Abstract

One challenge in speech translation is that plenty of spoken content is long-form, but short units are necessary for obtaining high-quality translations. To address this mismatch, we adapt large language models (LLMs) to split long ASR transcripts into segments that can be independently translated so as to maximize the overall translation quality. We overcome the tendency of hallucination in LLMs by incorporating finite-state constraints during decoding; these eliminate invalid outputs without requiring additional training. We discover that LLMs are adaptable to transcripts containing ASR errors through prompt-tuning or fine-tuning. Relative to a state-of-the-art automatic punctuation baseline, our best LLM improves the average BLEU by 2.9 points for English–German, English–Spanish, and English–Arabic TED talk translation in 9 test sets, just by improving segmentation.

1 Introduction

With the proliferation of long-form audiovisual content online, translation and captioning become paramount for accessibility. Cascade models remain the dominant approach for speech translation (Arivazhagan et al., 2020; Li et al., 2021), decomposing the problem into automatic speech recognition (ASR), post-processing of the transcript, and machine translation (MT).

The cascade’s MT component typically operates on sentence-like units, with each sentence translated independently of the others. When asked to translate long passages, models regularly fail or degenerate (Cho et al., 2014; Pouget-Abadie et al., 2014; Koehn and Knowles, 2017). This differs considerably from the expectations for automatic speech recognition models (e.g. Graves, 2012) that can process inputs of unbounded lengths. MT models must either be able to cope with potentially long, multi-sentence inputs or, alternatively, they must be able to determine cutpoints at which the transcript can be segmented into compact, independently translatable units. This work introduces a new, effective approach for the latter.

While numerous text segmentation techniques have been proposed to improve spoken language translation (§6), the problem remains hard and unsolved. Indeed, Li et al. (Li et al., 2021) demonstrate that poor sentence segmentation degrades performance almost twice as much as transcript lexical errors.

We cast sentence segmentation as a sequence-to-sequence task, rather than a traditional structured prediction task that tags sentence-final tokens. While this lets us leverage large language models, such models’ outputs can be ill-formed. Even by using additional data for fine-tuning, residual adapters (Tomanek et al., 2021; Chronopoulou et al., 2022), or future discriminators (Yang and Klein, 2021), simple syntactic constraints can be difficult to enforce. Moreover, all three require modifying the model or storing additional learned parameters. In light of these concerns, we introduce a simple, flexible, and modular approach to generating well-formed task-specific strings at inference time without any additional training. We compactly express constraints on the output format as finite-state machines, then efficiently enforce these via composition. While the approach is simple, it remains unexplored for large language models, and it yields automatic gains on downstream performance, advancing the state of the art for speech translation and thereby applicable to existing systems. Moreover, the approach is sufficiently general that it can be applied to other domains in a plug-and-play manner.

We benchmark our approach as a component in a speech translation cascade. Experiments in three language pairs indicate that our approach outperforms both a baseline cascade system that predicts punctuation marks before inferring sentence boundaries and a strong neural structured prediction
model. Overall, we improve the BLEU score on the IWSLT test sets by 2.9 points, closing 3/4 of the gap between the previous best and the oracle system. Our contributions are three-fold:

1. We propose a novel LLM-based approach for long-form speech translation, which can be applied to any ASR-MT speech translation cascade system and yield a significant increase in translation quality.

2. To the best of our knowledge, we are the first to investigate the use of finite-state decoding constraints in combination with LLMs to produce consistent improvements.

3. We report additional small but consistent improvements by prompt-tuning or fine-tuning LLMs on ASR transcripts containing lexical and grammatical errors.

