (see Figure 2). On the full dataset, using phones as the source for MT in a cascade performs ~2 BLEU better than using BPE, while at 40 and 20 hours this increases to up to 10 BLEU. We analyze the robustness of phone models further in Section 8.

Hybrid cascade. We additionally use a 'hybrid cascade' model to compare using phone features to improving ASR. Our hybrid cascade uses an ASR model with phone-informed downsampling and BPE targets (Salesky et al., 2019). This improves the WER of our ASR model to 28.1 on dev and 23.2 on test, matching Weiss et al. (2017)'s state-of-the-art on test (23.2) and approaching it on dev (25.7). Our hybrid cascade performs more similarly to Weiss et al.'s cascade on the full dataset, with 45.0 to their 45.5 on test, and is our best-performing ST model on the full dataset. However, at lower-resource conditions, it does not perform as favor-

		HIGH (160hr)			**MID** (40hr)			**LOW** (20hr)		
	Model	dev	test	Δ	dev	test	Δ	dev	test	Δ
Baseline	Baseline End-to-End	32.4	33.7	–	19.5	17.4	–	9.8	9.8	–
	Salesky et al. (2019)	37.6	38.8	**+5.2**	21.0	19.8	**+2.0**	11.1	10.0	**+0.8**
	Baseline Cascade	39.7	41.0	**+7.3**	29.8	27.1	**+10.0**	22.6	20.2	**+11.6**
Proposed	Phone End-to-End	40.5	42.1	**+8.3**	34.5	33.0	**+15.3**	26.7	26.2	**+16.7**
	Phone Cascade	41.6	43.3	**+9.4**	**37.2**	**37.4**	**+18.9**	**32.2**	**31.5**	**+22.1**
	Hybrid Cascade	**42.9**	**45.0**	**+10.9**	33.3	31.2	**+13.8**	23.2	21.5	**+12.6**

Table 3: Results in BLEU↑ comparing our proposed phone featured models to baselines. We compare three resource conditions, and show average improvement for dev and test (Δ). Best performance bolded by column.

ably compared to phone featured models – as shown in Figure 2, both the phone cascade and phone end-to-end models outperform the hybrid cascade at lower-resource conditions, by up to 10 BLEU at 20 hours. This suggests improving ASR may enable cascades to perform better at high-resource conditions, but under lower-resource conditions it is not as effective as utilizing phone features.

Training time. In addition to performance improvements, our models with phone features are typically more efficient with respect to training time, shown in Table 4. The fixed time to produce phone labels, which must be performed before translation, becomes a greater proportion of overall training time at lower-resource settings. In particular, the phone end-to-end model offers similar training time reduction over the baseline to Salesky et al. (2019), where downsampling reduces sequence lengths by up to 60%, with unreduced sequence lengths through earlier convergence; this model offers a better trade-off between time and performance.

Model	HIGH	MID	LOW	Δ
Baseline End-to-End	118hr	40hr	22hr	–
Salesky et al. (2019)	41hr	13hr	10hr	0.4×
Baseline Cascade	76hr	19hr	12hr	0.6×
Phone Cascade	57hr	39hr	27hr	0.7×
Phone End-to-End	42hr	20hr	13hr	0.4×
Hybrid Cascade	47hr	34hr	24hr	0.6×

Table 4: Total training time for all models (including time to generate phone features) on 3 resource conditions. The ASR and MT models in the baseline cascade can be trained in parallel, reflected here, while phone featured models may not as the MT requires phone features from ASR.

Comparing to previous work using additional data. Previous work used the parallel speech transcripts in this dataset for auxiliary tasks with gains of up to 2 BLEU; we show using the same data to generate phone supervision is far more effective. We note that our phone models further outperform previous work trained with additional corpora. The attention-passing model of Sperber et al. (2019) trained on additional parallel Spanish-English text yields 38.8 on test on the full dataset, which Salesky et al. (2019) matches on the full dataset and our proposed models exceed, with the phone cascade yielding a similar result (37.4) trained on only 40 hours. Pre-training with 300 hours of English ASR data and fine-tuning on 20 hours of Spanish-English data, Stoian et al. (2020); Bansal et al. (2019) improve their end-to-end models from ≈10 BLEU to 20.2. All three of our proposed models exceed this mark trained on 20 hours of Fisher.

8 Model Robustness & Further Analysis

In this section, we analyze the robustness of each of our models by varying the quality of our phone features, and further explore the strengths and limitations of each model.

