A Hybrid Seq-2-Seq ASR Design for On-Device and Server Applications

Cyril Allauzen, Ehsan Variani, Michael Riley, David Rybach, Hao Zhang

Google Research, United States
{allauzen, variani, riley, rybach, haozhang}@google.com

Abstract
This paper proposes and evaluates alternative speech recognition design strategies using the hybrid autoregressive transducer (HAT) model. The different strategies are designed with special attention to the choice of modeling units and to the integration of different types of external language models during first-pass beam-search or second-pass re-scoring. These approaches are compared on a large-scale voice search task and the recognition quality over the head and tail of speech data is analyzed. Our experiments show decent improvements in WER over conventional models with weighted finite state transducers (WFSTs) [3, 4].

1. Introduction
Automatic speech recognition (ASR) is becoming the predominant interface for users of smart assistant devices. While the early deployments of practical ASR systems were dominated by server-based conventional models [1, 2, 3, 4], the growing applications and recent developments in the chip design have led to many on-device ASR modeling paradigms based on the sequence-to-sequence (Seq-2-Seq) models [5, 6, 7, 8]. This diversity of applications and the myriad constraints imposed by the platform design, whether server [9, 10] or on-device [11], requires a deeper look into ASR modeling design.

There are several important design criteria for ASR modeling. One is the choice of training data. While the recognition quality usually improves with the amount of in-domain transcribed training data, the collection of huge amounts of transcribed data for different domains is costly and difficult due to privacy concerns. Another challenge with speech data is the long tail distribution of uncommon phrases. Models should be designed so that skewed or limited training data does not overly impact the quality on infrequently seen tail phrases.

Inference flexibility, by having principled ways to modify and adapt models after training, is another critical factor for a large-scale ASR design. Languages grow over time due to changes in culture, fashion, etc. Some of these changes are so sudden it is not possible to collect new data and retrain the model for these new words or phrases in a timely manner.

The computation load and memory footprint are yet other critical parameters in ASR design. The computation benefits of recently developed machine-learning based chips are usually apparent when they perform the same operation in batch. This is especially true when compared to the state-of-the-art approaches.

Index Terms: speech recognition, modularity, sequence-to-sequence, tail distribution

1. Hybrid autoregressive transducers
For an acoustic feature sequence \( x = x_1 \ldots x_T \) corresponding to a word sequence \( w \), assume \( y = y_1 \ldots y_U \) is a tokenization of \( w \) where \( y_i \in \mathcal{M} \) is either a phonetic unit or a character-based unit from a finite-size alphabet \( \mathcal{M} \). Since usually \( T \neq U \), a notation of alignment is defined between elements of \( x \) and \( y \). The alignment sequence \( \hat{y} \) can be defined as a sequence of \( T + U \) labels, where label \( \hat{y}_{i+U+1} \) is either equal to blank symbol \(<b>\) (suggesting consumption of time frame \( x_{i+1} \)) or is equal to \( y_{i+1} \). The HAT model formulates the local posterior distribution \( P(\hat{y}_{i+u} | x; \hat{y}_{1:i+u-1}) \) by a Bernoulli distribution with parameter \( b_{i,u} \) and a label distribution \( P_{i,u} \), as follows:

\[
\begin{align*}
    P_{i,u}(y_1, y_{i+1}) &= b_{i,u} P_{i,u}(y_{i+1}) + (1 - b_{i,u}) P_{i,u}(y_1 | x, y_{1:i-1}) \\
    \hat{y}_{i+U+1} &= <b> \\
    \hat{y}_{i+u} &= y_u
\end{align*}
\]
The HAT model does not provide any strict parametric form for neither \( b_{y,u} \) nor \( P_{1,u} \). This means that these distributions can be modeled by different neural architectures with or without sharing parameters. By chaining the local posterior probabilities over an alignment path, the alignment posterior \( P(y|x) \) is derived. The posterior probability \( y \) given \( x \) is then modeled by summing all the alignment posteriors:

\[
P(y|x) = \sum_{\tilde{y} \in D(y|x)} P(y|x) \tag{1}
\]

where \( B : \tilde{y} \rightarrow y \) is the function that maps alignment paths to their corresponding label sequence. In addition to modeling the posterior probability, the HAT model provides an estimate of the prior, or internal language model (ILM) probability, for any sequence \( y \) [8]:

\[
P_{ILM}(y) = \prod_{i} P_{1,u}(y_{\mu_i}, 0; y_{1..u-1})
\]

which is the chain of label distribution \( P_{1,u} \) over labels, assuming the encoder activations are zero. Using this quantity and Bayes’ rule, a pseudo-likelihood sequence-level score [16, 17, 18, 19] is derived:

\[
P_{RLM}(y|w) = \sum_{y \in ILM} P_{RLM}(y) \cdot \frac{P(y|x)}{P_{ILM}(y)}
\]

which divides each logit value by a constant before applying the softmax. It is clear that integrating the above score values over the alignment paths visited during the search leads to the total score for \( y \) in Eq. 2 excluding the external LM score and restricting the sum from Eq. 1 to visited alignment paths. The external LM score is computed separately by storing an additional LM state for each partial hypothesis and LM lookahead is performed when using a word-based external LM. All the inference parameters, \( \lambda_1 \), \( \lambda_2 \) and the smoothing parameters can be estimated by sweeping on a held out set.

3.2. Second-pass language model rescoring

The first-pass decoder generates a set of hypothesis, denoted by \( \mathbb{H}{x} \). These hypothesis can be rescoring using a large-scale word-based rescoring LM (RLM) as follow:

\[
\arg \max_{y \in \mathbb{H}{x}} \left( \log P(y|x) - \lambda_1 \log P_{ILM}(y) + \log P_{RLM}(w) \right)
\]

where \( \mu_1 \) and \( \mu_2 \) are scaling parameters that can be estimated by sweeping on a held out set. Observe that proper LM rescoring can also be applied in the absence of external LM in the first pass by replacing \( P_{RLM}(w) \) by \( \lambda_2 P_{ILM}(y) \) in Eq. 3 and Eq. 2. This also implies that the rescoring LM can be used instead of the (first-pass) external LM to disambiguate when the mapping from unit sequences to word sequence is not one-to-one. The advantage of adding external LM in the second-pass also opens a mean to exploit the benefits of accelerator chips via batching. The same process, rescoring with external LM, is parallelized over the hypothesis space \( \mathbb{H}{x} \). This is particularly useful feature for the on-device applications.

3.3. Modeling units and decoder graph

In order to match the HAT modeling units (phonemes or graphemes) with those of the external LM (written-domain words), the ASR decoder uses a decoder graph, a weighted finite-state transducer (WFST) whose input tape is over the HAT modeling units and output tape over words. The construction of this decoder graph varies slightly depending on the exact choice of modeling units: written graphemes, where \( y \) is the sequence of characters in the written-domain transcript, e.g. “meet at 5:10”; spoken graphemes, where \( y \) is the sequence of characters in the spoken-domain transcript, e.g. “meet at five ten”; and phonemes, where \( y \) is the sequence of phonemes in the phone

\[
\text{transcript, e.g. "a i t t \{ t s l f a l v v t E n".}
\]

Our word-based first-pass LM is a written-domain class-based \( n \)-gram model [20, 21]. Numeric classes are used to expand numerical entity coverage while using a closed written-domain vocabulary. We will consider two types of numeric class grammars, spoken-domain and written-domain, both represented as finite-state transducers. When using spoken-domain class grammars, the decoder output will contain class instances in the spoken domain, e.g. "\texttt{\textit{xbox 360 <m>} fifty dollar 〈/m> game}". A denormalization post-processing step will be applied to then convert to the desired written-domain form, e.g. "\texttt{\textit{\textit{\textit{xbox 360 \textit{S}50} game}}". This denormalization is performed by applying class-specific finite-state grammars. When using written-domain class grammars, the decoder output will be in the writ-

<table>
<thead>
<tr>
<th>HAT modeling unit</th>
<th>Decoder graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>phoneme</td>
<td>( T_{p} = L_{p} \circ V \circ \text{Replace}(C_{p}, y) )</td>
</tr>
<tr>
<td>spoken grapheme</td>
<td>( T_{w} = L_{w} \circ V \circ \text{Replace}(C_{w}, y) )</td>
</tr>
</tbody>
</table>