2 Windowing Approach

One major challenge in modeling and inference of long-form transcript segmentation is that the input sequences can be very long. For example, a TED talk can contain more than one thousand words (Li et al., 2021). We take a divide-and-conquer approach that operationalizes two straightforward principles in modeling. First, words on the left and right are both useful for deciding if a sentence delimiter should be present at the current word position. Second, distant words are less useful than nearby words. From these two principles, we design a top-level sliding window algorithm to balance the need for bidirectional modeling and efficiency of computation. We divide the passage into windows at both training and test time, with a small context window on each side to inform decisions at window edges (Figure 1). With this top-level inference algorithm, the sequence-to-sequence machine learning problem is now reduced to the window-level. The problem is now to predict a sequence of segmentation decisions \( y = y_1, \ldots, y_w \) for each text window of size at most \( w \) tokens: \( x = x_1, \ldots, x_w \).

3 Modeling Approaches

A classic approach to discriminative sequence modeling is the conditional random field (CRF) (Lafferty et al., 2001; Liu et al., 2005). This conditional graphical model allows incorporating arbitrary features of the transcript, including linguistic variables and word embeddings.

3.1 Structured Prediction Baseline: Bidirectional RNN Model

The limitation of the CRF is in the Markov assumption it makes, considering only the immediately previous word’s segmentation decision. Even higher-order CRFs can only consider a fixed-size history within \( y \). Instead, we introduce a neural autoregressive segmenter. It is an encoder–decoder neural network with monotonic hard attention to the bidirectionally encoded input at the current word position, admitting the same rich featurization of \( x \) as the CRF; its likelihood is

\[
p_\theta(y | x) = \prod_{t=1}^{w} p_\theta(y_t | y_{<t}, x) \quad (1)
\]

\[
: = \prod_{t=1}^{w} p_\theta(y_t | y_{<t}, \text{BiRNN}(x)_t) \quad (2)
\]

where \( p_\theta \) is parameterized by a recurrent neural network followed by a linear projection layer and a softmax to obtain a locally normalized distribution. Exact inference here is intractable (unlike a CRF); we approximate it with beam search. This model and a QRNN-based (Bradbury et al., 2017) automatic punctuation model will serve as baselines.

3.2 Large Language Models for Segmentation

More recently, the paradigm of pre-training followed by fine-tuning or few-shot learning has achieved great successes across many NLP tasks. The pre-training task is typically a variant of a language model (Brown et al., 2020; Chowdhery et al., 2022) or an autoencoder (Raffel et al., 2020) where a corrupted version of a sentence is mapped to its uncorrupted counterpart. We can encode segmentation as such a task: reproducing the input with inserted sentence delimiters. Concretely, we encode \( y = y_1, \ldots, y_w \), where \( y_t = \text{concat}(d_t, x_t) \) and \( d_t \in \{\epsilon, \text{■}\} \). For example, we feed \( \text{i am hungry} \) \( \text{i am sleepy} \) to the model, and it produces the sentence-delimited string \( \text{i am hungry■i am sleepy} \). We use the publicly available T5 (Text-to-Text Transfer Transformer) model (Raffel et al., 2020) and the GPT-style (Brown et al., 2020) PaLM model (Chowdhery et al., 2022) as the foundation for our text-based segmenters.

3.2.1 Prompting and Fine-tuning

Training examples for this task look like the input output pairs in Figure 2. In fine-tuning, we update the full set of parameters for a given model on
Figure 1: Processing overlapping windows instead of entire transcript passages. $w$ is the window size used in both training and inference. $b$ is the total context window size. $r$ ($\leq b$) is the right context window size. The underlines below the windows indicate which local segmentation decisions are taken as global decisions. Portions not underlined (i.e., the context window) are still provided to the segmentation model to inform segmentation of underlined portions.

$$[t_0, \ldots, t_{w-b}, \ldots, t_{w-r}, \ldots, t_{w-1}]$$
$$[t_{w-b}, \ldots, t_{w-r}, \ldots, t_{w-2b}, \ldots, t_{2w-b-r}, \ldots, t_{2w-b-1}]$$
$$[t_{2w-b}, \ldots, t_{2w-b-r}, \ldots, t_{3w-3b}, \ldots, t_{3w-2b-r}, \ldots, t_{3w-2b-1}]$$
$$\vdots$$
$$[\ldots, \ldots, t_{n-1}]$$

$t_0, \ldots$ $t_{w-1}$

Figure 2: Prompting PaLM to segment a text window (red) based on three examples.

