Improved Hybrid Streaming ASR with Transformer Language Models

Pau Baquero-Arnal, Javier Jorge, Adrià Giménez, Joan Albert Silvestre-Cerdà, Javier Iranzo-Sánchez, Albert Sanchis, Jorge Civera, Alfons Juan

Machine Learning and Language Processing (MLLP) research group, Valencian Research Institute for Artificial Intelligence (VRAIN), Univ. Politècnica de València (Spain)

{pabaar,jajorca,adgipas,juasilce,jairsan,josanna2,jorcisai,ajuanci}@vrain.upv.es

Abstract

Streaming ASR is gaining momentum due to its wide applicability, though it is still unclear how best to come close to the accuracy of state-of-the-art off-line ASR systems when the output must come within a short delay after the incoming audio stream. Following our previous work on streaming one-pass decoding with hybrid ASR systems and LSTM language models, in this work we report further improvements by replacing LSTMs with Transformer models. First, two key ideas are discussed so as to run these models fast during inference. Then, empirical results on LibriSpeech and TED-LIUM are provided showing that Transformer language models lead to improved recognition rates on both tasks. ASR systems obtained in this work can be seamlessly transfered to a streaming setup with minimal quality losses. Indeed, to the best of our knowledge, no better results have been reported on these tasks when assessed under a streaming setup.

Index Terms: streaming, hybrid ASR, language models, Transformer

1. Introduction

Fueled by recent progress in Automatic Speech Recognition (ASR), streaming ASR is attracting great attention from both the scientific community and the industry due to its immense applicability. When compared to conventional (off-line) ASR, streaming ASR is clearly more challenging for two main reasons. First, system output must be delivered in real time as the continuous input audio stream is processed. Second, the system cannot defer its decision (full output) until seeing the complete evidence, i.e. until the end of the acoustic stream. Thus, the challenge is how best to come close to the accuracy of state-of-the-art off-line ASR systems under streaming conditions; that is, under the constraint of delivering continuous output within a short delay, i.e. not much longer than a second, with respect to the incoming audio stream.

Following the very popular end-to-end approach to machine learning tasks, many authors are currently exploring the use of purely neural network-based systems to ASR in general and streaming ASR in particular [1, 2, 3, 4]. They are comparatively simple, easy-to-build systems from widely available deep learning toolkits and require minimal expert knowledge (human intervention). However, despite their simplicity and promising prospects, it is still unclear whether or not they will soon surpass state-of-the-art, hybrid systems combining separate acoustic and language neural network models under the conventional statistical decision framework.

This work is focused on language models (LMs) for hybrid ASR systems with a view to use them under the streaming setup. Following our previous work on real-time one-pass decoding with LSTM LMs [5], we recently introduced a novel streaming one-pass decoder that under strict streaming conditions, i.e. a delay of less than a second, achieves a minimal loss of recognition accuracy with respect to the off-line setting [6]. The system described in [6] used bidirectional LSTM (BLSTM) acoustic models and unidirectional LSTM LMs, both adjusted to work with a short-window limited view of the incoming audio stream. Inspired by the latest developments reported in [7] for language modelling in the off-line setup, in this work we report further improvements to our streaming decoder by replacing streaming-adapted LSTM LMs with Transformer models, with due adaptation. Empirical results on the LibriSpeech and TED-LIUM tasks show that Transformer language models (TLM) lead to top, state-of-the-art recognition rates and latencies under streaming conditions, so that these hybrid ASR systems can work both under the off-line and streaming scenarios with no significant differences in quality.

2. Streaming one-pass decoder

As indicated above, our streaming one-pass decoder was introduced in [6], as a continuation of previous work on real-time one-pass decoding with LSTM LMs [5]. In this decoder, hypotheses are organized by their history, and static look-ahead scores are precomputed in advance instead of being dynamically updated. To adapt the LSTM LMs, they are trained with Variance Regularization (VR) to avoid computation of the whole Softmax during search. The search is based on a lazy strategy regarding LM scores, that is, postponing their computation as much as possible. Also, two additional parameters for search pruning are introduced to control the trade-off between WER and RTF. On the acoustic side, BLSTMs are used with a sliding, overlapping window over the sequence, averaging outputs of all windows for each frame to obtain the acoustic score (see section 4.5 for experimental details). A short initial delay is introduced at the beginning of the stream for the purpose of initializing mean and variance statistics needed for feature normalization. This decoder structure and streaming setup demonstrated that can provide competitive performance in terms of WER and latency under a streaming regime in well-known benchmarks, as well as in our production systems, where it carries out the transcription of long recordings of lectures, seminars, and other similar contents.