8.1 Phone Cascade

Phone cascades use a representation for translation which may be more robust to non-phonetic aspects of orthography. However, as a cascaded model, this still requires hard decisions between ASR and MT, and so we may expect lower phone quality to lead to unrecoverable errors. Figure 3 compares the impact of phone quality on the performance of phone cascades trained on our high, medium, and low-resource conditions. We use alignments produced with gold transcripts as an upper bound on performance. We note that with gold alignments, translation performance is similar to text-based translation (see Section 6). We see that phone quality does have a significant impact on performance, with the MT model trained on low phone quality yielding similar translation performance using the full 160 hour dataset to the MT model with the highest quality phones trained on only 20 hours. However, we also see significantly more data-efficiency with this model, with less reduction in performance between $160hr \rightarrow 40hr \rightarrow 20hr$ training conditions than previous models.

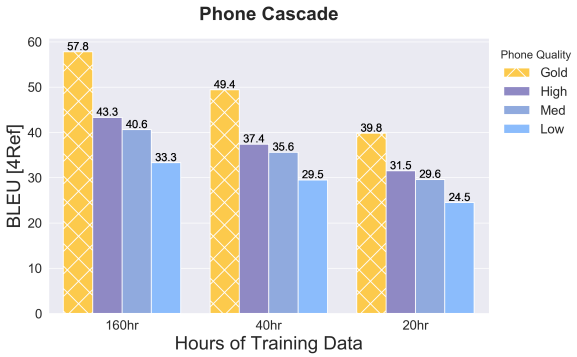

Figure 3: **Phone Cascade Robustness**: using phone labels in place of BPE as the text source for downstream MT. Comparing performance across our three data conditions and phone label qualities.

Redundancy. For the phone cascade models compared in Figure 3, we collapse adjacent consecutive phones with the same label, i.e. when three consecutive frames have been aligned to the same phone label *'B B B'* we have reduced the sequence to a single phone *'B'* for translation. We additionally compared translating non-uniqued phone sequences (e.g. the same sequence length as the number of frames) as a more controlled proxy for our model's handling of longer frame-based feature vector sequences compared to Salesky et al. (2019)'s downsampled feature vector sequences. The redundant phones caused consistent decreases in BLEU,

with much greater impact in lower-resource conditions. Translating the full sequence of redundant frame-level phone labels, for the full 160hr dataset, all models performed on average 0.6 BLEU worse; for 40hr, 1.8 BLEU worse; and with 20 hours, 4.1 BLEU worse – a 13% decrease in performance *solely from non-uniqued sequences*.

Phones correspond to a variable-length number of speech frames depending on context, speaker, and other semantic information. When translating speech feature vectors, speech features within a phone are similar but uniquely valued; using instead phone labels in a phone cascade, the labels are identical though still redundant. These results suggest our LSTM-based models are better able to handle redundancy and variable phone length at higher resource conditions with sufficient examples, but are less able to handle redundancy with less training data.

8.2 Phone End-to-End

Our phone end-to-end model concatenates trainable embeddings for phone labels to frame-level filterbank features, associating similar feature vectors *globally* across the corpus, as opposed to *locally* within an utterance as with the phone-averaged embeddings (Section 8.3). Figure 4 compares the results of these factored models using phone features of differing qualities, with 'gold' alignments as an upper bound. The phone end-to-end models compared do not reach the same upper performance as the phone cascades: comparing gold phone labels, the phone end-to-end model performs slightly worse at 160hr with more degradation in performance at 40hr and 20hr. While this comparison is even more pronounced for 'low' phone quality than 'gold,' the phone end-to-end model has more similar performance between 'gold' and 'high' phone quality than the cascade.

This model's input contains both the phone features used in the phone cascade and speech features of the baseline end-to-end model, but unlike the phone cascade or Salesky et al. (2019) the input sequence has not been reduced in length. That the end-to-end phone model achieves top performance and converges much faster than end-to-end baseline is unsurprising, as access to both speech feature vectors and phone labels mitigates the effects of long noisy input sequences. The significant performance improvements over Salesky et al. (2019), however, are more interesting, as these models make use of the similar information in different ways – the use of discrete embeddings seems to aid the phone end-to-end model, though the sequence length is not reduced. The model's performance degradation compared to the phone cascade in lower-resource conditions is likely due in part to these sequence lengths, as shown by our additional experiments with input redundancy for the cascade. The greater reduction in performance here using lower quality phones suggests the noise of the labels and concatenated filterbank features compound, further detracting from performance. Perhaps further investigation into the relative weights placed on the two embedding factors over the training process could close this additional gap.

