JOINT ASR AND LANGUAGE IDENTIFICATION USING RNN-T: AN EFFICIENT APPROACH TO DYNAMIC LANGUAGE SWITCHING

Surabhi Punjabi, Harish Arsikere*, Zeynab Raeesy†, Chander Chandak†, Nikhil Bhave†, Ankish Bansal, Markus Müller, Sergio Murillo, Ariya Rastrow, Andreas Stolcke, Jasha Droppo, Sri Garimella, Roland Maas, Mat Hans, Athanasios Mouchtaris, Siegfried Kunzmann*

Alexa Machine Learning, Amazon

ABSTRACT

Conventional dynamic language switching enables seamless multilingual interactions by running several monolingual ASR systems in parallel and triggering the appropriate downstream components using a standalone language identification (LID) service. Since this solution is neither scalable nor cost- and memory-efficient, especially for on-device applications, we propose end-to-end, streaming, joint ASR-LID architectures based on the recurrent neural network transducer framework. Two key formulations are explored: (1) joint training using a unified output space for ASR and LID vocabularies, and (2) joint training viewed as multi-task optimization. We also evaluate the benefit of using auxiliary language information obtained on-the-fly from an acoustic LID classifier. Experiments with the English-Hindi language pair show that: (a) multi-task architectures perform better overall, and (b) the best joint architecture surpasses monolingual ASR (6.4–9.2% word error rate reduction) and acoustic LID (53.9–56.1% error rate reduction) baselines while reducing the overall memory footprint by up to 46%.

Index Terms— joint modeling, multilingual, language identification, recurrent neural network transducer, code switching

1. INTRODUCTION

Digital voice assistants, embodied in far-field smart speakers, have become increasingly common over the last few years. When such devices are used in multilingual households, users prefer to interact with them in more than one language seamlessly, instead of restricting themselves to one language all the time or selecting their language of choice before every interaction. In order to support such dynamic language switching from one interaction to the next (which is different from code switching, where words from more than one language are used within the same interaction or utterance), the back-end intelligence must perform language identification (LID) along with automatic speech recognition (ASR) so that the detected language can be used to trigger appropriate downstream systems such as natural language understanding, text-to-speech synthesis, etc., which usually are language specific.

To enable dynamic language switching between two or more pre-specified languages, a commonly adopted strategy is to run several monolingual ASR systems in parallel along with a standalone acoustic LID module, and pass only one of the recognized transcripts downstream depending on the outcome of language detection [1]. While this approach works well in practice, it is neither cost-effective for more than two languages, nor suitable for on-device scenarios where compute resources and memory are limited. To this end, we propose all-neural architectures that can jointly perform ASR and LID for a group of pre-specified languages and thereby significantly improve the cost effectiveness and efficiency of dynamic language switching. We base our architectures on the recurrent neural network transducer (RNN-T) framework given its end-to-end, streaming nature, state-of-the-art ASR performance, and suitability for on-device applications [2–4].

Joint ASR-LID modeling, by definition, involves multilingual speech recognition. Multilingual ASR models are trained by pooling data from the languages of interest, and it is often observed that languages with smaller amounts of data benefit more from this [5–14]. In studies using popular end-to-end architectures such as RNN-T or Listen, Attend and Spell (LAS) [15], multilingual ASR performance is usually enhanced by providing auxiliary language inputs to the model [16–21]. Depending on whether or not the language spoken is known beforehand at runtime, language inputs can be provided in the form of constant *one-hot* vectors or on-the-fly prediction vectors (e.g., posteriors from a streaming LID model), respectively. Waters *et al.* train a streaming language detector using RNN-T loss and provide its predictions as auxiliary inputs to the encoder of a multilingual RNN-T ASR system [21]. In this paper, we train an acoustic LID model using cross-entropy loss and provide its predictions to the *joint network* of RNN-T based ASR-LID models (the motivation behind this is discussed in Section 2).

Recent studies on LID have shown that using both acoustic and lexical cues yields much better classification accuracies as compared to using either of them in isolation (especially for accented and/or code-switched speech) [1, 22]. We leverage this in the proposed joint ASR-LID architectures by (a) including language targets, along with ASR targets, in the RNN-T softmax layer (similar to the LAS-based study in [23]), or (b) modeling language targets separately as a function of RNN-T encoder, decoder and joint network states. The former approach introduces an implicit relationship between language detection and the evolution of RNN-T encoder (acoustic) and decoder (lexical) states, while the latter approach makes the relationship explicit by de-coupling ASR and LID tasks and training them in a multi-task fashion.