3. Streaming Transformer language models

As with LSTM LMs, a first key idea to adapt TLMs to our streaming one-pass decoder is to add a VR term during training as a way to ensure the sum of the Softmax deviates only minimally from a provided value [8]. This is critical to reduce the high computational cost of the linear projection before the Softmax activation in inference time, as we are enabled to approximate the normalization term by a constant.

In this way, the posterior probability of a word $p(w|h)$ is approximated as

$$p(w \mid h) = \frac{\exp(v_L(h)^T \cdot a_w)}{Z(h)} \approx \frac{\exp(v_L(h)^T \cdot a_w)}{D}\,, \quad (1)$$

where h denotes the current history, L is the number of hidden layers, $v_L(h)$ and $Z(h)$ are respectively the input vector to the Softmax layer and the corresponding normalization term, a_w is the weight vector for word w, and D is just a constant that can be tuned as described in [8].

A second key idea, particularly concerning TLMs, is to limit the word history size. The strength of self-attentive LMs resides in their capacity to attend to all previous words. This makes them more versatile than RNNs, as they are not required to compress all the previous history into a single vector, which can cause some information loss. However, this implies that the whole history must be processed every time a next word is predicted. This is especially troublesome in the streaming setup: if we allow the history to grow without limits, computational time will increase accordingly. To circumvent this difficulty, the Transformer history size is limited to the n previous words. Therefore, there is no need to store any internal state, alleviating the memory requirements. However, we should compute the output again with each call. Despite TLMs are considered well-optimized performant models, this possible drawback will be assessed during the empirical study.

4. Experiments

4.1. Experimental setup

We evaluate the performance of our streaming decoder and models on the LibriSpeech ASR corpus [9], and on the TED-LIUM release 2 corpus [10]. Acoustic models were trained on the 961 hours for LibriSpeech and the 207 hours provided in TED-LIUM release 2. As development and test data, we used the **-other* for LibriSpeech, with 5.3 hours for dev and 5.1 hours for test, and the **-legacy* ones for TED-LIUM, with 1.6 and 2.6 hours for dev and test, respectively. Regarding the text data, we used the ∼800M text provided for LibriSpeech, whereas for TED-LIUM the data comes from the six provided subsets plus the TED-LIUM training audio transcriptions with up to 230M running words. System vocabularies were restricted to 200K and 153K words for LibriSpeech and TED-LIUM, respectively. Out-of-vocabulary (OOV) ratios were less than 1% in both tasks.

The acoustic models (AM) were trained as follows. First, we trained context-dependent feed-forward DNN-HMMs with three left-to-right states, using the transLectures-UPV toolkit (TLK) [11]. The state-tying schema follows a phonetic decision tree approach [12]. This resulted in 8.3K and 10.8K tied states for LibriSpeech and TED-LIUM, respectively. Feed-forward models were used to bootstrap BLSTM-HMMs [13], trained with TLK and TensorFlow [14]. BLSTMs are composed of 8 layers, with 512 cells per layer and direction, and trained using the cross-entropy criterion. Back-propagation through time was limited to a window size of 50 frames.

For language modelling, we trained TLMs using fairseq [15], with a custom implementation of the VR criterion [8]. We followed the ‘base’ configuration, with 512 cells per layer, a feed-forward of 2048, and 8 attention heads. Only in TED-LIUM, the intersection between the provided vocabulary and the vocabulary of the training set resulted in a smaller output layer (of 144K units). We ran our experiments with three different models per task, corresponding to a different number of Transformer layers: 12, 18, and 24.

In addition, we also explored LM combination by interpolating Transformer models with n-gram and/or LSTM LMs. Regarding n-gram LMs, we used the 4-gram ARPA LM (*fglarge*) provided with the LibriSpeech dataset, while for TED-LIUM we trained a standard Kneser-Ney smoothed 4-gram LM with the same data as [10] using SRILM [16]. A pruned version of these models was used to estimate the static look-ahead tables. LSTM LMs were trained with both the Noise Contrastive Estimation (NCE) [17] and the VR [8] criterions, using the CUED-RNNLM toolkit [18]. They consisted of a 256-unit embedding layer and two LSTM layers of 2048 units. Output Softmax layers have the same number of units as TLMs.