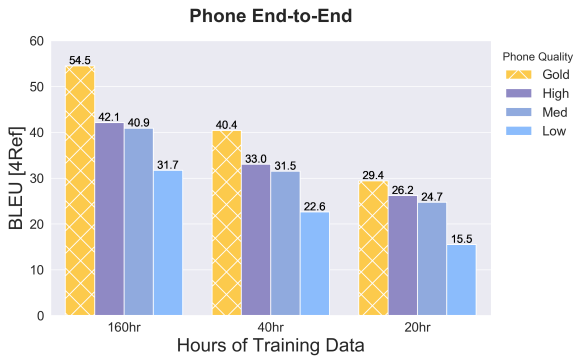

Figure 4: **Phone End-to-End Robustness**: trainable embeddings for phone labels are concatenated to frame-level filterbank features. Comparing performance across three data conditions and phone label qualities.

8.3 Phone Segmentation: Salesky et al. (2019)

We also compare to the models from Salesky et al. (2019) as a strong end-to-end baseline. That work introduced downsampling informed by phone segmentation – unlike our other models, the *value* of the phone label is not used, but rather, phone alignments are used only to determine the *boundary* between adjacent phones for variable-length downsampling. Their model provides considerable training and decoding time improvements due to the reduced source sequence length, and shows consistent improvements over the baseline end-to-end model using the original filterbank feature sequences which increase with the amount of training data. However, their model has lower overall performance and with much smaller performance improvements over our baselines in lower-resource conditions than the phone featured models we propose here. We hypothesize that the primary reason for their BLEU improvements is the reduction in local redundancy between similar frames, as discovered in the previous section. We refer readers to their paper for further analysis.

8.4 Quality of Phone Labels

We show two examples of phone sequences produced with each overall model quality in Figure 5, uniqued within consecutive frame sequences with the same label for space constraints. Individual phones are typically 5-20 frames. We see the primary difference in produced phones between different models is the label values, rather than the boundaries. While we do see some cases where the boundaries shift, they chiefly vary by only 1-3 frames. It is not the case that there are significantly more or fewer phone segments aligned per utterance by quality, though there are outlying utterances (Example 2 – 'Low').

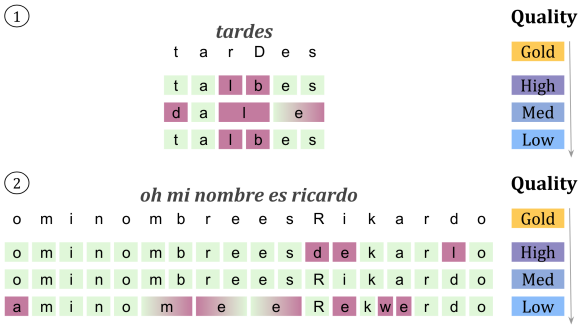

Figure 5: Two examples of phone sequences demonstrating differences across qualities of phone features. (See Table 1 for the mapping between quality and generation procedure). Note: word-level segmentation is not marked, as it is also not present in {*speech,phone*} source sequences for translation.

Relating our observed trends to the differences between our phone cascades and phone end-to-end models, we note that differences in frame-level phone boundaries would not affect our phone cascaded models, where the speech features are discarded, while they would affect our phone end-to-end models, where the phone labels are concatenated to speech feature vectors and associate them across the corpus. While errors in phone labels may be seen as 'unrecoverable' in a cascade, for the end-to-end model, they add noise to distribution of filterbank feature associated with each phone label embedding, which appears to have a more negative impact on performance than the hard decisions in cascades. Though the concatenated filterbank features may allow our end-to-end models to recover from discrete label errors, our results testing various phone qualities suggest this may only be the case under higher-resource settings with sufficient examples.

9 Related Work

Speech translation was initially performed by cascading separately trained ASR and MT models, allowing each model to be trained on larger data sources without parallel speech, transcriptions, and translations, but potentially yielding unrecoverable errors between models. Linking models through lattices with both phrase-based (Kumar et al., 2014) and neural MT (Sperber et al., 2017) reduced many such errors. Using one model to directly translate speech was later enabled by attentional encoder-decoder models.

Direct end-to-end speech translation was first explored as a way to reduce both error propagation, and also the need for high quality intermediate transcriptions (e.g. for unwritten languages). The first such models were investigated in Bérard et al. (2016); Duong et al. (2016), but these used, respectively, a small synthetic corpus and evaluated on speech-to-text alignments rather than translation. Subsequently Weiss et al. (2017) extended these neural attentional models to deep, multitask models with excellent results on Fisher Spanish–English, exceeding a cascade for the first time. However, efforts from the community have not yet replicated their success (Stoian et al., 2020; Sperber et al., 2019; Salesky et al., 2019). End-to-end models have performed inconsistently compared to cascades on other corpora: Bérard et al. (2018) perform well on high-resource audiobooks but do not exceed a cascade; Anastasopoulos and Chiang (2018) found 'triangle' models performed better than cascades for 2 of 3 very low-resource language pairs; and in the most recent IWSLT evaluation campaigns, cascades have remained the highest-performing systems (Niehues et al., 2018, 2019).

