IberSPEECH 2021
24-25 March 2021, Valladolid, Spain

Sigma-UPM ASR Systems for the IberSpeech-RTVE 2020 Speech-to-Text Transcription Challenge

Juan M. Perero-Codosero[1,2], *Fernando M. Espinoza-Cuadros*[1,2], *Luis A. Hernández-Gómez*[2]

[1]Sigma Technologies S.L.U., Madrid, Spain
[2]GAPS Signal Processing Applications Group, Universidad Politécnica de Madrid, Spain
{jmperero,fmespinoza}@sigma-ai.com, luisalfonso.hernandez@upm.es

Abstract

This paper describes the Sigma-UPM Automatic Speech Recognition (ASR) systems submitted to IberSpeech-RTVE 2020 Speech-to-Text Transcription Challenge. Deep Neural Networks (DNNs) are becoming the most promising technology for ASR at present. Since last few years, traditional hybrid models are being evaluated and compared to other end-to-end ASR systems in terms of accuracy and efficiency. In this challenge, we contribute with two different approaches: a primary hybrid ASR system based on DNN-HMM and two contrastive state-of-the-art end-to-end ASR systems, based on lattice-free maximum mutual information (LF-MMI). Our analysis of the results from the last edition led us to conclude that some adaptation should be accomplished to improve the performance of the systems. In particular, data augmentation techniques and Domain Adversarial Training (DAT) have been applied to the aforementioned approaches. Multi-condition data augmentation applied to our hybrid DNN-HMM models has demonstrated WER improvements in noisy scenarios (about 10% relative). In contrast, results obtained using an end-to-end Pychain-based ASR system are far from our expectations. Nevertheless, we found that when including DAT techniques a relative WER improvement of 2.87% was obtained as compared to the Pychain-based system.

Index Terms: TV shows Speech-to-Text transcription, ASR systems, Hybrid DNN-HMM, end-to-end Deep Learning, Domain Adversarial Training

1. Introduction

State-of-the-art ASR approaches are mainly based on DNNs. Particularly, in traditional hybrid systems acoustic models use the Hidden Markov Model (HMM) state probabilities to train a DNN. These DNN-HMM acoustic models (AM) are combined with other models: pronunciation (PM) and language models (LM). According to some research studies [1] these hybrid-models perform better in many scenarios with small amount of training data.

Nevertheless, hybrid ASR systems have some limitations such as a high complexity associated to the training process of DNN-HMM models. They require phoneme alignments for frame-wise cross entropy and a sophisticated beam search decoder [2]. Furthermore, they usually require strong context-dependency trees to train *chain models* [3].

Trying to overcome these limitations, end-to-end ASR systems have been emerging in the last years. Several approaches appeared such as Connectionist Temporal Classification (CTC) [4], Recurrent Neural Network Transducer (RNN-T) [5] and sequence-to-sequence attention-based encoder-decoder [6, 7]. This trend presents an easy-to-use and easy-to-update pipeline. First, training process has not several stages, only a single model is required instead of the traditional ones (e.g., AM, PM, LM). Second, the continuous advances in deep learning-based technologies have allowed the quick development of powerful open-source libraries for machine learning, such as PyTorch [8] or TensorFlow [9], among others.

The promising results reported for many end-to-end ASR systems depend on the scenarios as well as the availability of datasets. There is an evident gap between these end-to-end systems and hybrid models. Nevertheless, recent developments focused on reducing this gap have achieved good results, as is the case of Pychain [10]. In Pychain, the end-to-end LF-MMI criterion, which is the state-of-the-art for hybrid models in Kaldi [3], is implemented by combining a single stage training and a fully parallelization under PyTorch framework.

Besides that, in order to improve the performance of end-to-end ASR systems, a variety of techniques commonly applied in Deep Learning have been introduced. Data augmentation techniques [11, 12] have been developed to increase the quantity of training data following some criteria to improve model robustness. Thus, a variety of scenarios can be simulated trying to cover more challenging acoustic conditions in a cost-effective way. Other recent deep learning-based technique, namely Domain Adversarial Training (DAT) [13] has been applied to improve ASR performance by learning features invariant to different conditions, such as acoustic variabilities [14, 15], accented speech [16], and inter-speaker feature variability [17].