The proposed joint ASR-LID architectures are trained and evaluated using data from Indian English and spoken Hindi, a language pair that exemplifies real-world usage of dynamic language switching. The two languages make for an interesting case study given that they share content (types of queries, named entities, etc.) and exhibit frequent code switching (e.g., a spoken Hindi utterance of five or six words can sometimes have just one Hindi word). We present both ASR and LID metrics for all the joint architectures explored in this paper, and show for the chosen language pair that a joint ASR-LID

*Equal contribution †Equal contribution

model can outperform the components comprising a conventional dynamic language switching setup.

To our knowledge, this is the first body of work to (a) propose and evaluate a variety of joint ASR-LID architectures in the RNN-T framework, and (b) present a detailed analysis of both ASR and LID metrics in the context of dynamic language switching.

2. JOINT ASR-LID ARCHITECTURES

As depicted by Figure 1, we propose two RNN-T-based, joint ASR-LID modeling paradigms: (1) *coupled* training, where the ASR and LID tasks are modeled using a single softmax layer and RNN-T loss function, and (2) *multi-task* training, where the two tasks are modeled using separate, yet jointly minimized losses.

RNN-T, which is a streaming, alignment-free sequence transduction model, typically consists of a transcription network (or encoder), a prediction network (or decoder) and a joint network [2, 3]. The encoder and decoder are generally composed of multiple LSTM layers. The encoder converts an input acoustic feature vector, $\mathbf{x}_t$, to a hidden representation, $\mathbf{h}_t^{enc}$, while the decoder consumes the last non-blank symbol observed, y_{u-1}, and outputs a hidden representation, $\mathbf{h}_u^{dec}$. The joint network, which is usually composed of one or more feedforward layers, combines the encoder and decoder representations to produce logits, $\mathbf{z}_{t,u}$:

$$\mathbf{z}_{t,u} = f^{joint}(\mathbf{h}_t^{enc}, \mathbf{h}_u^{dec}). \quad (1)$$

The logits in turn are passed through a softmax layer to produce a probability distribution for the next output symbol, y_u, over the set $\bar{\mathcal{V}} = \mathcal{V} \cup \phi$, where ϕ denotes the blank symbol and $\mathcal{V}$ denotes the original model vocabulary.

To improve ASR performance through auxiliary language inputs while not incurring additional runtime latency, we experiment with frame-level language representations obtained on-the-fly from an acoustic LID model that is trained beforehand. This LID model is a simple LSTM-based network that produces a language representation vector, $\mathbf{l}_t$, corresponding to an input acoustic feature vector, $\mathbf{x}_t$. Unlike previous studies where the auxiliary language information is provided to one or more RNN-T encoder layers [18, 21], we provide frame-level language representations to the RNN-T joint network. The rationale here is that language information might be more influential deeper inside the network where higher-level acoustic and lexical representations are combined. This hypothesis is indeed confirmed via preliminary experiments where we observe a 2-4% relative improvement by providing language representations to the joint network instead of concatenating them with acoustic features (i.e. encoder inputs). When language representations are provided to the joint network, the logit computation in Eq. (1) is modified as below:

$$\mathbf{z}_{t,u} = f^{joint}(\mathbf{h}_t^{enc}, \mathbf{l}_t, \mathbf{h}_u^{dec}). \quad (2)$$

2.1. Coupled training

We first explore a training paradigm that facilitates strong coupling between ASR and LID. In this approach, the RNN-T output symbol space jointly models both ASR targets and language targets as indicated by the dashed box marked "(a)" in Figure 1, which implies that at each step, the model can output either the blank symbol, one of the language symbols or one of the ASR symbols. The resulting augmented vocabulary can be written as $\bar{\mathcal{V}} = \mathcal{V}_{asr} \cup \mathcal{V}_{lid} \cup \phi$, where $\mathcal{V}_{asr}$ and $\mathcal{V}_{lid}$ denote ASR and LID vocabularies, respectively.

