INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

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 common speech phrases and significant gains on uncommon ones compared to the state-of-the-art approaches.

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

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 very appealing for server-based applications, which allow processing of multiple queries from different users in one batch. For on-device applications, the benefit is less clear due to the sequential nature of speech models and the single user. For the memory footprint, on-device models usually need to be smaller than server-based models.

State-of-the-art conventional [1, 2] and sequence-to-sequence (Seq-2-Seq) [5, 6, 7] models are quite different regarding these design factors. The conventional models estimate the posterior probability of a word sequence given an acoustic sequence by decomposing it into an acoustic model (AM) and a language model (LM) probability using Bayes' rule. The AM and LM scores are commonly combined using weighted finite state transducers (WFSTs) [3, 4]. The Seq-2-Seq models directly estimate the posterior probability of a word sequence given an acoustic sequence. The modular design of conventional systems makes them more favorable than Seq-2-Seq models in terms of the training data and inference flexibility design criteria. A decent AM can be trained with a relatively small amount of transcribed audio data whereas the LM can be trained on text-only data, which is available in vast amounts for many languages. The model components of a conventional model can be modified by principled methods such as vocabulary augmentation [12], LM adaptation and contextual biasing [13]. Conversely, the Seq-2-Seq models are usually preferred over conventional models in terms of memory footprint [14, 15] which is a critical design parameter for on-device applications.

This paper proposes and evaluates different ASR design strategies based on the hybrid autoregressive transducer (HAT) model [8]. The HAT model can be seen as an extension of the RNN-T model [6] with a different probabilistic formulation that admits an estimate of the internal language model contribution. This estimate allows flexible decoding and rescoring strategies within the hybrid formulation of conventional models [1, 2]. These decoding and rescoring strategies combined with the strong Seq-2-Seq-based probabilistic HAT model allow design strategies that fulfill all aforementioned design parameters. A HAT model can be used as a first-pass system decoded with an external (neural or non-neural) LM, applied either during the first-pass or in a second-pass rescoring. As a special case, HAT can be used as a drop-in replacement of an acoustic model in a conventional system and with a significantly smaller external language model. The HAT model performs equally well across different choices of modeling units, on difficult tail phrases, and it scales well with training data size.

2. Hybrid autoregressive transducers

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

$$\begin{cases} b_{t,u} & \tilde{y}_{t+u} = \text{<b>} \\ (1 - b_{t,u}) P_{t,u}(y_u|x, y_{1:u}) & \tilde{y}_{t+u} = y_u \end{cases}$$

Copyright © 2021 ISCA

http://dx.doi.org/10.21437/Interspeech.2021-658

The HAT model does not provide any strict parametric form for neither $b_{t,u}$ nor $P_{t,u}$. 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 $P(\tilde{y}|x)$ is derived. The posterior probability of y given x is then modeled by summing all the alignment posteriors:

$$P(y|x) = \sum_{\tilde{y}:B(\tilde{y})=y} P(\tilde{y}|x) \quad (1)$$

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]: $P_{\text{ILM}}(y) = \prod_{1:U} P_{t,u}(y_u|\mathbf{0}, y_{1:u-1})$ which is the chain of label distribution $P_{t,u}$ 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 which can be used for integration with an external language model either during a first-pass beam search or second-pass rescoring [8].

3. Flexible ASR design with HAT

HAT's provision of an internal LM probability estimate opens up alternative design choices for server-based and on-device based ASR: (1) HAT can be combined with an external LM during first-pass decoding but also without one as a drop-in RNN-T replacement, (2) the choice of external LM is wide open including n-gram models and more general WFSTs to neural LMs, (3) HAT always provides a language model estimate in (first-pass) decoding (either from the internal or external LM) enabling critical features such contextual language model biasing and proper second-pass language model rescoring, and (4) a word-level external LM, applied in the first or second pass, can be used to disambiguate among word sequences corresponding to the same unit sequence, removing the constraint of having a one-to-one mapping between y and w and opening up the choice of modeling units. Formally, let $y \triangleright w$ denote that the unit sequence $y \in M^*$ is a possible instantiation of the word sequence $w \in V^*$. Assuming that our external LM assigns probabilities to word sequences in V^*, HAT model inference can then be formulated as the search for the pair $(y^\diamond, w^\diamond) \in M^* \times V^*$ for which $y \triangleright w$ and which maximizes:

$$\lambda_1 \, log P(y|x) - \lambda_2 \log P_{\text{ILM}}(y) + \log P_{\text{ELM}}(w) \quad (2)$$

where the first term is the posterior score for the unit sequence y scaled by scalar λ_1, the second term is the prior score for unit sequence y scaled by scalar λ_2 and the third is the external language model score for word sequence w.

3.1. Decoding strategy

The decoding strategy used here is time-synchronous with breadth-first search. At each time frame t a beam of N partial hypotheses is used. Each partial hypothesis $\tilde{y}_{t,u}$ holds the feature encoder state at the time frame t and the label state sequence of the first u labels. The partial paths are extended by up to a fixed number of label expansions followed by a time expansion. Extending partial hypothesis $\tilde{y}$ with label y' updates the partial beam score as: $s(\tilde{y}_{t,u} \cdot y') \mathrel{+}= \lambda_1 \log P(y'|x, B(\tilde{y}_{t,u})) - \lambda_2 \log P(y'|B(\tilde{y}_{t,u}))$ where the first term is the local posterior calculated from HAT model and the second term is the HAT prior score for label y. Similarly the blank extension updates the partial score by: $s(\tilde{y}_{t,u} \cdot \texttt{<b>}) \mathrel{+}= \lambda_1 \log P(\texttt{<b>}|x, \tilde{y}_{t,u})$. Both the posterior and prior values are calculated by logit smoothing 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, λ_1, λ_2 and the smoothing parameters can be estimated by sweeping on a held out set.

Table 1: *Decoder graph construction from the following component WFSTs: L_p, L_{sg}, L_{wg}, phonemic, spoken and written graphemic lexicons; V, verbalizer; G, top-level n-gram LM; $\mathcal{C}_s$, $\mathcal{C}_w$, spoken and written-domain class grammars.*

HAT modeling unit	Decoder graph
phoneme	$T_p = L_p \circ V \circ \text{Replace}(G, \mathcal{C}_s)$
spoken grapheme	$T_{sg} = L_{sg} \circ V \circ \text{Replace}(G, \mathcal{C}_s)$
written grapheme	$T_{wg} = L_{wg} \circ \text{Replace}(G, \mathcal{C}_w)$

3.2. Second-pass language model rescoring

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

$$\underset{(y,w)\in\mathbb{H}(x)}{\text{argmax}} \; \lambda_1 \log P(y|x) - \lambda_2 \log P_{\text{ILM}}(y) + \mu_1 \log P_{\text{ELM}}(w) + \mu_2 \log P_{\text{RLM}}(w) \quad (3)$$

where μ_1 and μ_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 $P_{\text{ELM}}(w)$ by $\lambda_2 P_{\text{ILM}}(y)$ 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 phonemic transcript, e.g. "`m i t { t sil f aI 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. "*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. "*xbox 360 $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-

ten domain but as a sequence of single-character words that needs to be concatenated together. e.g. "*xbox 360 <m> $ 5 0 </m> 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 *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.*