In this paper, our aim is to contribute to the evaluation of both hybrid and end-to-end ASR systems under the conditions of the IberSpeech-RTVE 2020 Speech-to-Text Transcription Challenge [18]. For this purpose, we firstly report the use of data augmentation techniques to improve our Kaldi-based hybrid ASR system presented in the previous IberSpeech-RTVE edition [19]. Then we evaluate a baseline Pychain system and an improved version of it including DAT. These evaluations on IberSpeech-RTVE Challenge allow us to compare the performance of our systems in a variety of TV shows and broadcast news in specific conditions, noisy environments or challenging scenarios.

2. Architectures

2.1. Primary system: Hybrid ASR

This system is based on the Sigma ASR [19] submitted to the Albayzin-RTVE 2018 Speech-to-Text Challenge [20], where it was in the top-2 of the ranking for both closed and open condition evaluations.

This hybrid ASR system was built using Kaldi Toolkit [1]. The architecture consists of the classical sequence of three modules: acoustic model, pronunciation model and language model.

10.21437/IberSPEECH.2021-23

The acoustic model is based on Deep Neural Networks and Hidden Markov Models (DNN-HMM) following the so-called *chain models* [3].

Our main conclusion from Albayzin-RTVE 2018 results was the need for more robust DNN training looking for accuracy improvements required in the more challenging scenarios (street interviews, game shows, risky sports documentaries...). Environmental robustness of acoustic models has been significantly improved by using multi-condition training data. However, data collection processes are very costly compared to the artificial generation of new training data, which has become a common alternative [11].

Thus, in our current contribution we extended the amount of training data through data augmentation techniques. In particular, we added reverberation to the available training speech data, following the approach presented in [21]. Depending on the expected scenarios and distances, different room impulse responses (RIRs) can be used. The authors sample the room parameters and receiver position in the room and then randomly generate a number of RIRs according to different speaker positions. Basically, three sets of simulated RIRs have been applied: small room (1-10 m), medium room (10-30 m) and large room (30-50 m). The real computation is carried out at the feature extraction level where the original data is mixed with its reverberated copies. The result is a 2-fold training set.

This data augmentation technique has been added to other data augmentation techniques already used in the recipe followed in our previous system, such as volume and speed perturbations [11, 12].

2.2. Contrastive system 1: end-to-end LF-MMI ASR

Aiming to explore the new state-of-the-art end-to-end ASR systems, we have evaluated an alternative to the developed Kaldi-based hybrid ASR system. That is the new end-to-end ASR lattice-free maximum mutual information (LF-MMI) approach [3], which is also used in Kaldi's *chain models*. Thus, we found it interesting to perform a reliable comparison of these two systems based on LF-MMI under RTVE IberSpeech 2020 Challenge scenarios.

This first end-to-end contrastive system is based on PyChain [10], a powerful PyTorch-based implementation which is intended to have an easy-to-use pipeline in which data preparation and final decoding are carried out in Kaldi for efficiency, while data loading and network training are made in PyTorch [8]. It should be noted that it is not necessary any alignment, i.e., HMM-GMM training stage is not required, unlike other systems [22, 23].

Data preparation consists in both feature extraction (40-dimensional MFCC) and the generation of the numerator/denominator graphs stored as Finite State Transducers (FSTs). By following the Kaldi's method for LF-MMI, HMM graphs are used as supervision. The probability distribution function (pdf) is used to estimate the likelihood of an HMM emission [1]. In this case, the network output and the occupation probability are computed from a pdf-index (pdf-id) instead of an HMM state. More details are presented in [10].

Once the data is loaded, the Pytorch model tries to simulate a time-delay neural network (TDNN) [24] by including 1D dilated convolution in addition to batch normalization, ReLU and dropout. This sequence is stacked in this order up to six layers with residual connections. At the end of the sequence, a fully connected layer is added (as described in [10]). From now on this system will be called Pychain-based baseline system.