<table>
<thead>
<tr>
<th>Input</th>
<th>Segmented Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>well first of all thank you so much that is a beautiful compliment i do think he is the best interviewer alive</td>
<td>well first of all thank you so much &lt;SENT&gt; that is a beautiful compliment &lt;SENT&gt; i do think he is the best interviewer alive</td>
</tr>
<tr>
<td>i remember when my dad had to leave our home in scranton pennsylvania to find work i grew up in a family where if the price of food went up you felt it that's why one of the first things i did as president is to pass the american rescue plan</td>
<td>i remember when my dad had to leave our home in scranton pennsylvania to find work &lt;SENT&gt; i grew up in a family where if the price of food went up you felt it &lt;SENT&gt; that's why one of the first things i did as president is to pass the american rescue plan</td>
</tr>
<tr>
<td>we're done talking about infrastructure weeks we're going to have an infrastructure decade</td>
<td>we're done talking about infrastructure weeks we're going to have an infrastructure decade</td>
</tr>
<tr>
<td>i remember when my dad had to leave our home in scranton pennsylvania to find work</td>
<td>i remember when my dad had to leave our home in scranton pennsylvania to find work</td>
</tr>
<tr>
<td>it is going to rain today remember to bring an umbrella</td>
<td>it is going to rain today remember to bring an umbrella</td>
</tr>
</tbody>
</table>

Levenshtein Alignment for Post-processing

The generation models’ ability to produce arbitrary outputs may be seen as a strength: the model could correct transcription errors and remove disfluencies, if so trained. Therefore, we can let the model generate freely without enforcing structural constraints, then enforce well-formedness post-hoc. **Kumar and Byrne (2002)** describe a WFST for Levenshtein alignment between two strings. We use it to align the generated string with $x$. We then project segment boundaries across alignment links from the generated string onto $x$ to determine $y$. In this way, annotations can be salvaged when LLM does not precisely recreate the input.

Finite-State Constraints in Decoding

A natural strategy to force well-formed outputs is constrained decoding (e.g. Zhang et al., 2019). In it, we compose the input FSA $x$ and a special FST $T$ encoding all possible segmentation decisions, then project the FST to the output tape to obtain a determinized FSA for the output space. The FST $x \circ T$ is shown in Figure 3.

An advantage of the finite-state approach is that any constraint expressible as a regular language is possible. Consequently, our implemented system is applicable a large class of tagging and parsing problems in NLP, not just sentence segmentation. For instance, NP chunking (Ramshaw and Marcus, 1995) and BIO tagging, truecasing (Lita et al., 2003), retokenization, tetra-tagging for syntactic
parsing (Kitaev and Klein, 2020), and lexically constrained decoding (Hasler et al., 2018) can all be framed as finite-state transformations of an input sequence.

4 Experiments

We evaluate our proposed method for using large language models for long-form speech translation with three sets of experiments: (1) analysis of hyperparameters, (2) comparison with competing methods, and (3) robustness to speech recognition errors. In each case, we are concerned with translation quality as measured by BLEU. We also assess the LLM output directly by qualitative analysis, well-formedness percentage, and (for diagnostic purposes, following Goldwater et al., 2009) segmentation $F_1$ score against the sentence-segmented reference.

Our experiments are carried out on the IWSLT speech translation data sets, subjected to the same pre-processing as described in Li et al. (2021). We use the 2014 data for dev and 2015 and 2018 for test. The fourteen reference transcripts in our dev set range from 861 to 1234 words; by contrast, the median length of a sentence in written English is close to 17 words (Kučera and Francis, 1970). We use the publicly available Speech-to-Text Google API\(^1\) to generate ASR transcripts. We remove the automatically predicted punctuation and lowercase the ASR transcripts and use English–{German,Spanish,Arabic} MT models trained with the same preprocessing on the source side as Li et al., 2021. The MT model is a Transformer with a model dimension of 1024, hidden size of 8192, 16 attention heads, 6 encoder layers, and 8 decoder layers. We decode with a beam size of 4. In our experiments, the three MT model instances and the ASR model (and thereby its transcripts) are fixed while we vary the sentence segmentation policies.