WER		**Spoken Queries**						**TTS Long-tail Queries**		
first-pass (10-best oracle) second-pass		All			Hard					
B0	Written grapheme RNN-T	6.9	(1.6)	6.3	33.5	(17.4)	29.5	40.0	(22.1)	36.9
W0	Written grapheme HAT w/ internal LM	6.9	(1.6)	6.0	33.5	(17.4)	28.3	40.0	(22.1)	34.7
W1	Written grapheme HAT w/ 5-gram ELM	6.0	(2.1)	5.8	26.8	(15.6)	24.8	33.3	(20.5)	29.6
W2	Written grapheme HAT w/ neural ELM	6.4	(1.3)	6.0	30.2	(14.7)	26.7	34.4	(18.0)	31.8
S1	Spoken grapheme HAT w/ 5-gram ELM	6.3	(1.7)	6.0	25.8	(14.1)	25.0	30.6	(16.6)	29.0
P1	Phoneme HAT w/ 5-gram LM	6.8	(1.6)	6.2	22.6	(10.2)	20.8	26.6	(11.8)	22.4
S2	S1 with neural numeric class denorm	6.1	(1.6)	5.8	26.0	(14.0)	25.2	30.9	(16.9)	29.1
P2	P1 with neural numeric class denorm	6.5	(1.4)	6.0	22.9	(8.5)	21.1	26.7	(12.0)	22.5
W3	W1 with alternate spelling lexicon	6.1	(2.0)	5.9	26.4	(15.6)	24.2	31.5	(18.4)	27.4
S3	S1 with alternate spelling lexicon	6.2	(1.6)	5.9	25.7	(13.9)	24.1	28.8	(14.7)	26.7
S4	S2 with alternate spelling lexicon	6.1	(1.5)	5.8	25.9	(10.5)	24.4	29.0	(14.9)	27.1

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.*

WER		**Spoken Queries** (All)					
first-pass second-pass		P1		S1		W1	
ELM	97M n-gram	6.8	6.2	6.3	6.0	6.0	5.8
	64M n-gram	6.9	6.2	6.4	6.0	6.1	5.8
	45M n-gram	7.0	6.2	6.5	6.1	6.1	5.8
	32M n-gram	7.1	6.2	6.5	6.1	6.2	5.8
	22M n-gram	7.2	6.3	6.5	6.1	6.2	5.8
	16M n-gram	7.3	6.3	6.6	6.1	6.3	5.8
	11M n-gram	7.5	6.4	6.7	6.1	6.4	5.8
	8M n-gram	7.6	6.4	6.8	6.1	6.5	5.8
	5.6M n-gram	8.1	6.6	7.0	6.3	6.7	5.8

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-curated 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 lookup 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 "*daveed*", the graphemic lexicon will provide two possible spellings as sequence of graphemes, the normal spelling "`d a v e e d`" and the alternate spelling "`d a v i d`" (since "*daveed*" is found to be pronounced "`d @ 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.

4.4. Size of the fist-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

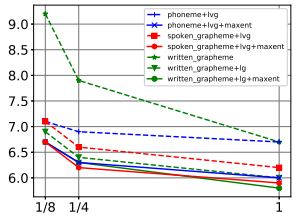

Figure 1: *WER as a function of the fraction of the training data.*

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.

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 graphemic 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.

6. References

[1] F. Jelinek, "Continuous speech recognition by statistical methods," *Proceedings of the IEEE*, vol. 64, no. 4, pp. 532–556, 1976.

[2] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 2, pp. 179–190, 1983.

[3] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[4] ——, "Speech recognition with weighted finite-state transducers," in *Handbook of Speech Processing*, J. Benesty, M. Sondhi, and Y. Huang, Eds. Springer, 2008, ch. 28, pp. 559–582.

[5] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[6] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[7] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[8] E. Variani, D. Rybach, C. Allauzen, and M. Riley, "Hybrid autoregressive transducer (HAT)," in *ICASSP*, 2020, pp. 6139–6143.

[9] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, and B. Strope, ""your word is my command": Google search by voice: A case study," in *Advances in speech recognition*. Springer, 2010, pp. 61–90.

[10] M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope, "Deploying goog-411: Early lessons in data, measurement, and testing," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2008, pp. 5260–5263.