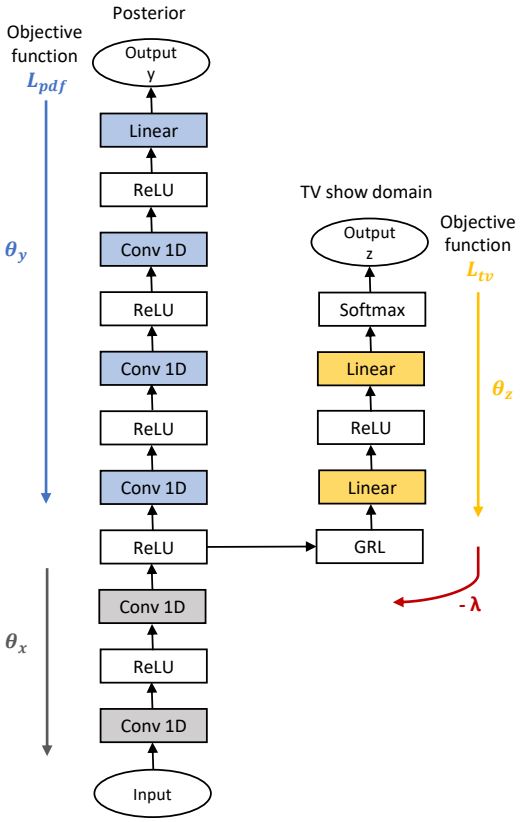

Figure 1: *Architecture of the contrastive system 2. An adversarial branch (TV show classifier) is added to the second layer of the main Pychain architecture (posterior classifier).*

2.3. Contrastive system 2: Domain Adversarial Training

Different acoustic conditions on TV shows can have a negative impact on the Pychain-based baseline performance. In order to reduce this effect, we explore the integration of Domain Adversarial Training (DAT) [13] trying to improve the Pychain-based baseline system. More specifically, in this approach we aim to generate acoustic representations invariant to the domain of the TV show characteristics by using a Domain Adversarial Neural Network (DANN).

For this adversarial architecture, a training dataset denoted as $\{x_i, y_i, z_i\}_N^{i=1}$ is composed of x_i, which are the acoustic features, and y_i, z_i which are the posteriors of the senones and the type of TV show, respectively.

Differently from the Pychain-based baseline system training, in which the acoustic representation is trained so as to minimize the LF-MMI loss function, in DAT the acoustic representations are learned adversarially to the secondary task (i.e., TV show classification). In this way, the domain-dependent information is supressed from the representation as it is irrelevant for the primary task (i.e., posterior classifier).

As it can be seen in Figure 1, the parameters of our adversarial architecture consist of three parts, $\theta = \{\theta_x, \theta_y, \theta_z\}$. Where θ_x denotes the parameters of the first layers of the TDNN used as feature extractor, and θ_y and θ_z denote the parameters of the pdf posteriors and the TV show classifier sub-networks respectively.

A gradient reversal layer (GRL) [13] is placed between the

feature extractor and the TV show classifier. In the forward propagation, GRL keeps the input unchanged and reverses the gradient by multiplying it by a negative coefficient during the backpropagation.

According to [13], for this adversarial training, the objective functions for the pdf classifier L_{pdf} and TV show classifier L_{tv} are defined as:

$$L_{pdf}(\theta_x, \theta_y) = -\sum_{i=i}^{N} \log P(y_i|x_i; \theta_x, \theta_y) \quad (1)$$

$$L_{tv}(\theta_x, \theta_z) = -\sum_{i=i}^{N} \log P(z_i|x_i; \theta_x, \theta_z) \quad (2)$$

DNN acoustic model and the adversarial branch are jointly trained to optimize the following:

$$\min_{\theta_x, \theta_y} \max_{\theta_z} L_{pdf}(\theta_x, \theta_y) - \lambda L_{tv}(\theta_x, \theta_z), \quad (3)$$

where λ is a trade-off parameter between the pdf classification loss L_{pdf} corresponding to the LF-MMI loss, and the TV show classification loss L_{tv}, whose goal is to make deep acoustic features invariant to the domain of the TV show characteristics.

3. Experimental setup

3.1. Datasets

The proposed ASR systems have been evaluated under the Albayzin-RTVE 2020 Challenge conditions. RTVE2020 Database [25] has been provided to all the participants. It is an extension of RTVE2018 Database which contains a collection of TV shows and broadcast news from 2015 to 2019.