4.1 Context Window Size

In §2, we introduced the top-level sliding window inference algorithm above all modeling choices. To compare different models fairly, we fix the hyperparameters $(w, b, r) = (40, 10, 5)$ for the algorithm throughout the experiments. This choice is guided by a linear search over the window lengths $w$ in the range of $[20, 100]$. The overlapping buffer size for both ends is set to 5 based on findings of segmentation for punctuated text (Wicks and Post, 2021). According to Figure 4, translation quality degrades slightly as window size approaches 20. But very large windows do not appear to be beneficial. The observation validates the two guiding principles of our sliding window approach.

4.2 Choice of Prompt

The manual prompt in Figure 2 is the one we selected from a few variants for the decoder-only PaLM models. Instead of exploring the unbounded space of prompts, we resorted to the more principled method of prompt tuning (Lester et al., 2021) to optimize the prompt in the embedding space for the segmentation task. For prompt tuning, the only hyperparameter is the length of the embedding prompt (the embedding size is tied to the corresponding model). In Figure 5, we show that for the PaLM models of 62B and 540B, an embedded prompt as short as 10 can achieve much higher $F_1$ than our hand-written prompt. But it is also notable that the gap between prompt tuning and manual prompting shrank from 25 percent to 10 percent as the model size increased from 62B to 540B, indicating the increasingly stronger generalization capability of extremely large language models. Based on Figure 5, we use 30-token soft

\(^1\)https://cloud.google.com/speech-to-text

Figure 3: FST representing all possible segmentations for the transcript “i came i saw i conquered”.

Figure 4: BLEU for English–German as context window size for segmentation increases. Each dot represents a T5 segmentation model trained with the same window size for inference time.
4.3 Effect of Finite-State Constraints

We make a contrast between greedy search and beam search, with either the segmentation FST constraint §3.2.2 inside the decoder or post-hoc Levenshtein alignment §3.2.2 for repairing invalid output. We also vary the model types and model sizes to analyze the impact of constrained decoding in different situations. Table 1 shows that constraints are crucial for smaller models in prompt-tuned scenarios. For example, the rate of output being well-formed is only 14.5% using greedy search for the PaLM 8B model. Even when the model size is increased to 62B, the well-formedness rate is still below 90%. The Levenshtein post-alignment algorithm is effective. But the more general finite-state constraint is even more effective. For the 8B model, the improvement in \( F_1 \) is 1–2% absolute. For the 62B model, the improvement is nearly 3% absolute. On the other hand, if the cost of fine-tuning is acceptable, LLMs can adapt to this task very well. The fine-tuned T5 base model has a well-formedness rate of 99.4% (the rate is even higher for the T5 11B model: 99.8%). But we shall point out that for the results to be useful to downstream applications, either of the two types of constraints is necessary to completely eliminate hallucinations from LLMs. And the FST constraints are more general and more effective as they affect beam search by rejecting non-well-formed hypotheses during search.

4.4 Main Results: LLMs against Structured Prediction Models

Using the IWSLT TED datasets as preprocessed by Li et al. (2021), we compare LLM models against their approach, two strong custom structured prediction baselines. We also report the performance of an oracle segmenter.