To assess and compare LM performances, we provide perplexities (PPL) computed over the development sets of each task. And, to evaluate overall ASR system quality, we compute Word Error Rate (WER%) figures on the corresponding development and/or test sets. Additionally, we carried out time-measuring experiments for the ASR systems when processing the development sets both under off-line and streaming setups. For the off-line case, we compute Real Time Factor (RTF) values. RTF is defined as the ratio between the time needed by an off-line ASR system to sequentially transcribe the whole development set, and the duration of that set. For the streaming case, we provide mean system latencies. We define latency as the time elapsed between the system receiving the last input frame of an uttered token, and the point in time when the first hypothesis for that token is delivered. These time-measuring experiments were conducted on an Intel Xeon(R) CPU E5-1620@3.50GHz, and a GPU GTX2080Ti with 12GB. The estimation of the scores for the BLSTM, LSTM, and TLM was performed on GPU, while the estimation of the n-gram scores and the rest of the decoding was carried out on CPU.

Experiments are structured as follows. First, Section 4.2 gauges the performance of the different LMs considered in this work, and studies the effect of limiting the history of TLMs. Section 4.3 analyses the ASR performance considering WER and RTF, for different number of Transformer layers and search parameters. Next, Section 4.4 assesses the interpolation of the best performing Transformers with the other LMs. Finally, Section 4.5 compares the performance of the ASR systems, using the best LM combination, in both off-line and streaming conditions, also providing latencies for the latter case.

4.2. Language model evaluation

First, Table 1 shows the PPLs on the development sets for the three types of LM models considered in this work: n-gram, LSTM, and Transformer (12, 18 and 24 layers). As expected, TLMs obtain significantly better PPLs than the other LMs considered. Moreover, increasing the number of Transformer layers provided slight improvements of PPL. This observation leads us to explore whether a streaming ASR system can benefit from them, in terms of WER and latency.

Table 1: *LM perplexities for LibriSpeech and TED-LIUM.*

Model	LibriSpeech	TED-LIUM
n-gram	140.9	117.5
LSTM	72.5	86.7
Transformer 12L	60.7	74.3
Transformer 18L	58.1	72.4
Transformer 24L	56.2	71.0

Next, Figure 1 shows the impact of limiting the history window in the evaluation of TLMs, in terms of PPL (left vertical axis) as a function of the history size (in words), computed over the development sets of LibriSpeech (left plot) and TED-LIUM (right plot). This analysis is performed for Transformer models of 12, 18, and 24 layers. The discontinuous line indicates the percentage of fully-seen sentences (right vertical axis).

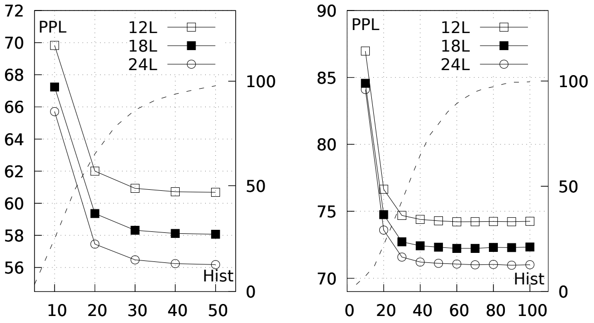

Figure 1: *Transformer LM PPLs (left vertical axis) for different number of layers (in legend), and percentage (right vertical axis) of fully-seen sentences (discontinuous line), as a function of history size, for LibriSpeech (left) and TED-LIUM (right).*

From this figure we realize that PPLs consistently and greatly improve in all cases as we increase the history window size, until we reach size 40, approximately. For history sizes higher than 40, PPL improvements are negligible. In LibriSpeech, this is the expected behaviour, considering that 94% of sentences are shorter than 40. This result is more interesting in TED-LIUM, where a significant number of sentences (35%) are longer than 40 words. This latter fact is more representative of a streaming ASR scenario, for example a long lecture, where most of the history would not be available with any realistic window size. Due to these reasons, we used a history window size of 40 words for the rest of the experiments of the paper.

4.3. ASR systems with Transformer language model

In this section we analyse the performance of hybrid ASR systems with TLMs of varying number of layers, under an offline setup. The idea is to find a good trade-off between quality (WER) and speed (RTF). As we aim to bring these systems to a streaming setup, our search for the best WER-RTF trade-off is limited to those with RTF < 1, to ensure they are able to process input audio streams in real-time. System speed is adjusted by using different search (prune) parameters.