RTVE2018 training partition was prepared under a manual supervision process [19] in order to have reliable transcriptions aligned with the speech signal. Two datasets are used for the system evaluation: RTVE_train350 (350 hours of train set) and RTVE_train100 (a RTVE_train350 subset of 100 hours). Validation datasets are a 20% of training data. For testing purposes, several datasets corresponding to 1 hour of duration each were built from RTVE_dev1 and RTVE_dev2 partitions. Moreover, the whole RTVE2020 database was used as test partition for this challenge. It is composed of more than 70 hours of audio and it has been used to present results after the submission.

Additional datasets were added to train the system in open training condition scenario, as used in [19]: VESLIM [26] (103 hours of Spanish clean voice) and OWNMEDIA (162 hours of TV contents, interviews and lectures).

Data augmentation techniques related to hybrid ASR system are carried out by means of the reverberation database[1], which has been described in Section 2.1.

3.2. Training setup

The acoustic model of the hybrid ASR system was trained using RTVE_train350, VESLIM and OWNMEDIA databases, following the SWBD Kaldi recipe for *chain models*. Some modifications were included following the ASpIRE recipe for multi-condition tasks. The rest of the setup is the same as in [19].

End-to-end LF-MMI models (both contrastive systems) were trained by using only RTVE_train100. This relatively small amount of data allows a light training process to test the system performance. We use Pychain-example[2] as a reference by adding some changes in terms of data loading in order to use more than one GPU in parallel.

Data preparation is carried out in Kaldi. In order to convert input features into PyTorch tensors, we use kaldi_io[3] as suggested in [10].

For the adversarial training, note that that the number of the pdf posteriors is $y_i = 62$, and the number of TV shows is $z_i = 13$ because of the different TV shows that RTVE_train100 partition contains. To reduce the bias effect into system training due to unbalanced classes in TV shows, the data was previously merged according to their acoustic characteristics. As a result, four new groups were defined: 0) interviews, 1) TV-game shows, 2) documentaries, and 3) live TV shows. Thus, the labels of the training data for the adversarial branch (i.e., the TV show classifier sub-network) are defined as $z_i = \{0, 1, 2, 3\}$.

In the adversarial architecture, the second hidden layer of the TDNN is used as input to the adversarial branch, which consists of a dense layer of size 384 and ReLU activation function followed by a *softmax* output layer, whose output dimension corresponds to the number of TV shows (i.e., 4). Cross-entropy loss function was used on the adversarial training. To select the optimal trade-off parameter λ, several values were tested. The best performance was achieved for $\lambda = 0.041$.

In addition, all the systems have been evaluated with the same N-gram LM. As described in [19], several corpora were used: subtitles provided in RTVE2018 Database, supervised transcriptions, news between 2015 and 2018, interviews and file captions. A selected lexicon containing 120k words was extracted to train LMs. A 3-gram LM was trained for decoding stage in both hybrid ASR and Pychain-based systems. Instead, a 4-gram LM was only used for the Pychain rescoring stage.

3.3. Resources

Several computation resources have been required to carry out this work. A server with 2 Xeon E5-2630V4, 2.2 GHz, 10C/20 TH and 3 GPUs Nvidia GTX 1080 Ti. Regarding the hybrid ASR system, GPU calculation is necessary for the DNN stage and only CPU mode is used for the HMM stage and final decoding. In case of the Pychain-based systems, only 2 GPUs were used for both training and decoding stages.

4. Results

4.1. Hybrid ASR

The proposal of applying data augmentation techniques to improve the hybrid ASR performance has been fulfilled. The addition of reverberation to our whole training dataset (over 600 hours of speech) has improved the performance in most of the scenarios represented by every TV show set. As shown in Table I, the model applied to CA (*Comando Actualidad*) dataset achieved a relative improvement around 10%, as compared to the baseline system. This might be due to the trained model having learned to model these speech artifacts which can appear in challenging scenarios described in Section 2.1. However, the improvements in the rest of TV shows are not so remarkable (e.g., 20H or LM) because contents are related to daily news with more favorable acoustic conditions.