<table>
<thead>
<tr>
<th>MODEL</th>
<th>CONSTRAINT</th>
<th>SEARCH</th>
<th>WELLFORMED</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>T5 BASE</td>
<td>UNCONSTRAINED</td>
<td>GREEDY</td>
<td>99.4%</td>
<td>–</td>
</tr>
<tr>
<td>FINE TUNED</td>
<td>LEVENSHTEIN</td>
<td>GREEDY</td>
<td>100.0%</td>
<td>0.786</td>
</tr>
<tr>
<td></td>
<td>FST</td>
<td>GREEDY</td>
<td>100.0%</td>
<td>0.786</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>99.4%</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>100.0%</td>
<td>0.788</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>100.0%</td>
<td>0.788</td>
</tr>
<tr>
<td>PALM 8B  PROMPT TUNED</td>
<td>UNCONSTRAINED</td>
<td>GREEDY</td>
<td>14.5%</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>52.7%</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>LEVENSHTEIN</td>
<td>GREEDY</td>
<td>100.0%</td>
<td>0.715</td>
</tr>
<tr>
<td></td>
<td>FST</td>
<td>GREEDY</td>
<td>100.0%</td>
<td>0.717</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>100.0%</td>
<td>0.727</td>
</tr>
<tr>
<td>PALM 62B  PROMPT TUNED</td>
<td>UNCONSTRAINED</td>
<td>GREEDY</td>
<td>85.9%</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>89.0%</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>LEVENSHTEIN</td>
<td>GREEDY</td>
<td>100.0%</td>
<td>0.735</td>
</tr>
<tr>
<td></td>
<td>FST</td>
<td>GREEDY</td>
<td>100.0%</td>
<td>0.737</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>100.0%</td>
<td>0.761</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BEAM=4</td>
<td>100.0%</td>
<td>0.764</td>
</tr>
</tbody>
</table>

Table 1: Effect of finite-state decoding constraints and Levenshtein post alignment on segmentation \( F_1 \).

**FIXEDLENGTH** Separates the transcript into disjoint segments with the same number of tokens. While this requires no external segmentation model, the resulting segments are non-sentential (Tsiamas et al., 2022).

**ORACLE** Uses punctuation from the reference transcripts to segment. The segmentation is projected onto Levenshtein-aligned words in the noisy ASR transcripts (§3.2.2).

**PUNCTUATE** An interpretable two-pass segmentation that first infers punctuation (Soboleva et al., 2021), then uses a fixed set of inference rules to differentiate sentence-terminal punctuation marks from sentence-internal ones as in “St. John” and “The end.”

**biGRU F.T.** On the IWSLT data, fine-tunes a shallow biGRU model (§3.1) trained on the C4 data set (Raffel et al., 2020) using the same rules in PUNCTUATE to derive sentence boundaries as supervision. The model has 1 left-to-right GRU layer, 1 right-to-left GRU layer, and 1 GRU layer in the decoder. It uses embeddings of character \( n \)-gram projections (Zhang et al., 2019).

**T5-{BASE,11B}** Fine-tunes the base or 11B (xxl) T5 model (Raffel et al., 2020) on the IWSLT data.

---

2 A true oracle would optimize corpus-level BLEU over all 2\(^n\) segmentations, but this is intractable.
T5-11B-ASR: Fine-tunes the 11B T5 model on the ASR output of IWSLT train and dev set. Sentence boundaries are projected from reference transcripts in the same way as ORACLE.

**PALM-PROMPT_TUNED-62B{-62B,540B}{,-ASR}**
Prompt-tunes the PaLM model (Chowdhery et al., 2022) on the IWSLT data.


The peer-reviewed state of the art for long-form speech translation is Li et al. (2021) on the IWSLT data set for EN→DE. Compared to ORACLE, there is still a large gap of 3 BLEU points which can be closed by improving segmentation alone.

Table 2 lists the complete set of results. BiGRU F.T. already beats Li et al. (2021) by more than 1 BLEU point for EN→DE, proving itself as a strong structured prediction baseline. T5 and PaLM models improve the results furthermore. Within the T5 group, T5-11B improves over T5-BASE by 3% in segmentation $F_1$ which translates to consistent BLEU score improvement in almost all data sets. Within the PaLM group, the prompt-tuned 540B model is about 5% more accurate than the 62B counterpart. Given the large number of parameters, fine-tuning PaLM models is very expensive. For the completeness of comparison, we include the fine-tune result for PaLM 62B. Its result is on-par with the T5 11B model. This fact indicates that T5’s encoder–decoder architecture has an inductive bias advantage over the PaLM model’s decoder only architecture for this task, from a parameter efficiency point of view. But the strength of the PaLM family lies in its largest member. The 540B model with a tiny tuned prompt is as effective as the fully fine-tuned T5 11B or PaLM 62B.