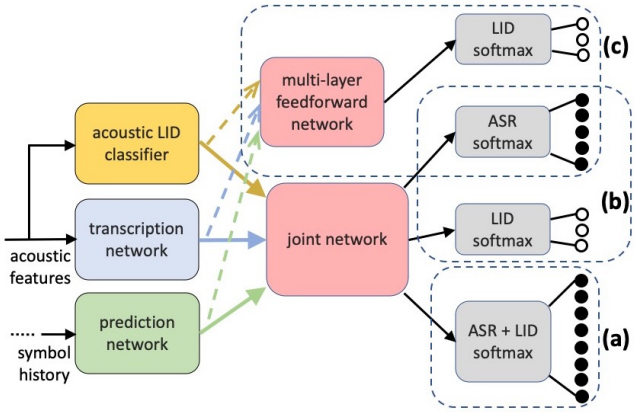

Fig. 1. A block schematic of the proposed RNN-T based joint ASR-LID architectures. For softmax layers, filled circles denote RNN-T loss and empty circles denote cross-entropy loss. The *coupled* training paradigm (Sec. 2.1) is captured by box "(a)", and the *shared* and *decoupled multi-task* training paradigms (Sec. 2.2) are captured by boxes "(b)" and "(c)", respectively.

Given a training sample with ASR targets $(w_1 w_2 \dots w_U)$, where U denotes the target sequence length and $w_i \in \mathcal{V}_{asr}$ for $i \in [1, U]$, and an associated language label l, where $l \in \mathcal{V}_{lid}$, the target sequence for this architecture becomes $(w_1 w_2 \dots w_U\ l)$. The intuition behind having language targets as the utterance-final tokens is that the network's belief of the underlying language improves with the incoming acoustic feature frames and the partially predicted text. Akin to the end-of-sentence or <eos> symbol that is used to indicate utterance ends and optionally perform endpointing [24], appending the language target to ASR targets teaches the network to delay language detection for as long as possible, ideally to a point where all of the acoustic and lexical evidence has been observed.

Owing to the fact that the coupled joint model implicitly sees language targets as end-of-sentence tokens during training, the network exhibits a tendency to emit language symbols prematurely, thereby leading to high ASR deletion rates. To mitigate this issue, beam-search decoding is constrained using the penalty scheme proposed previously for joint ASR and endpointing [24]. In this scheme, a language symbol $y_u \in \mathcal{V}_{lid}$ is allowed to appear in the decoding beam only if the condition in Eq. (3) is satisfied.

$$P(y_u|t,u)^{\alpha} \geq \beta, \text{ where } P(y_u|t,u) = \text{softmax}(\mathbf{z}_{t,u}) \quad (3)$$

In Eq. (3), the tunable hyperparameters α and β serve to first scale and then threshold the language symbol posteriors. It is found from preliminary experiments that larger values of β lead to lower ASR deletions, with the extreme case of β = 1, which essentially amounts to not emitting the language symbols at all, yielding the best ASR performance (i.e. the lowest deletion rate). We therefore fix β to 1 for the coupled joint ASR-LID architecture. Since language symbols are not emitted in the model's output when β is 1, language decisions are made simply based on the unscaled language symbol posteriors available after the last audio frame.

2.2. Multi-task training

The architecture described in Section 2.1 introduces tight coupling between ASR and LID by virtue of using a common output symbol space, softmax layer and loss function (RNN-T loss). In contrast, the multi-task training paradigm makes the tasks loosely coupled while

still allowing mutual learning via joint optimization of a combined loss function. Since LID is an utterance-level task with unit-length training targets—unlike ASR, where a sequence-to-sequence mapping must be learned—, we use cross-entropy (CE) loss for LID and RNN-T loss for ASR in our formulation. For a training sample with acoustic feature sequence $\mathbf{X}$, ASR target sequence Y_{asr}, and unit-length LID target sequence Y_{lid}, the standard multi-task loss can be written using Eq. (4):

$$\mathcal{L} = \lambda \mathcal{L}_{\text{RNN-T}}(\theta_{asr}; (\mathbf{X}, Y_{asr})) + (1-\lambda)\mathcal{L}_{\text{CE}}(\theta_{lid}; (\mathbf{X}, Y_{lid})), \quad (4)$$

where θ_{asr} and θ_{lid} denote the ASR- and LID-specific parameters, respectively, and λ denotes the ASR task weight. The following variants are explored depending on the choice of θ_{asr} and θ_{lid}.