As we mentioned in [19], the reference master of transcriptions was not reviewed. As usual, we have evaluated the possible impact of transcription errors by means of a new test using an external dataset. It consists of 3.5 hours of TV news

[1] http://www.openslr.org/28/

[2] https://github.com/YiwenShaoStephen/pychain_example

[3] https://github.com/vesis84/kaldi-io-for-python

Table I: *WER(%) on the different datasets for hybrid and end-to-end ASR systems. The duration of each dataset is an hour.*

	20H_dev1	**AP_dev1**	**CA_dev1**	**LM_dev1**	**Mill_dev1**	**LN24H_dev1**
Hybrid ASR						
Kaldi-based baseline [19]	14.88	20.94	49.55	21.44	17.01	24.13
Reverb. data augmentation	**14.76**	21.00	**44.69**	**21.03**	**16.42**	**23.62**
Kaldi-based baseline (RTVE_train100)	16.09	22.32	51.23	23.02	17.70	25.53
End-to-end LF-MMI ASR						
Pychain-based baseline	23.66	33.31	59.34	29.95	35.09	25.08
Domain Adversarial Training	**23.53**	**32.99**	**59.25**	**29.91**	**34.67**	25.16

Table II: *WER(%) on the RTVE2020 test partition for the all the systems. Results were obtained after the submission.*

	RTVE2020_test
Hybrid ASR	
Kaldi-based baseline [19]	31.01
Reverb. data augmentation	27.68
End-to-end LF-MMI ASR	
Pychain-based baseline	40.90
Domain Adversarial Training	42.89

Table III: *Real-time factor (RT) for the different stages carried out in the Pychain-based baseline according to the different datasets.*

Datasets	**Decoding**	**LM rescoring**
20H_dev1	0.033	0.115
AP_dev1	0.035	0.175
CA_dev1	0.225	1.976
LM_dev1	0.092	0.450
Mill_dev1	0.082	0.442
LN24H_dev1	0.383	0.148

broadcasts (similar to 20H). Applying the reverb-trained models of our primary system, we reduced the WER from 8.51% to 7.96%, being our new best results achieved so far.

After the submission, we also evaluated this system on RTVE2020 test partition which contains mostly challenging TV contents. Table II shows data augmentation maintains the WER improvement around 10% relative.

4.2. End-to-end LF-MMI ASR

Our Pychain-based baseline has a good performance in relation to the number of parameter and the easier training process as compared to other end-to-end frameworks. The WER achieved for standard TV news (e.g., 20H, LN24H) are between 23% and 26%, as shown in Table I. These results are within the expected range where commercial ASR systems operate.

In order to compare both hybrid and end-to-end ASR systems, we also trained a hybrid model by using only RTVE_train100 partition. In this case, multi-condition data augmentation has not been applied. Pychain-based system is still far from Kaldi-based hybrid system, with a WER increase of 17% in the worst-case scenario. This gap could be reduced with the application of some data augmentation techniques (e.g., speed or volume perturbation). Despite the fact that augmented data causes a slightly improvement on WER for Pychain system [10], TV shows can contain some acoustic characteristics which could be better modeled by some audio perturbations.

Regarding time consumption, Pychain carries out two stages: a decoding stage and a 4-gram rescoring stage. Table III shows time requirements depends on the characteristics of the test partitions. Less time is required for good acoustic conditions and less challenging scenarios. It makes sense as far as the confusion of the graph model is less complex to transcribe accurately. This behaviour is the same when applying DAT.

On the other hand, we evaluated the effect of learning acoustic representations invariant to TV shows domain. After applying DAT, results in Table I show improvements, in terms of WER, up to 2.87% as compared to the Pychain-based baseline. We are aware that those results are far from the primary hybrid ASR system based on DNN-HMM. Nevertheless, results in Table I give us an insight on how the use of DAT into the end-to-end LF-MMI model can improve its performance in most of the scenarios. It seems that DAT is able to generate deep acoustic features invariant to different TV shows with different acoustic conditions without the need of data augmentation techniques. In addition, Table II shows that applying DAT does not reduce the WER on RTVE2020 test partition. The main reason is DAT alleviates labeled domain conditions in the training dataset. Thus, invariant features were trained without regarding these unseen external factors.

The output transcriptions for this system were not submitted on time due to the required computational demand.