Similarly-motivated work exists in speech translation. In addition to Salesky et al. (2019); Sperber et al. (2019) addressed above, preliminary cascades using phone-like units have been explored for low-resource speech translation, motivated by translation of unwritten languages where a traditional cascade would not be possible. To this end, Bansal et al. (2018) utilized unsupervised term discovery, and Wilkinson et al. (2016) synthesized speech; but these approaches were only evaluated in terms of precision and recall and were not tested on both 'higher-resource' and natural speech data conditions.

10 Conclusion

We show that phone features significantly improve the performance and data efficiency of neural speech translation models. We study the existing performance gap between cascaded and end-to-end models, and introduce two methods to use phoneme-level features in both architectures. Our improvements hold across high, medium, and low-resource conditions. Our greatest improvements are seen in our lowest-resource settings *(20 hours)*, where our end-to-end model outperforms a strong baseline cascade by ≈5 BLEU, and our cascade outperforms prior work by ≈9 BLEU. Generating phone features uses the same data as auxiliary speech recognition tasks from prior work; our experiments suggest these features are a more effective use of this data, with our models matching the performance from previous works' performance without additional training data. We hope that these model comparisons and results inform development of more robust end-to-end models, and provide a stronger benchmark for performance on low-resource settings.

Acknowledgments

The authors thank Andrew Runge, Carlos Aguirre, Carol Edwards, Eleanor Chodroff, Florian's cluster, Huda Khayrallah, Matthew Wiesner, Nikolai Vogler, Rachel Wicks, Ryan Cotterell, and the anonymous reviewers for helpful feedback and resources.

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. *Proc. of NAACL*. arXiv:1802.06655.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *Proc. of ICLR*. arXiv:1409.0473.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2018. Low-resource speech-to-text translation. *Proc. of Interspeech*. arXiv:1803.09164.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2019. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. *Proc. of NAACL*. arXiv:1809.01431.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *Proc. of ICASSP*.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. *NIPS Workshop on End-to-end Learning for Speech and Audio Processing*. arXiv:1612.01744.

Aditi Chaudhary, Chunting Zhou, Lori Levin, Graham Neubig, David R Mortensen, and Jaime G Carbonell. 2018. Adapting word embeddings to new languages with morphological and phonological subword representations. *Proc. of EMNLP*. arXiv:1808.09500.

Long Duong, Antonios Anastasopoulos, David Chiang, Steven Bird, and Trevor Cohn. 2016. An attentional model for speech translation without transcription. In *Proc. of NAACL*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. *Proc. of NeurIPS*.

D Jurafsky and J Martin. 2000. *Speech and Language Processing*, 3rd edition. Prentice Hall.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *Proc. of ICLR*. arXiv:1412.6980.

Gaurav Kumar, Matt Post, Daniel Povey, and Sanjeev Khudanpur. 2014. Some insights from translating conversational telephone speech. *Proc. of ICASSP*.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. *Proc. of EMNLP*.

Graham Neubig, Matthias Sperber, Xinyi Wang, Matthieu Felix, Austin Matthews, Sarguna Padmanabhan, Ye Qi, Devendra Singh Sachan, Philip Arthur, Pierre Godard, et al. 2018. XNMT: The eXtensible neural machine translation toolkit. *Proc. of AMTA*. arXiv:1803.00188.

Toan Q Nguyen and David Chiang. 2018. Improving lexical choice in neural machine translation. *Proc. of NAACL*. arXiv:1710.01329.

Jan Niehues, Ronaldo Cattoni, Sebastian Stüker, Mauro Cettolo, Marco Turchi, and Marcello Federico. 2018. The iwslt 2018 evaluation campaign.

Jan Niehues, Ronaldo Cattoni, Sebastian Stüker, Matteo Negri, Marco Turchi, Thanh-Le Ha, Elizabeth Salesky, Ramon Sanabria, Loïc Barrault, Lucia Specia, and Marcello Federico. 2019. The iwslt 2019 evaluation campaign.