Table 1: Decoder graph construction from the following component WFSTs: \( L_{p} \), \( L_{w} \), phonemic, spoken and written graphemic lexicon; \( V \), verbalizer; \( G \), top-level \( n \)-gram LM; \( C_{p} \), \( C_{w} \), spoken and written-domain class grammars.
ten domain but as a sequence of single-character words that needs to be concatenated together. e.g. ’xbox 360 <cm> $ 5 0 <cm> game’. A verbalizer [22] is used to convert words in the written vocabulary into sequences of words in the spoken vocabulary. Spoken and written domain graphemic lexicons implement the mapping from grapheme sequences to words in the spoken and written vocabulary. A phonemic lexicon maps from phoneme sequences to sequences of words in the spoken vocabulary. It is derived from a pronunciation dictionary containing both human-generated and/or human-verified pronunciations, as well as pronunciations generated by a G2P model.

All these components are represented as WFSTs. Table 1 gives the decoder graph construction for each choice of HAT modeling unit, where composition and replacement are performed as described in [23, 24]. Language model lookahead is performed as part of the decoder graph construction [25].

4. Experiments

The 40M utterance training set (30k hours of speech), 8K utterance development set (10 hours), and 25-hour test set are all anonymized, hand-transcribed representatives of Google spoken-queries traffic. The training examples are 256-dim. log Mel features extracted from a 64 ms window every 30 ms [26]. The training examples are noisified with 25 different noise styles as detailed in [27]. Each training example is forced-aligned to get the frame-level phoneme alignment used to derive reference labels for training the phoneme-based and spoken-grapheme-based models. Models are trained to predict either 42 phonemes or 75 graphemes. The model architecture for RNN-T and HAT models are explained in [8].

The first-pass word-level LM is a 5-gram model with a 4M-word vocabulary trained on anonymized audio transcriptions and web documents. The second-pass word-level rescoring LM is a large-scale maximum entropy LM [28] trained on the same data and covering the same vocabulary. It is applied using an additional lattice re-scoring pass and the corresponding hyper-parameters are swept on a separate development set. The grapheme-based recurrent neural network (RNN) LM is trained on the same data and has 4 LSTM layers (2048 cells per layer).

For the evaluation, we examine several test sets to gauge performance on both the head and tail of the query distribution. The spoken queries test set consists of about fourteen thousands anonymized utterances from Google spoken queries traffic. A thousand of these utterances were deemed more difficult to recognize and were selected to form the hard spoken queries test set. These utterances appear to be from the tail of the query distribution. To get a deeper insight into the long tail, roughly twenty thousand queries from the LM training data were identified as tail queries based on several criteria and were synthesized using a Google TTS system to form the TTS long-tail queries test set [29].

4.1. External LM and modeling units

The written grapheme case offers the most design options for integration with an external language model: an internal LM treating HAT as a drop-in RNN-T replacement with shallow-fusion style second-pass LM rescoring (B0), an internal LM but also leveraging it in the second pass for proper LM rescoring (W0), word-based 5-gram LM (W1) or a grapheme-based RNN LM (W2). Results in Table 2 show that leveraging HAT only in the second pass already brings significant gains versus RNN-T (B0 vs W0) with even more gains achieved when using an external LM in the first pass. These gains are even stronger on the hard spoken queries and long-tail TTS test sets.

For spoken graphemes and phonemes, LM choices are more limited since we currently rely on the decoder graph for the conversion from modeling units to words, although it would be possible to rely on the second-pass rescoring LM for that too. Phoneme HAT (P1) performs only slightly worse than the written-grapheme HAT systems on spoken queries after second-pass rescoring but significantly outperforms them on the hard spoken queries and long-tail TTS queries. We suspect that this is because the written grapheme systems better handle the text normalization issues affecting WER scoring in the spoken queries test set (Section 4.2) whereas the phoneme HAT benefits from the use of pronunciation dictionaries when handling rare words (Section 4.3). Spoken grapheme HAT (S1) performance is in-between phoneme and written graphemes HAT systems.

Finally note that proper second-pass rescoring with the large-scale MaxEnt model is extremely effective, significantly reducing the gaps observed in the first pass for the various systems on the full spoken queries test set.