The simplest architecture is one where all the network components are shared between the two tasks except for the final projection (affine transform plus softmax) layers, as depicted by the dashed box marked "(b)" in Figure 1. In this case, both θ_{asr} and θ_{lid} are composed of the encoder, decoder and joint network parameters.

Full parameter sharing as described above may not be the best strategy because ASR and LID tasks are different enough (sequence transduction versus classification, respectively) for them to benefit from different network paths and granularities for the backpropagated gradients. In order to allow for dedicated LID parameters, we explore architectures where language detection is done using a separate shallow feedforward network (dashed box marked "(c)" in Figure 1). In this case, θ_{lid} differs from θ_{asr} since it includes the feedforward network parameters instead of the joint network parameters. Three variants are studied depending on what the feedforward network consumes: (1) the final encoder representation corresponding to the last acoustic frame and the final decoder representation corresponding to the last ASR symbol, (2) averaged encoder and decoder representations, and (3) normalized encoder and decoder representations using an attention mechanism. The first variant should ideally encapsulate sequence information owing to the recurrent nature of the encoder and decoder, but the second and third variants are expected to work better since they can circumvent the potential loss of information caused by the "forgetting" behavior of LSTM networks. Note that the second variant is a special case of the third since averaging is equivalent to uniform attention.

3. EXPERIMENTAL SETUP

The corpus used for RNN-T training consists of in-house, far-field, de-identified voice-assistant recordings amounting to about 3.8k and 12.5k hours of spoken Hindi and Indian English data, respectively. The acoustic LID classifier (used for baseline LID and for providing language representations to RNN-T) is trained using 2k hours of balanced English-Hindi data. A key characteristic of spoken or colloquial Hindi is the high frequency of within-utterance code switching induced by common English verbs and nouns such as "play", "off", "alarm", "song", etc. Similarly, Indian English utterances often contain Hindi (and other language) named entities in the form of album names, song titles and other proper nouns. Since ASR and LID performance can be significantly different for "pure" and code-switched utterances, we partition our evaluation data into four test sets: English-pure (21 hours), English-mixed (4.8 hours), Hindi-pure (3 hours) and Hindi-mixed (2.6 hours).

The acoustic front-end for RNN-T and LID classifier training consists of 64-dimensional log Mel-filterbank energy features extracted using 25 ms windows spaced at 10 ms intervals. The features are normalized using global mean-variance statistics, stacked with two left-context frames and downsampled to a frame rate of 30 ms. SpecAugment [25] is applied to all RNN-T models; two frequency masks, each with a maximum width of 24 channels, are applied to each training utterance. LID classifier training uses data augmentation via simulated reverberation.

Unless mentioned, all the RNN-T models have the following architecture: 5 encoder LSTM layers and 2 decoder LSTM layers with 1024 units each, a 512-dimensional embedding layer at the decoder's input, and a single-layer, 512-unit feedforward joint network with $\tanh$ non-linearity. The acoustic LID classifier has 3 LSTM layers with 256 units each, followed by a 32-dimensional projection layer. For the decoupled multi-task architecture (box "(c)" in Figure 1), the language detector uses a 2-layer feedforward network with 512 units each. The ASR task weight λ is set to 0.9 for multi-task training. In all experiments, the ASR vocabulary, $\mathcal{V}_{asr}$, comprises 4000 subword units determined via byte pair encoding [26]. For the LID classifier (trained using CE loss), utterance-level language labels are propagated to all the acoustic training vectors.

All models are trained in a distributed fashion using Horovod [27]. RNN-T models and the acoustic LID classifier are trained using 24 and 16 GPUs, respectively, with a per-GPU batch size of 64. For parameter learning, Adam optimizer is used with a learning rate schedule that involves a short warmup phase, a constant hold phase and an exponential decay phase. A beam width of 16 is used for RNN-T beam search decoding; no temperature scaling is applied.