5. Conclusions

In this paper, we have developed both hybrid and end-to-end ASR approaches applying some techniques focused on improving the performance of Text-to-Speech task. Hybrid DNN-HMM models can be adapted to the TV show domain by means of multi-condition data augmentation. The addition of reverberated data to the training data decreases WER significantly (10% relative). A WER of 7.96% is achieved in better conditions. Moreover, we have demonstrated that the lack of data augmentation techniques could be the main reason of the gap between Kaldi hybrid system and Pychain-based system. However, other easy-to-apply techniques, such as DAT, can overcome this gap yielding improvements in end-to-end ASR systems. Acoustic features invariant to the TV show domain are learned by the model achieving a WER improvement of 2.87%. As future work, we believe that adding perturbations to the training data or exploring speech enhancement techniques could help to close the performance gap between Kaldi hybrid system and Pychain-based system. In addition, unsupervised machine learning methods or automatic perceptual speech quality methods could contribute to a more accurate TV shows classification prior to DAT.

6. References

[1] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[2] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," *arXiv preprint arXiv:1805.03294*, 2018.

[3] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi." in *Interspeech*, 2016, pp. 2751–2755.

[4] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

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

[6] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 4945–4949.

[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] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, 2019, pp. 8026–8037.

[9] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[10] Y. Shao, Y. Wang, D. Povey, and S. Khudanpur, "Pychain: A fully parallelized pytorch implementation of lf-mmi for end-to-end asr," *arXiv preprint arXiv:2005.09824*, 2020.

[11] V. Peddinti, G. Chen, V. Manohar, T. Ko, D. Povey, and S. Khudanpur, "Jhu aspire system: Robust lvcsr with tdnns, ivector adaptation and rnn-lms." in *ASRU*, 2015, pp. 539–546.

[12] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[13] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[14] D. Serdyuk, K. Audhkhasi, P. Brakel, B. Ramabhadran, S. Thomas, and Y. Bengio, "Invariant representations for noisy speech recognition," *arXiv preprint arXiv:1612.01928*, 2016.

[15] Y. Shinohara, "Adversarial multi-task learning of deep neural networks for robust speech recognition." in *Interspeech*. San Francisco, CA, USA, 2016, pp. 2369–2372.

[16] S. Sun, C.-F. Yeh, M.-Y. Hwang, M. Ostendorf, and L. Xie, "Domain adversarial training for accented speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4854–4858.

[17] Z. Meng, J. Li, Z. Chen, Y. Zhao, V. Mazalov, Y. Gang, and B.-H. Juang, "Speaker-invariant training via adversarial learning," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5969–5973.

[18] E. Lleida, A. Ortega, A. Miguel, V. Bazán-Gil, C. Pérez, M. Gómez, and A. de Prada, "Albayzin evaluation: Iberspeech-rtve 2020 speech to text transcription challenge," http://catedrartve.unizar.es/reto2020/EvalPlan-S2T-2020-v1.pdf, 2020, [Online].

[19] J. M. Perero-Codosero, J. Antón-Martín, D. T. Merino, E. L. Gonzalo, and L. A. H. Gómez, "Exploring open-source deep learning asr for speech-to-text tv program transcription." in *IberSPEECH*, 2018, pp. 262–266.

[20] E. Lleida, A. Ortega, A. Miguel, V. Bazán-Gil, C. Pérez, M. Gómez, and A. de Prada, "Albayzin 2018 evaluation: the iberspeech-rtve challenge on speech technologies for spanish broadcast media," *Applied Sciences*, vol. 9, no. 24, p. 5412, 2019.

[21] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[22] M. Ravanelli, T. Parcollet, and Y. Bengio, "The pytorch-kaldi speech recognition toolkit," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6465–6469.

[23] D. Can, V. R. Martinez, P. Papadopoulos, and S. S. Narayanan, "Pykaldi: A python wrapper for kaldi," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5889–5893.

[24] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[25] E. Lleida, A. Ortega, A. Miguel, V. Bazán-Gil, C. Pérez, M. Gómez, and A. de Prada, "Rtve2020 database description," http://catedrartve.unizar.es/reto2020/RTVE2020DB.pdf, 2020, [Online].

[26] D. T. Toledano, L. A. H. Gómez, and L. V. Grande, "Automatic phonetic segmentation," *IEEE transactions on speech and audio processing*, vol. 11, no. 6, pp. 617–625, 2003.