JOURNAL OF CRITICAL REVIEWS

ISSN- 2394-5125 VOL 9, ISSUE 04, 2022

Autoregressive Intuitive Scripting and Generative Pre-Training Transformer For Speech

Dr. Y. Narasimha Rao[1], T. Balaji[2], D. Sai Krishna [3], P. Akhil[4], B. Sathwik[5]

[1]Professor and HOD, Department of Computer Science and Engineering

[2,3,4,5] Student, Department of Computer Science and Engineering

[1,2,3,4,5]QIS College of Engineering and Technology, Ongole.

Abstract- Unannotated speech still presents a significant challenge when it comes to learning general representations that can be applied to a wide range of tasks. Here, we describe a new self-supervised goal, autoregressive predictive coding (APC), as a pre-training strategy for the development of general, non-specific speech representations. APC is pre-trained on large-scale unlabeled data before we execute transfer learning experiments on three speech applications: speech recognition, speech translation, and speaker identification. Each of these applications requires distinct knowledge about speech features to work successfully. On all three challenges, extensive studies demonstrate that APC not only outperforms other common representation learning approaches (such as log Mel spectrograms) but significantly reduces downstream labelled data size and model parameters. The usage of Transformers for APC modelling is also investigated and shown to be better than RNNs.

Keywords—Representation learning, self-supervised learning, pre-training, transfer learning, autoregressive modeling

I. INTRODUCTION

Vocal representation learning seeks a transition from waveform and spectrogram surface data into higher-level qualities of speech (e.g. phonetic content, speaker attributes and even emotional signals). For learning representations, unlabeled data, which can be obtained for considerably less money and with more scalability than datasets that need annotation, is particularly attractive for unsupervised or self-supervised aims. An advantage of using unsupervised methods is that the representations learnt are less likely to be biassed toward a certain set of issues [1, 2, 3, 4, 5].

When it comes to pre-training in this work, we're looking for a way to develop a broad and meaningful speech representation that can be used to a range of different speech tasks that may need knowledge about various aspects of speech in order to perform effectively. For example, phonetic content may be more important to voice recognition, but speaker-related applications may be more interested in the speaker's identity.

Keeping as much information about the original signals as feasible in representations is crucial to ensure that downstream models are able to pick the information that is most relevant to the job at hand. As a result, a lot of the currently available representation learning objectives [6, 7, 8, 9, 10] risk omitting information that could be useful for unanticipated downstream tasks because they aim to eliminate noise or speaker variabilities. On the other hand, it has been demonstrated that autoregressive predictive coding (APC) [1] is able to learn representations that preserve information about the original signals, making them more accessible for downstream use, where accessibility is defined as the linear separation of the representations. As a result, APC is an excellent generative pre-training method for enhancing transferability.

Here are the sections of the paper. Section 2 briefly explains the purpose of APC and introduces two kinds of architectures that may serve as its backbone. Using APC for transfer learning is explained in Section 3. Section 4 presents the results of research on voice recognition, speech translation, and speaker identification. Section 5 serves as a wrap-up, highlighting some promising future directions.

II. RELATEDWORKS

[1] The sequential nature of speech is considered in Autoregressive Predictive Coding (APC). APC seeks to forecast information about the next frame. APC is taught to comprehend what a legitimate spectrogram should look like and encode such information in the representations, inspired by the neural language modelling aim for text [11], which predicts the probability of a series of tokens to appear as a valid language. APC comprises an encoder Enc that encodes each frame xi one at a time autoregressively till the current frame xk, and attempts to anticipate a future

frame xk+n that is n steps ahead of xk. Instead than relying on the local regularity of signals, n 1 encourages Enc to infer more overall patterns in speech.
Enc generates an output prediction yi with the same dimensionality as xi at every time step. By right-shifting the input sequence x by n time steps, enc is able to create the target sequence t = (t1; t2;:::; tN) from the anticipated sequence y = (y1; y2;:::; yN).

$$\sum_{i=1}^{N-n} |t_i - y_i|, t_i = x_{i+n}. \qquad (1)$$

APC is self-supervised and may take use of vast amounts of unlabeled data since the training objective is formed from its input.
For processing x = (x1; x2;::; xN), we explore two RNN and Transformer implementations of the encoder Enc [12] for the purpose of auto-regressive processing of the encoder Enc. typical L-layer unidirectional GRUs [13] will be used for the RNN

$$\begin{aligned} \mathbf{h}_0 &= \mathbf{x}, \\ \mathbf{h}_l &= \mathrm{GRU}^{(l)}(\mathbf{h}_{l-1}), \forall l \in [1, L], \\ \mathbf{y} &= \mathbf{W}\mathbf{h}_L, \end{aligned} \qquad (2)$$