4. RESULTS AND DISCUSSION

Table 1 summarizes the performance of the proposed joint ASR-LID approaches along with that of the monolingual ASR (A0 and A1) and acoustic LID classifier (L0) baselines. $J_{coupled}$ (A2–A5) is used to denote the coupled training paradigm, while $J_{multi\text{-}shared}$ (A6) and $J_{multi\text{-}decoupled}$ (A7–A10) are used to denote the multi-task training paradigms with full parameter sharing and the use of a decoupled feedforward LID network, respectively. To assess ASR performance, we report normalized word error rate (nWER) numbers obtained by dividing the original WERs by the lowest WER observed across test sets and experiments; in other words, nWER is greater than or equal to 1.0. Normalized LID error rate (nLER) values, computed in a similar fashion, are used to assess language identification performance. The code-switched ("mixed") test sets show significantly worse ASR and LID performance compared to their "pure" counterparts—with the exception of Hindi nWERs—, confirming that code-switched utterances are inherently harder to recognize and classify.

4.1. Results with coupled training (A2–A5)

Relative to monolingual ASR (A0 and A1), coupled training using a unified output space (A2) leads to a regression in ASR performance for pure English and pure Hindi test sets. Mixed test sets, however, show improved nWERs, suggesting that data pooling improves generalization for code-switched speech. ASR performance further improves when auxiliary language inputs are provided to the RNN-T joint network (A2 vs. A3, A4). Furthermore, comparing A3 and A4, we observe that providing language posteriors yields higher gains as compared to providing language embeddings (i.e., acoustic LID classifier outputs obtained before the softmax layer).

Reduced nLERs are observed across test sets for coupled training (A2) indicating that language detection benefits from implicit ASR information. This observation is consistent with [22] where the LID performance was shown to benefit from the use of ASR 1-best hypotheses. Auxiliary language inputs improve LID performance on

Table 1. Normalized WERs (nWER) and normalized LID error rates (nLER) for the proposed joint ASR-LID architectures (A2–A10). Lower error rates indicate better performance, *with the lowest values of nWER and nLER being 1.00*. Monolingual ASR models (A0, A1) and the acoustic LID classifier (L0) serve as the ASR and LID baselines, respectively. For A7, A8 and A9, "last", "attention" and "average" specify the language classification mechanism based on the RNN-T encoder and decoder states. Based on the collective ASR and LID metrics across all four test sets, A9 is the best 5-encoder-layer joint model relative to the conventional dynamic language switching setup (A0+A1+L0).

Expt.	Model	Auxiliary language inputs	nWER				nLER			
			English		Hindi		English		Hindi	
			Pure	Mixed	Pure	Mixed	Pure	Mixed	Pure	Mixed
A0	Monolingual Hindi	-	-	-	2.45	2.17	-	-	-	-
A1	Monolingual English	-	1.03	2.02	-	-	-	-	-	-
L0	Acoustic LID	-	-	-	-	-	4.03	6.47	2.23	2.40
A2	$J_{coupled}$	None	1.17	1.69	2.56	2.12	2.35	2.95	1.15	2.15
A3	$J_{coupled}$	Embeddings	1.14	1.65	2.58	2.14	1.90	2.48	1.60	2.65
A4	$J_{coupled}$	Posteriors	1.12	1.61	2.50	2.07	2.05	2.58	1.33	2.40
A5	A4 + prepend LID targets	Posteriors	1.09	1.66	2.62	2.19	1.00	2.20	15.95	16.43
A6	$J_{multi\text{-}shared}$	Posteriors	1.11	1.72	2.59	2.05	1.90	2.83	2.08	3.05
A7	$J_{multi\text{-}decoupled}$ (last)	Posteriors	1.06	1.56	2.39	1.99	1.83	2.55	1.88	2.70
A8	$J_{multi\text{-}decoupled}$ (attention)	Posteriors	1.07	1.57	2.37	1.98	1.70	2.35	1.80	2.85
A9	$J_{multi\text{-}decoupled}$ (average)	Posteriors	**1.04**	**1.55**	**2.36**	**1.96**	**1.55**	**2.40**	**1.90**	**2.95**
A10	A9 + 8-layer encoder	Posteriors	1.00	1.50	2.30	1.93	1.55	2.20	1.73	2.60

English while degrading on Hindi (A2 vs. A3, A4), and using posteriors instead of embeddings offers a better operating point when the overall LID performance is considered (A4 vs. A3). For the remaining experiments, results are always shown with language posteriors as auxiliary inputs (given their collective ASR and LID benefit).