### 4.5 Robustness to Speech Recognition Errors

One key difference between cascade speech translation and typical document-level translation is that transcription errors can be introduced, which propagate into the translation. When the input to segmentation models contains speech recognition errors, can such models still predict sentence boundaries accurately? The answer is yes, to a certain extent. To test this, we replace the tuning data from ground-truth transcripts with punctuation-derived sentence boundaries to ASR transcripts that have sentence boundaries projected from their parallel ground-truth transcript counterparts. For example, we will tune the models to predict the segmentation for the passage: *this train leaves at for <SENT> the next train will arrive in ten minutes*, even though there is a lexical error (*for versus four*).

Table 2 shows that training on the ASR transcripts is indeed beneficial. On top of the strong results of the T5 11B model trained on ground-truth transcripts, the ASR version obtains another 1% $F_1$ improvement. The same is true for the PaLM 62B prompt-tuned and fine-tuned models. The relative
improvement is consistent across different prompt sizes and fine-tuning methods (Figure 6). Still, the small segmentation improvement does not translate into significant BLEU score improvements.

5 Error Analysis

5.1 Segment Length Histogram Analysis
To understand the improvements and the remaining errors, we first compare the length distribution of the ORACLE, the small model BiGRU, T5-11B, and PALM-540B. Figure 7 indicates that the more very long (≥ 50) segments a model has, the lower its $F_1$ and BLEU scores tend to be. Both LLM models were able to reduce the number of very long segments, bringing it closer to the oracle.

5.2 Qualitative Analysis
Table 3 shows examples where the T5-11B-ASR model outperforms competing models. In the first two examples, the LLM model is able to capture the larger context and therefore make the correct prediction. The third example typifies the cases where T5-11B, which is fine-tuned on ground-truth transcripts without ASR errors, tends to make more wrong predictions when the input text is not fluent.

Table 4 shows typical errors the T5-11B-ASR model makes. In the first two, ASR errors make the transcript difficult to parse. The third one is linguistically ambiguous. In the last one, the model’s prediction is actually closer to the ground-truth segmentation than the Levenshtein-(mis)aligned ASR transcript.

Overall, LLMs such as T5-11B-ASR made real progress in predictions requiring longer context. However, even though fine-tuning on ASR transcripts improved robustness to disfluent input, overcoming ASR errors remains challenging.

6 Related Work

Speech translation. While end-to-end systems for speech translation have exceeded the performance of cascade models on short sequences (Weiss et al., 2017) even on public data (McCarthy et al., 2020), long-form audio is typically translated with cascades. Previous work uses tagging approaches to separate text into independently translatable units. Segmenting long texts into units suitable for translation has been a recurring topic in MT research (Li et al., 2021; Tien and Thi Minh, 2019; Pouget-Abadie et al., 2014; Doi and Sumita, 2003; Goh and Sumita, 2011). To bridge the gap between ASR and MT, Li et al. (2021) address long-form speech translation. Claiming that segmentation is the bottleneck, they adapt their MT model to work with automatic segmentations, however inaccurate they may be.

We are training our models to minimize the loss of source sentence segmentation. The ultimate objective is improving the downstream translation quality. It is interesting to explore reinforcement learning for segmentation (Srinivasan and Dyer, 2021), but the state space is vast for the long-form segmentation problem compared to prior work on RL-based segmentation.

Finally, one may consider additional sources of data or training examples to improve modeling. Using prosodic features when they are available is viable (Tsiamas et al., 2022); however, we show that LLMs close most of the accuracy gap without these. As a contrasting approach, Kumar and Byrne (2002) focus on segmenting an ASR lattice, rather than the decoded transcript. Finally, data augmentation (Pino et al., 2019; McCarthy et al., 2020; Li et al., 2021) can complement our approach.
Reference: this great renaissance for ancient Egyptian art, architecture, and religion. Egyptologists have always known the site.