Wprojects hL to x's dimensionality from the output of the final RNN layer W. Training parameters for an RNN-based APC include fW, GRU(1), and GRU(L)g.
Consider a stack of L identical decoder blocks from the original design [12] in the Transformer, as in [14, 15]. Multi-headed self-attention over the input sequence is followed by a position-wise feedforward layer that produces the input for the following block. Positional information for the model is encoded using the sinusoidal positional encodings, which do not introduce new parameters, in accordance with [12].

$$\begin{aligned} \mathbf{h}_0 &= \mathbf{W}_{\mathrm{in}}\mathbf{x} + P(\mathbf{x}), \\ \mathbf{h}_l &= \mathrm{TRF}^{(l)}(\mathbf{h}_{l-1}), \forall l \in [1, L], \\ \mathbf{y} &= \mathbf{W}_{\mathrm{out}}\mathbf{h}_L, \end{aligned} \qquad (3)$$

an affine mapping between the dimensions of an object and its function,Win Affinity for both the Transformer hidden state and Wout returns the dimensionality of the final Transformer output hL to x. A Transformer-based APC's set of trainable parameters is as follows: fWin;Wout;TRF(1);:::;TRF(L)g;::: Win and Win are tied in practise.
Setting Win =WT eliminates it. out of a sense of normalcy.

III. PROPOSED SYSTEM ARCHITECTURE

For APC training, we use the LibriSpeech corpus (only the speech part) [16]. There are 360 hours of audio generated by 921 speakers in the train-clean-360 subset, which is utilised in this example 80-dimensional log-Mel spectrograms (normalised to zero mean and unit variance for each speaker) are the input characteristics. We also look at the influence of various n values on our results (Equation 1). In our case, the log Mel spectrograms, we use an APC feature extractor Enc to transform the surface features of the training dataset into a higher-level representation, which we then use to create the new dataset f(Enc(xj); cj)gSj =1). This dataset is then used to train the APC feature extractor Enc on the downstream labelled dataset f(xj; cj)gSj =1. Depending on the job, cj may either be a series of values or a single value.
Enc(x) = hL in Equations 2 and 3 represents the extracted representations from RNNs and Transformers, although there are superior ways that aggregate the internal representations across all layers [17].
After the model has been trained, there are two options: either maintain Enc frozen and just optimise the model, or change Enc so that the extracted representations are more suited to your target job. Section 4 focuses on both techniques. APC is built around two architectural pillars, both of which were discussed in further detail in Section 2. In Equation 2, the RNN (R-APC) and Transformer (T-APC) are designated as R- and T-APC. To implement R-APC, we use GRUs with 512 hidden units in a 4-layer unidirectional structure. Between two successive layers, we use residual connections [18] after [1].
T-APC uses a 4-layer decoder-only Transformer with a hidden size of 512 and an 8-headed self-attention module followed by a 1-layer MLP and a GELU activation function [19]. Each layer has 2048 hidden units. There are 100

epochs of training for R-APC and T-APC using Adam [20] in which 32 batches and an initial learning rate of 103 are used. Specifically, we compare APC to two newly suggested self-supervised representation learning objectives: CPC and PASE.

In both CPC and APC, information about a future frame xk+n is predicted based on the previous frame's history H = xk+n (x1; x2; :::; xk). To learn representations that are most discriminative between xk+n and a collection of randomly selected frames fxg, CPC instead tries to train representations that include information that is most directly predictive of xk+n given H through regression. The information conveyed in the representations will be heavily influenced by the origin distribution from which fxg are selected. There is a good chance that speaker information will be lost when f and xg originate from the same utterance as xk+n. However, because of its lack of adaptability for learning generic representations, CPC may not be an acceptable generative pre-training technique for problems where the kind of important information is known (such that sampling strategy may be selected accordingly).

CPC is mostly implemented in accordance with [6], with a few alterations detailed in [1]. We use LibriSpeech's train-clean-360 subset to teach APC as well.

Using the input signals as learning targets, PASE trains an extractor of features by jointly maximising numerous self-supervised goals. For transfer learning, a more generic representation is ideal since it incorporates past knowledge from each assignment.

The difficulty of concurrently maximising several goals means that we are unable to use train-clean-360 to train our own PASE. PASE has been pre-trained and published by the authors, therefore we will utilise that model. These models were created by randomly selecting utterances from a pool of around 1,000 hours of LibriSpeech audio made by 2,484 speakers and then training them on approximately 15 seconds of training material from that pool, according to the authors [3].

Additionally, we train APC and CPC with around 10 hours of random selections of audio from Train Clean 360.

It's worth noting that PASE has seen more presenters, but each speaker also has a smaller quantity of training material. If a model (e.g., CPC10) is trained on just 10 hours of audio, we add a subscript 10.

Using automated speech recognition as a starting point, the rest of our findings are summarised in the table below (ASR).