In [23], the approach of prepending language targets to ASR targets was studied for joint ASR-LID training of hybrid CTC-attention architectures. Since streaming RNN-T models, unlike non-streaming attention-based models, do not encode utterances in their entirety before decoding, they must perform language detection with very little context when they are trained using prepended language targets. Given this limitation, one would expect poor LID—and therefore poor ASR—performance from the prepending approach, and this is indeed what we see empirically (A5 vs. A4): while the metrics improve for pure English owing to its major share in the training corpus, Hindi metrics, especially nLERs, degrade significantly.

4.2. Results with multi-task training (A6–A10)

For multi-task variants, sharing all network components creates an optimization bottleneck and results in suboptimal performance compared to architectures having a decoupled LID feedforward network (A6 vs. A7–A9). Among the decoupled variants A7–A9, using averaged encoder and decoder representations yields the lowest nWERs (A9). Comparing the overall performance of coupled and multi-task models, we conclude that it is best to jointly optimize ASR and LID losses while having dedicated parameters for the two tasks.

A9 surpasses monolingual ASR baselines for all test sets (3.5–22.5% reduction) except pure English, where we still observe a 1.2% increase in the nWER. A9 also reduces the nLERs compared to the acoustic LID baseline L0 (14.6–64% reduction), except for the code-switched Hindi test set (20% increase). Since conversational-agent interactions typically contain language-agnostic wakewords such as "Alexa", "Hey Google", etc., masking such segments during multi-task loss computation might further improve the LID performance of the average- and attention-based decoupled multi-task models. Since joint models must recognize speech in two or more languages and also learn to classify them, increasing their capacity (i.e. number of parameters) is another mechanism to achieve better performance. By training A9 with an 8-layer encoder (A10), the model surpasses monolingual ASR baselines on all test sets (2.8–25.9% reduction); it also yields superior LID performance on English and pure Hindi test sets (21.4–66% reduction relative to L0) with reduced deterioration on Hindi code-switched data (6% increase relative to L0).

Table 2 compares A9 and A10, the best multi-task joint models, with the conventional dynamic language switching setup. They yield significant performance improvements while reducing the total memory footprint by up to ∼46%, thereby satisfying the primary objectives of this study.

Table 2. A comparison of the best multi-task joint models with the conventional dynamic language switching setup (A0+A1+L0). The nWER and nLER metrics are aggregated across all test sets. Numbers in parentheses denote reductions relative to A0+A1+L0.

Model	#Params (M)	nWER	nLER
A0+A1+L0	168	1.41	4.10
A9	91.2 (45.7%)	1.32 (6.4%)	1.89 (53.9%)
A10	136 (19.0%)	1.28 (9.2%)	1.80 (56.1%)

5. CONCLUSIONS

In this paper, joint ASR-LID architectures based on RNN-Ts are proposed as an efficient, on-device-suitable alternative to conventional dynamic language switching solutions. Two primary joint modeling paradigms are explored: coupled training, where ASR and LID vocabularies share the RNN-T output space, and multi-task learning, where ASR and LID losses are modeled using dedicated parameters but minimized jointly. Experiments with Indian English and spoken Hindi show that: (a) code-switched utterances are inherently difficult to recognize and classify, (b) multi-task learning provides superior ASR performance whereas coupled training offers better LID accuracy, and (c) multi-task models with a dedicated LID feedforward network offer the best performance overall. The proposed joint ASR-LID architectures are language agnostic and, in principle, can be scaled to more than two languages. Exploring architectures with parameter sharing between RNN-T encoder and acoustic LID classifier is a promising future direction for further decreasing memory and compute footprint.

6. REFERENCES

[1] Shengye Wang, Li Wan, Yang Yu, and Ignacio Lopez Moreno, "Signal combination for language identification," *arXiv preprint arXiv:1910.09687*, 2019.

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

[3] Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming 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.

[4] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S. Chang, W. Li, R. Alvarez, Z. Chen, C. Chiu, D. Garcia, A. Gruenstein, K. Hu, A. Kannan, Q. Liang, I. McGraw, C. Peyser, R. Prabhavalkar, G. Pundak, D. Rybach, Y. Shangguan, Y. Sheth, T. Strohman, M. Visontai, Y. Wu, Y. Zhang, and D. Zhao, "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *Proceedings of ICASSP*, 2020, pp. 6059–6063.