Matt Post, Gaurav Kumar, Adam Lopez, Damianos Karakos, Chris Callison-Burch, and Sanjeev Khudanpur. 2013. Improved speech-to-text translation with the Fisher and Callhome Spanish–English speech translation corpus.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. 2011. The Kaldi speech recognition toolkit. *Proc. of ASRU*.

Elizabeth Salesky, Matthias Sperber, and Alan Black. 2019. Exploring phoneme-level speech representations for end-to-end speech translation. *Proc. of ACL*.

Rico Sennrich and Barry Haddow. 2016. Linguistic input features improve neural machine translation. *Proc. of WMT*. arXiv:1606.02892.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2017. Neural lattice-to-sequence models for uncertain inputs. *Proc. of EMNLP*. arXiv:1704.00559.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2019. Attention-passing models for robust and data-efficient end-to-end speech translation. *Proc. of TACL*. arXiv:1904.07209.

Mihaela C Stoian, Sameer Bansal, and Sharon Goldwater. 2020. Analyzing ASR pretraining for low-resource speech-to-text translation. *Proc. of ICASSP*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. *Proc. of CVPR*.

Ron J Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly transcribe foreign speech. *Proc. of INTERSPEECH*. arXiv:1703.08581.

Andrew Wilkinson, Tiancheng Zhao, and Alan W Black. 2016. Deriving phonetic transcriptions and discovering word segmentations for speech-to-speech translation in low-resource settings. *Proc. of INTERSPEECH*.

Yu Zhang, William Chan, and Navdeep Jaitly. 2017. Very deep convolutional networks for end-to-end speech recognition. *Proc. of ICASSP*.

A Single-Reference BLEU Scores

These tables contain the same results as our tables and figures as in the main paper, but show *average single-reference* BLEU scores in place of *multi-reference (4-reference)* BLEU. WER for ASR is unchanged: the dataset contains a single reference transcript for ASR. Results from prior work report only multi-reference BLEU and so are not included below.

	ASR↓		MT↑		Cascade		End-to-End	
Data	**dev**	**test**	**dev**	**test**	**dev**	**test**	**dev**	**test**
Full	33.3	30.4	34.5	33.6	23.2	23.7	19.0	19.6
40hr	44.8	46.7	29.9	28.3	17.4	15.7	11.5	10.4
20hr	56.3	59.1	22.4	22.6	13.2	11.8	5.9	5.3

Table 8: **Baseline results** for end-to-end and cascaded speech translation models, with component ASR and MT model performance for cascades (blue). ASR results in WER↓ and translation results in BLEU↑.

Phone Quality	**160hr**		**40hr**		**20hr**	
	dev	test	dev	test	dev	test
Gold	33.3	33.2	29.3	28.5	24.4	23.0
High	24.1	25.1	21.6	21.7	18.9	18.3
Med	23.1	23.4	20.6	20.7	17.6	17.2
Low	18.2	19.1	16.4	17.0	14.1	14.2

Table 5: **Phone Cascades**. We use frame-level phone labels as the text source for downstream MT. Comparing method robustness to phone quality and resource conditions.

Phone Quality	**160hr**		**40hr**		**20hr**	
	dev	test	dev	test	dev	test
Gold	34.1	31.3	27.9	23.4	20.5	17.2
Med	24.0	23.7	20.8	18.4	16.5	14.6
Low	20.5	18.3	17.0	13.0	12.2	8.7

Table 6: **Phone End-to-End**. Trainable embeddings for phone labels are concatenated to frame-level filterbank features. Comparing method robustness to phone quality and resource conditions.

		Full (160hr)			**40hr**			**20hr**		
	Model	dev	test	Δ	dev	test	Δ	dev	test	Δ
Baseline	Baseline End-to-End	19.0	19.6	–	11.5	10.4	–	5.9	5.3	–
	Salesky et al. (2019)	22.0	21.9	**+2.7**	12.6	11.6	**+1.2**	6.7	6.2	**+0.9**
	Baseline Cascade	23.2	23.7	**+4.2**	17.4	15.7	**+5.6**	13.2	11.8	**+6.9**
Proposed	Phone End-to-End	24.0	23.7	**+4.6**	20.8	18.4	**+8.7**	16.5	14.6	**+10.0**
	Phone Cascade	24.1	25.1	**+5.3**	**21.6**	**21.7**	**+10.7**	**18.9**	**18.3**	**+13.0**
	Hybrid Cascade	**24.9**	**25.9**	**+6.1**	19.6	18.2	**+8.0**	13.6	12.6	**+7.5**

Table 7: Results in BLEU↑ comparing our proposed phone featured models to baselines. We compare three resource conditions, and show average improvement for dev and test (Δ). Best performance bolded by column.