4.2. Written-domain WER and text normalization

We suspect that the WER performance gap between the written-grapheme systems and spoken systems are mostly due to our written-domain scoring and transcription conventions. For instance, the spoken utterance a two hour window is transcribed by the human labelers as 2-hour window and recognized as such by the written grapheme systems, whereas the spoken-domain systems prefers a 2 hour window which counts as two errors (one substitution and one insertion). One hypothesis is that since the written-grapheme Seq-2-Seq models are trained directly on human transcriptions, they can learn the transcription conventions. This can lead to a spurious WER gain that does not translate into better recognition quality.

To validate this hypothesis, we trained a neural denormalization model that can replace the spoken-to-written denorm finite-state grammars used in the systems with spoken numeric classes in their decoder output (phonemes and spoken graphemes). By training on transcribed utterances, such a denorm model should be able to learn transcription conventions to some extent. We model the denorm problem as a two-level transduction task [30]. The input to the model is the spoken-domain text. The first-level of transduction identifies text spans that require edits. In the example above, the text span 2 hour will be identified and the rest of the text will be copied to the output. In the second level of the model, the identified text spans along with a fixed-sized vector of embedded sentence context are fed to another component to generate the written-domain output. In this particular case, 2-hour will be the output. The entire model is trained end-to-end using pairs of the written-domain human-labeled transcript and a spoken-domain transcript obtained from force alignment. This denorm model is then applied on the 1-best hypothesis after MaxEnt LM rescoring. Results in Table 2 (S1 vs S2 and P1 vs P2) support our assumption that the written-domain systems are better at learning the transcription conventions: using the neural denormer brings significant improvements on the full spoken queries test set where transcription conventions play a key role and slight regressions on the TTS test set where these conventions are irrelevant. In future work, we will consider CER and spoken-domain WER to complement written-domain WER and provide better insights into the performance differences between these models.

4.3. Pronunciations and alternate spellings

One hypothesis to explain the better performance of the phoneme-based systems on the hard spoken queries and long-
Table 2: WER (first-pass, 10-best oracle and second-pass after MaxEnt LM rescoring) for RNN-T and HAT models on all test sets.

<table>
<thead>
<tr>
<th>WER</th>
<th>Spoken Queries All</th>
<th>Spoken Queries Hard</th>
<th>TTS Long-tail Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-T</td>
<td>6.9 (1.6)</td>
<td>6.0</td>
<td>40.0 (22.1)</td>
</tr>
<tr>
<td>W1</td>
<td>6.0 (2.1)</td>
<td>5.8</td>
<td>33.5 (17.4)</td>
</tr>
<tr>
<td>W2</td>
<td>6.4 (1.3)</td>
<td>6.0</td>
<td>26.8 (15.6)</td>
</tr>
<tr>
<td>S1</td>
<td>6.3 (1.7)</td>
<td>6.0</td>
<td>26.8 (14.1)</td>
</tr>
<tr>
<td>P1</td>
<td>6.8 (1.6)</td>
<td>6.2</td>
<td>22.6 (10.2)</td>
</tr>
<tr>
<td>S2</td>
<td>6.1 (1.6)</td>
<td>5.8</td>
<td>26.0 (14.0)</td>
</tr>
<tr>
<td>P2</td>
<td>6.5 (1.4)</td>
<td>6.0</td>
<td>22.9 (8.5)</td>
</tr>
<tr>
<td>W3</td>
<td>6.1 (2.0)</td>
<td>5.9</td>
<td>26.4 (15.6)</td>
</tr>
<tr>
<td>S3</td>
<td>6.2 (1.6)</td>
<td>5.9</td>
<td>25.7 (13.9)</td>
</tr>
<tr>
<td>S4</td>
<td>6.1 (1.5)</td>
<td>5.8</td>
<td>25.9 (10.5)</td>
</tr>
</tbody>
</table>

Table 3: WER (first-pass and second-pass after MaxEnt LM rescoring) on Spoken Queries as a function of the size of the first-pass 5-gram external language model.