[5] Tanja Schultz and Alex Waibel, "Language-independent and language-adaptive acoustic modeling for speech recognition," *Speech Communication*, vol. 35, no. 1-2, pp. 31–51, 2001.

[6] Arnab Ghoshal, Pawel Swietojanski, and Steve Renals, "Multilingual training of deep-neural networks," in *Proceedings of the ICASSP*, Vancouver, Canada, 2013.

[7] Karel Vesely, Martin Karafiat, Frantisek Grezl, Milos Janda, and Ekaterina Egorova, "The language-independent bottleneck features," in *Proceedings of the Spoken Language Technology Workshop (SLT), 2012 IEEE*. 2012, pp. 336–341, IEEE.

[8] Stefano Scanzio, Pietro Laface, Luciano Fissore, Roberto Gemello, and Franco Mana, "On the use of a multilingual neural network front-end," in *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[9] Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz, "Automatic speech recognition for under-resourced languages: A survey," *Speech Communication*, vol. 56, pp. 85–100, 2014.

[10] Quoc Bao Nguyen, Jonas Gehring, Markus Müller, Sebastian Stüker, and Alex Waibel, "Multilingual shifting deep bottleneck features for low-resource ASR," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 5607–5611.

[11] Ngoc Thang Vu, Dau-Cheng Lyu, Jochen Weiner, Dominic Telaar, Tim Schlippe, Fabian Blaicher, Eng-Siong Chng, Tanja Schultz, and Haizhou Li, "A first speech recognition system for mandarin-english code-switch conversational speech," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 4889–4892.

[12] S. Thomas, S. Ganapathy, and H. Hermansky, "Multilingual mlp features for low-resource lvcsr systems," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4269–4272.

[13] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7304–7308.

[14] Harish Arsikere, Ashtosh Sapru, and Sri Garimella, "Multi-dialect acoustic modeling using phone mapping and online i-vectors," *Proc. Interspeech 2019*, pp. 2125–2129, 2019.

[15] William Chan, Navdeep Jaitly, Quoc V Le, and Oriol Vinyals, "Listen, attend and spell," *arXiv preprint arXiv:1508.01211*, 2015.

[16] H Seki, S Watanabe, T Hori, J Le Roux, and JR Hershey, "An end-to-end language-tracking speech recognizer for mixed-language speech," 2018.

[17] Markus Müller, Sebastian Stüker, and Alex Waibel, "Neural language codes for multilingual acoustic models," *arXiv preprint arXiv:1807.01956*, 2018.

[18] Anjuli Kannan, Arindrima Datta, Tara N Sainath, Eugene Weinstein, Bhuvana Ramabhadran, Yonghui Wu, Ankur Bapna, Zhifeng Chen, and Seungji Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," *arXiv preprint arXiv:1909.05330*, 2019.

[19] Bo Li, Tara N Sainath, Khe Chai Sim, Michiel Bacchiani, Eugene Weinstein, Patrick Nguyen, Zhifeng Chen, Yanghui Wu, and Kanishka Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4749–4753.

[20] Bo Li, Yu Zhang, Tara Sainath, Yonghui Wu, and William Chan, "Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5621–5625.

[21] Austin Waters, Neeraj Gaur, Parisa Haghani, Pedro Moreno, and Zhongdi Qu, "Leveraging language id in multilingual end-to-end speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 928–935.

[22] Chander Chandak, Zeynab Raeesy, Ariya Rastrow, Yuzong Liu, Xiangyang Huang, Siyu Wang, Dong Kwon Joo, and Roland Maas, "Streaming language identification using combination of acoustic signals and asr hypothesis," *Submitted to ICASSP 2021*.

[23] Shinji Watanabe, Takaaki Hori, and John R Hershey, "Language independent end-to-end architecture for joint language identification and speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 265–271.

[24] Shuo-Yiin Chang, Rohit Prabhavalkar, Yanzhang He, Tara N Sainath, and Gabor Simko, "Joint endpointing and decoding with end-to-end models," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5626–5630.

[25] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617.

[26] Rico Sennrich, Barry Haddow, and Alexandra Birch, "Neural machine translation of rare words with subword units," 2015.

[27] Alexander Sergeev and Mike Del Balso, "Horovod: fast and easy distributed deep learning in TensorFlow," *arXiv preprint arXiv:1802.05799*, 2018.