[11] I. McGraw, R. Prabhavalkar, R. Alvarez, M. G. Arenas, K. Rao, D. Rybach, O. Alsharif, H. Sak, A. Gruenstein, F. Beaufays *et al.*, "Personalized speech recognition on mobile devices," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5955–5959.

[12] C. Allauzen and M. Riley, "Rapid vocabulary addition to context-dependent decoder graphs," in *Interspeech 2015*, 2015.

[13] K. Hall, E. Cho, C. Allauzen, F. Beaufays, N. Coccaro, K. Nakajima, M. Riley, B. Roark, D. Rybach, and L. Zhang, "Composition-based on-the-fly rescoring for salient n-gram biasing," in *Interspeech 2015, International Speech Communications Association*, 2015.

[14] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[15] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6381–6385.

[16] N. Morgan and H. Bourlard, "Continuous speech recognition using multilayer perceptrons with hidden markov models," in *International conference on acoustics, speech, and signal processing*. IEEE, 1990, pp. 413–416.

[17] E. Variani, E. McDermott, and G. Heigold, "A gaussian mixture model layer jointly optimized with discriminative features within a deep neural network architecture," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4270–4274.

[18] N. Kanda, X. Lu, and H. Kawai, "Minimum bayes risk training of ctc acoustic models in maximum a posteriori based decoding framework," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4855–4859.

[19] E. McDermott, H. Sak, and E. Variani, "A density ratio approach to language model fusion in end-to-end automatic speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 434–441.

[20] H. Sak, Y. Sung, F. Beaufays, and C. Allauzen, "Written-domain language modeling for automatic speech recognition," in *INTERSPEECH*, 2013, pp. 675–679.

[21] L. Vasserman, V. Schogol, and K. B. Hall, "Sequence-based class tagging for robust transcription in ASR," in *INTERSPEECH*, 2015, pp. 473–477.

[22] H. Sak, F. Beaufays, K. Nakajima, and C. Allauzen, "Language model verbalization for automatic speech recognition," in *ICASSP*, 2013, pp. 8262–8266.

[23] C. Allauzen and M. Riley, "Pre-initialized composition for large-vocabulary speech recognition," in *INTERSPEECH*, 2013, pp. 666–670.

[24] P. S. Aleksic, C. Allauzen, D. Elson, A. Kracun, D. M. Casado, and P. J. Moreno, "Improved recognition of contact names in voice commands," in *ICASSP*, 2015, pp. 5172–5175.

[25] C. Allauzen, M. Riley, and J. Schalkwyk, "A generalized composition algorithm for weighted finite-state transducers," in *INTERSPEECH*, 2009, pp. 1203–1206.

[26] E. Variani, T. Bagby, E. McDermott, and M. Bacchiani, "End-to-end training of acoustic models for large vocabulary continuous speech recognition with tensorflow," in *INTERSPEECH*, 2017, pp. 1641–1645.

[27] C. Kim, E. Variani, A. Narayanan, and M. Bacchiani, "Efficient implementation of the room simulator for training deep neural network acoustic models," *arXiv preprint arXiv:1712.03439*, 2017.

[28] F. Biadsy, M. Ghodsi, and D. Caseiro, "Effectively building tera scale maxent language models incorporating non-linguistic signals," in *INTERSPEECH*, 2017, pp. 2710–2714.

[29] C. Peyser, S. Mavandadi, T. N. Sainath, J. Apfel, R. Pang, and S. Kumar, "Improving Tail Performance of a Deliberation E2E ASR Model Using a Large Text Corpus," in *INTERSPEECH*, 2020, pp. 4921–4925.

[30] C. Peyser, H. Zhang, T. N. Sainath, and Z. Wu, "Improving performance of end-to-end ASR on numeric sequences," in *INTERSPEECH*, 2019, pp. 2185–2189.

[31] A. Stolcke, "Entropy-based pruning of backoff language models," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, 1998, pp. 270—-274.