BigRU: this great renaissance for ancient Egyptian art, architecture, and religion. Egyptologists have always known the site.

T5-11B-ASR: this great renaissance for ancient Egyptian art, architecture, and religion. Egyptologists have always known the site.

Reference: looking for layers of human occupation and five meters down underneath a thick layer of mud, we found a dense layer of pottery.

BigRU: looking for layers of human occupation and five meters down underneath a thick layer of mud, we found a dense layer of pottery.

T5-11B-ASR: looking for layers of human occupation and five meters down underneath a thick layer of mud, we found a dense layer of pottery.

Table 3: Cases where the T5-11B-ASR model’s prediction is wrong.

Reference: how actually actually started in 1984 BC at a not lost for long city found from above.

BigRU: how actually actually started in 1984 BC at a not lost for long city found from above.

T5-11B: how <SENT> actually actually started in 1984 BC at a not lost for long city found from above.

Table 4: Cases where the T5-11B-ASR model’s prediction is wrong.

Text normalization and segmentation. Mansfield et al. (2019) model text normalization as a sequence-to-sequence problem, using <self> tags to bias toward copying, but they place no search constraints to ensure well-formedness. Zhang et al. (2019) also use finite automata intersected with a neurally generated lattice during decoding.

Wicks and Post (2021) provide a unified solution for segmenting punctuated text in many languages; however, ground-truth punctuation is not present in speech recognition output.

Structured prediction as sequence-to-sequence. Vinyals et al. (2015) show that attention-enhanced sequence-to-sequence models can be trained for complex structured prediction tasks such as syntactic parsing. Raffel et al. (2020) take a step further to model all text-based language problems in a text-to-text format. Paolini et al. (2021) framed many NLP tasks as translation between augmented natural languages.


7 Conclusion

We have presented new methods for long-form speech translation by coupling source-side large language models with finite-state decoding constraints, allowing large language models to be used for structured prediction with a guarantee for well-formedness in the output space. Finite-state constraints are especially effective when the model is decoder-only, relatively small, or has not been completely fine-tuned (only prompt-tuned, or few-shot-learned) for the structured prediction task. We also observe that even though complete fine-tuning and enlarging model size can reduce the rate of invalid output, models alone are not capable of completely eliminate invalid output.

Fine-tuning on in-domain ASR transcripts containing recognition errors and disfluency improves segmentation accuracy over training on clean transcripts. Our qualitative analysis shows the largest category of remaining errors is ASR errors which make transcripts difficult to parse and segment. The fact that LLMs are capable of adapting to ASR errors points to future research directions of contextualized ASR error recovery.
Acknowledgments

We thank Chu-Cheng Lin and Nicholas Tomlin for comments that improved the presentation of the work. A.D.M. is supported by an Amazon Fellowship and a Frederick Jelinek Fellowship.

Limitations

Large language models are more expensive and slower compared to dedicated smaller models for sentence segmentation. The additional latency introduced in the speech-to-text cascade by such models can be too high for online processing.

We use a sliding window algorithm to combine segmentation outputs from adjacent fixed-size text windows. The choice is a sub-optimal heuristic efficiency-accuracy tradeoff. Recently, large language models have become increasingly capable of handling long paragraphs. We can simplify the system by applying large language models directly on long paragraphs. However, as the output length increases, the likelihood of hallucinations increases, making decoding constraints more important. Moreover, there may always be long-form audio whose transcriptions exceed the context length of even the largest language models.

Finally, adhering to the cascade architecture—speech-recognition followed by text-to-text translation—introduces the problem of error propagation. Our error analysis has shown that speech recognition errors form the main category within the remaining errors made by our systems. Can text-only large language models systematically correct speech recognition errors without introducing hallucinations? Furthermore, can a speech recognition model that incorporates a large language model jointly recognize and segment the transcription better than a cascade system?

References