Figure 2 shows WER (vertical axis) and RTF (horizontal axis) curves for ASR systems with TLMs of 12, 18 and 24 layers, computed over the development sets of LibriSpeech (left-most plot) and TED-LIUM (right-most plot). As expected, in both cases, the lowest RTF values are obtained with 12 layers. But, as we are seeking a good WER-RTF balance, the best option is different for each particular task. For LibriSpeech, the LM with 24 layers provides the best quality (5.6% WER) that complies RTF constraints ($\sim$0.9 RTF). On the other hand, for TED-LIUM, the LM with 12 layers is not only the fastest, but also provides the best quality (5.8% WER, $\sim$0.8 RTF).

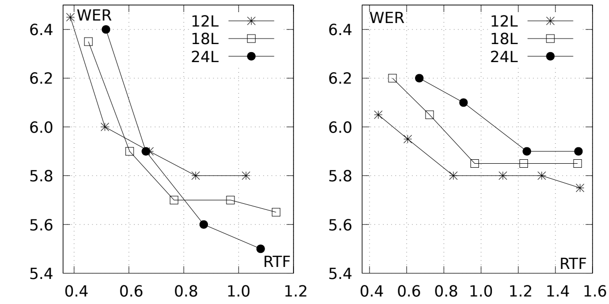

Figure 2: *WER vs RTF of ASR systems with different Transformer LMs, for LibriSpeech (left) and TED-LIUM (right).*

Considering the results of this experiment, we interpret this in the light that WER is specially sensible to PPL changes in the LibriSpeech task. In TED-LIUM, however, this correlation is weaker, and, as such, more exploration of the search space is always preferred, even with a LM with inferior PPL. In other words, time is better spent exploring more prefixes, instead of computing more Transformer layers for each candidate prefix. Moreover, it is important to remark that LibriSpeech provides four times more text data than TED-LIUM, then the improvements of increasing the size of the model are somehow limited. To us, all of the above explains the different behaviour of the two tasks in figure 2. For the remaining experiments, we selected the best-performing TLMs giving RTFs < 1, this is, 24 and 12 layers for LibriSpeech and TED-LIUM, respectively.

4.4. ASR systems with language model combination

In this section we aim to increase overall ASR performance by LM combination, studying the effects of combining the best performing TLMs with the corresponding n-gram and/or LSTM LMs on our streaming decoder. For simplicity and taking into account PPL results, we only tried combinations involving Transformer LMs. LM combinations are done by performing a linear interpolation that minimizes PPL on the development sets. First, we study the impact of these combinations on the PPLs, and then, we analyse how PPL gains are translated into ASR performance, in terms of quality and speed.

Table 2 shows interpolation weights (W%) and PPLs computed for each possible combination over the development set of both tasks. Only the weights of the n-gram (ng) and LSTM (ls) models are shown. Therefore, the remaining (up to 100%) is for the Transformer model.

We observe that LM combination has a positive effect on the PPL, but the magnitude of this effect is diverse. On LibriSpeech, the improvement is very slight, a 3% reduction on PPL considering the three LMs, but on TED-LIUM, the improvement is significant, obtaining a PPL reduction of a 18% with the same combination. Correspondingly, n-gram and LSTM models take lower weights on LibriSpeech (up to 15%), and higher on TED-LIUM (up to 46%).

Next, we analyse how these PPL gains translate to ASR system performance. As LM combination increases the com-

Table 2: *Interpolation weights (W%) and PPLs computed over the development sets of each task for all LM combinations.*

	LibriSpeech		TED-LIUM	
Model	W(%)	PPL	W(%)	PPL
Transformer	–	56.2	–	74.4
+ ng	4	54.9	27	63.2
+ ls	14	54.6	39	68.3
+ ng+ls	2+13	54.4	27+19	61.0

putational complexity of the decoding, we must again explore different ASR system speeds, with different search parameters, to ensure that they comply the RTF < 1 constraint while keeping a good balance with quality. Figure 3 shows WER (vertical axis) and RTF (horizontal axis) curves for all the LM combinations shown in Table 2, computed over the development sets of LibriSpeech (left plot) and TED-LIUM (right plot).