<table>
<thead>
<tr>
<th>WER</th>
<th>Spoken Queries (All)</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-pass &amp; Second-pass</td>
<td>1/8</td>
</tr>
<tr>
<td>5M n-gram</td>
<td>6.8</td>
</tr>
<tr>
<td>16M n-gram</td>
<td>6.9</td>
</tr>
<tr>
<td>45M n-gram</td>
<td>7.0</td>
</tr>
<tr>
<td>84M n-gram</td>
<td>7.1</td>
</tr>
<tr>
<td>116M n-gram</td>
<td>7.2</td>
</tr>
<tr>
<td>150M n-gram</td>
<td>7.3</td>
</tr>
<tr>
<td>225M n-gram</td>
<td>7.5</td>
</tr>
<tr>
<td>300M n-gram</td>
<td>7.6</td>
</tr>
<tr>
<td>450M n-gram</td>
<td>8.1</td>
</tr>
</tbody>
</table>

4.4. Size of the first-pass external LM

We experimented with reducing the size of the first-pass external LM using relative-entropy pruning [31]. Table 3 shows the effect of this size reduction on WER on the full spoken queries test set. Strikingly, for the written-grapheme system, all of the

tail TTS queries is that the grapheme-based system performs worse on rare words. These rare words might be too infrequent in the training data for the grapheme-based Seq-2-Seq models to effectively learn their pronunciations. However, these pronunciations are available to the phoneme-based systems through the hand-constructed pronunciation lexicon.

To validate this hypothesis, we devised the following approach for injecting pronunciations in the grapheme-based systems. The idea is to augment the graphemic lexicon with alternate spellings of some words, chosen to be closer to the actual pronunciation of those words. The lexicon will map the standard and alternate spellings to the corresponding word and the external LM will be used to disambiguate between words now sharing spellings. To find alternate spellings for a given word \( x \), we look up each of its phonemic pronunciations \( p \) in a pronunciation dictionary. We then apply a finite-state-based G2P system in reverse on \( p \) to produce the most likely (highest probability) spelling \( s \) for \( p \). If \( s \) is not the normal spelling for \( x \), we add \( s \) as an alternate spelling for \( x \) in the graphemic lexicon. For instance, for the rare proper name "daved", the graphemic lexicon will provide two possible spellings as sequence of graphemes, the normal spelling "d a v e d" and the alternate spelling "d a v i d" (since "daved" is found to be pronounced "d a v i d" for which the most probable spelling in the G2P model is "d a v i d").

Results in Table 2 (W1 vs W3, S1 vs S3) show the approach is working very well and leading to decent WER reductions on hard spoken queries and long-tail TTS queries test sets. Combined with the neural denorm system, alternate spellings make the spoken-grapheme-based HAT with 5-gram external LM (S4) the best performing system on spoken queries.

5. Conclusion

We summarize our principal findings as follows:

**Language modeling matters:** By providing an internal LM estimate, HAT allows for better integration with external LMs than the approaches like shallow fusion that are typically used with Seq-2-Seq models. The usual observations hold: applying the LM earlier and using larger LMs bring more improvements.

**Pronunciation matters:** Pronunciation helps on rare words and can be leveraged in graphemic systems through the use of alternate spellings.

**Written-domain WER is a biased metric:** Written-domain WER depends heavily on transcription conventions and thus favors systems with written-domain trained Seq-2-Seq models which are able to learn these conventions.

**All modeling units work well:** The usual wisdom that phonemic units work better than phonemes for Seq-2-Seq in ASR does not seem to hold. Combined with an external LM, phoneme Seq-2-Seq models perform very closely to grapheme models on the head of the distribution where phonemes-based models are handicapped by the written-domain WER metric but outperform them on the tail.

**Hybrid approach brings flexibility:** HAT brings flexibility in modeling and inference (choice of unit and LM type) but also flexibility in application design: the same Seq-2-Seq model can be deployed on device, with the small neural LM, to minimize footprint but also on server, with pronunciation-derived lexicons and a large LM, for the best performance.

Figure 1: WER as a function of the fraction of the training data.

WER loss can be recovered in the second-pass even when pruning all the way to 5.6 million n-grams, keeping all 4 million unigrams and only 1.6 million higher-order n-grams. For the phoneme and spoken-grapheme systems, the effect on WER is more pronounced but still a significant size reduction can be achieved without WER loss after second-pass LM rescoring.

4.5. Amount of training data

Figure 1 shows the WER achieved on the Spoken Queries test set when training the HAT models only on 1/8th and 1/4th of the available data. The use of an external LM, in the first pass for the written-grapheme system and second pass for all systems, has an even larger impact on WER when less data is used.
6. References