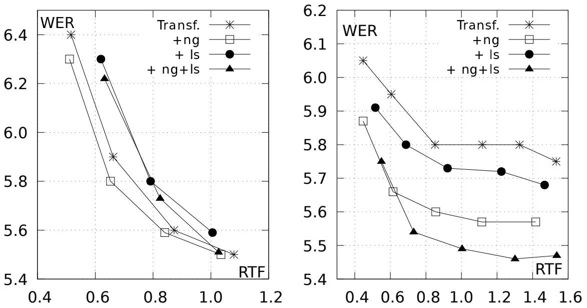

Figure 3: *WER vs RTF of ASR systems with different LM combinations, for LibriSpeech (left) and TED-LIUM (right).*

For LibriSpeech, the interpolation has little effect on the WER, and the main explainer of WER differences are the search parameters. This is aligned with the PPL results, where we observed limited improvements. However, the Transformer + n-gram combination provides slight but consistent improvements in both quality and speed over the baseline TLM, when search parameters are adjusted. For TED-LIUM, the significant improvements seen in PPL due to LM combination translate into a similar fashion on WER figures, being the Transformer + n-gram + LSTM combination the best performing in terms of quality at acceptable RTF rates.

At this point, we select, as final ASR systems, those using the LM combination that exhibits the lowest WER with RTF lower than 1. These are: Transformer + n-gram for LibriSpeech, and Transformer + n-gram + LSTM for TED-LIUM.

4.5. Streaming ASR systems

To conclude our experiments, the two final systems of both tasks are tested under a streaming setup. First, we measure WERs over the development and test sets, and compare them with the performance under an off-line setting, to quantify the expected WER degradation. Then we compare these results with other works, and, finally, we provide system latencies. The streaming setup that provided the best result in [6] was used during these experiments (in particular, an overlapping window of size 50 frames and stride 1 was used for the AM).

Tables 3 and 4 show WER figures of the final ASR systems for LibriSpeech and TED-LIUM, respectively, evaluated on off-line (constrained to RTF¡1) and streaming conditions, plus comparative results with related works. First, we can observe a very small WER degradation when we move from off-line to a streaming setup due to our inability to compute acoustic statistics for normalization from the whole input stream. Second, under streaming conditions, we obtain competitive results even if compared with other works assessed under an off-line setting; and top, state-of-the-art results when compared with other works evaluated under the same streaming conditions.

Table 3: *LibriSpeech results summary*

System	dev	test
Off-line (RTF <1)	5.6	5.9
Streaming	5.9	6.4
Lüscher et al. [19] (Off-line)	4.5	5.0
Moritz et al. [1] (Off-line)	6.0	6.1
Moritz et al. [1] (Streaming)	7.2	7.3
Zhang et al. [2] (Off-line)	–	5.6
Zhang et al. [2] (Streaming)	–	10.0

Table 4: *TED-LIUM results summary*

System	dev	test
Off-line (RTF <1)	5.5	6.2
Streaming	5.7	6.4
Zhou et al. [20] (Off-line)	5.1	5.6

As for latencies, our theoretical latency (0.6s) is dominated by the look-ahead window of 0.5 seconds, plus 0.1 seconds due to batch processing. Actual latencies are slightly higher: our measurements are 0.9±0.4s in LibriSpeech, and 0.8±0.3s in TED-LIUM. We can compare this to a theoretical latency from [1] of 2.2 seconds, and from [2] of 1.1 seconds. As a reference, the UK Office of Communications recommends, to TV broadcasters, a maximum latency of 3 seconds in live subtitling [21].

5. Conclusions and future work

Following our previous work on real-time one-pass decoding with hybrid ASR systems and LSTM language models, in this work we have reported further improvements by replacing LSTMs with Transformer models. Two key ideas have been described to adapt TLMs to our streaming one-pass decoder: the incorporation of a VR term during training and the limitation of the word history size. Empirical results show that TLMs lead to top recognition rates on both tasks, LibriSpeech and TED-LIUM, under the streaming setup.

Although LibriSpeech and TED-LIUM are free, widely used tasks for comparison purposes, they are not well-suited to accurately represent all the issues streaming systems have to face. Accordingly, a first short-term goal is to test our decoder on more realistic tasks, such as the Europarl-ST corpus [22] or the RTVE Database [23].

6. Acknowledgments

The research leading to these results has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement no. 761758 (X5Gon); the Government of Spain’s research project Multisub, ref. RTI2018-094879-B-I00 (MCIU/AEI/FEDER,EU); and the Generalitat Valencianas predoctoral research scholarship ACIF/2017/055.

7. References

[1] N. Moritz, T. Hori, and J. Le, "Streaming automatic speech recognition with the transformer model," in *Proc. of ICASSP*, 2020, pp. 6074–6078.

[2] Q. Zhang, H. Lu *et al.*, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *ICASSP*, 2020, pp. 7829–7833.

[3] H. Miao, G. Cheng, C. Gao, P. Zhang, and Y. Yan, "Transformer-based Online CTC/attention End-to-End Speech Recognition Architecture," *arXiv preprint arXiv:2001.08290*, 2020.

[4] A. Zeyer, P. Bahar, K. Irie, R. Schlter, and H. Ney, "A comparison of Transformer and LSTM encoder decoder models for ASR," in *Proc. of ASRU*, 2019, pp. 8–15.

[5] J. Jorge, A. Giménez, J. Iranzo-Sánchez, J. Civera, A. Sanchis, and A. Juan, "Real-time one-pass decoder for speech recognition using LSTM language models," in *Proc. of InterSpeech*, 2019, pp. 3820–3824.

[6] J. Jorge, A. Giménez, J. Iranzo-Sánchez, J. A. Silvestre-Cerdà, J. Civera, A. Sanchis, and A. Juan, "LSTM-based one-pass decoder for low-latency streaming," in *Proc. of ICASSP*, 2020, pp. 7814–7818.

[7] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, "Language Modeling with Deep Transformers," in *Proc. of InterSpeech*, 2019, pp. 3905–3909.

[8] Y. Shi, W.-Q. Zhang, M. Cai, and J. Liu, "Efficient one-pass decoding with NNLM for speech recognition," *IEEE Signal Processing Letters*, vol. 21, no. 4, pp. 377–381, 2014.

[9] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Proc. of ICASSP*, 2015, pp. 5206–5210.

[10] A. Rousseau, P. Deléglise, and Y. Esteve, "Enhancing the ted-lium corpus with selected data for language modeling and more ted talks." in *Proc. of LREC*, 2014, pp. 3935–3939.

[11] M. del Agua, A. Giménez, N. Serrano, J. Andrés-Ferrer, J. Civera, A. Sanchis, and A. Juan, "The translectures-UPV toolkit," in *Iber-SPEECH*, 2014, pp. 269–278.

[12] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proc. of Workshop on Human Language Technology*, 1994, pp. 307–312.

[13] A. Zeyer, P. Doetsch, P. Voigtlaender, R. Schlüter, and H. Ney, "A comprehensive study of deep bidirectional LSTM RNNs for acoustic modeling in speech recognition," in *Proc. of ICASSP*, 2017, pp. 2462–2466.

[14] M. Abadi, A. Agarwal *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015.

[15] M. Ott, S. Edunov, A. Baevski *et al.*, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proc. of NAACL-HLT*, 2019, pp. 48–53.

[16] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proc. of ICSLP*, 2002, pp. 901–904.

[17] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," *Proc. of ICML*, 2012.

[18] X. Chen, X. Liu *et al.*, "CUED-RNNLM – An open-source toolkit for efficient training and evaluation of recurrent neural network language models," in *Proc. of ICASSP*, 2016, pp. 6000–6004.

[19] C. Lüscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "RWTH ASR Systems for LibriSpeech: Hybrid vs Attention," in *Proc. of InterSpeech*, 2019, pp. 231–235.

[20] W. Zhou, W. Michel, K. Irie, M. Kitza, R. Schlter, and H. Ney, "The RWTH ASR System for Ted-Lium Release 2: Improving Hybrid HMM With SpecAugment," in *Proc. of ICASSP*, 2020, pp. 7839–7843.

[21] United Kingdom Office of Communications, "Measuring live subtitling quality. Results from the first sampling exercise," 2014.

[22] J. Iranzo-Sánchez, J. A. Silvestre-Cerdà, J. Jorge, N. Roselló, A. Giménez, A. Sanchis, J. Civera, and A. Juan, "Europarl-ST: A Multilingual Corpus For Speech Translation Of Parliamentary Debates," in *Proc. of ICASSP*, 2020, pp. 8229–8233.

[23] E. Lleida, A. Ortega, A. Miguel, V. Bazán-Gil, C. Pérez, M. Gómez, A. de Prada, "Albayzin 2018 Evaluation: The IberSpeech-RTVE Challenge on Speech Technologies for Spanish Broadcast Media," *Applied Sciences*, vol. 9, no. 24, 2019.