IV. RESULTS AND DISCUSSION

The Wall Street Journal (WSJ) [21] corpus is used for our ASR studies. Training takes up 90 per cent of si284 (72 hours), while development takes up the remaining 10 per cent. WER is reported on dev93. An encoder and a decoder make up the ASR model we utilise, which is an end-to-end, sequence-to-sequence with attention architecture [22]. Input features are downsampled using two convolutional layers, followed by a four-layer bidirectional 256-dim GRU network in the encoder.

Decoding is a one-layer, 256-diameter GRU network. Adam is used to train the seq2seq model for 100 epochs with a batch size of 16 and a learning rate of 103. With a beam size of 5, the decoding process is carried out via the beam search. When log Mel spectrograms are used as input features, the baseline WER is 18.3. APC (n in Equation 1) and a transfer learning strategy (updating pre-trained APC weights) are the focus of the first experiment, which is summarised in Table 1. Besides the seq2seq model, we consider the situation when APC is set up and trained from start.

Table 1 shows that, independent of the transfer learning strategy, there is a sweep spot when n is varied for R-APC and T-APC. For a small n, APC can exploit local smoothness in the spectrograms to predict the target future frame (since, for a small n, xk can be very similar to xk+n) and thus does not need to learn to encode information useful for inferring more global structures; for a large n, however, the prediction task becomes too difficult for APC to generalise across the training set. In terms of R-APC and T-APC, the optimal n for each is 3. APC Frozen (*-APC Frozen) works better than finetuning APC weights (*-APC Finetuned) for all n, unexpectedly, but the latter still beats the baseline. Even lower than the baseline performance is training APC from scratch using the seq2seq model (*-APC Scratch). WER is lowered by more than a quarter from 18.3 to 13.7 using APC transfer learning.

We use R-APC Frozen with n = 3 and T-APC Frozen with n = 5 for the remainder of the studies.

Additionally, transfer learning may be effective in lowering the amount of the downstream dataset and model required to achieve the same performance as an existing model on a standard dataset. Automatic feature extraction, it is assumed, may be learned without starting from scratch if past information is used. A smaller model with competitive performance may help to alleviate the challenge of having limited computational or storable resources in low-resource languages with a limited number of training pairings. APC transfer learning has shown to be

successful in each of these areas. With increasing quantities of labelled data, we compare APC to other feature extractors in Table 2. This implies we just need 72 1/16 = 4:5 hours of si284 time for training, for example. We discover that cutting the training size in half increases WER significantly for all input characteristics. It is clear that RAPC and T-APC consistently beat log Mel in all ratios, and this difference widens as training size shrinks. R-APC and T-APC outperform log Mel trained on the whole set (18.3) even when only half of si284 is used for training (16.4 vs. 18.3).
By utilising just half the training data log Mel does, T-APC consistently beats it.
Using additional pre-training data is clearly beneficial, as we can see in Table 2 by comparing the bottom half (where feature extractors are trained on merely 10 hours of audio) with the top half.
To the extent that having more pre-training data leads to higher transfer learning outcomes, this discovery is in line with current NLP work [23, 24]. Finally, we can conclude that APC outperforms CPC and PASE on the majority of the time. However, when just 1/8 of the si284 is accessible, PASE is marginally better than T-APC10, although it is still inferior to R-APC10. The size of the downstream model is the next thing we look at, and here is where transfer learning comes in. Table 3 shows the results of building the encoder in the seq2seq model with various numbers of GRU layers 2 f1; 2; 3; 4g. APC10 and *-APC always outperform other features when running on identical layers. Utilizing only two layers of T-APC, the model performs as well as log Mel using four layers (18.6 vs. 18.3), demonstrating the efficiency of APC transfer learning for lowering downstream model size. Automated speech translations (AST) are the second of our two projects, and the purpose is to convert spoken language into textual form. This endeavour requires the use of an English-to-French translation dataset [25] supplemented by the LibriSpeech corpus [16]. An English waveform and its French translation are included in each pair of data. On the dev and test sets, we provide BLEU scores [27] based on the percentage of the original training set that was used for training, which was around 100 hours of audio. AST is an RNN-based, end to end seq2seq with attention architecture similar to [28].. The dev and test sets' BLEU baseline scores are 12.5 and 12.9 when utilising log Mel as an input feature.
We use the cascaded system's performance in [28] as a benchmark. To do this, the cascaded system uses an ASR module to convert the input voice to text, followed by an MT module to perform the actual translation. To train a cascaded system, intermediate audio transcriptions in the original language are required, although this provides as a firm baseline. A newly suggested Transformer-based end-to-end AST model (named S-Transformer [29]), which has been found to outperform RNNs, is also included in this study. Regardless of how much training data is used and the kind of Enc, APC always beats log Mel, CPC, and PASE on both the dev and test sets of data. We also outperform the cascaded system (13.8) using our RNN-based model, which has T-APC properties (14.3). (14.6). It is our last task, speaker identification (SID), that investigates how much transferrable speaker information is retained by the representations taught through various aims. For our SID research, we rely on the Wall Street Journal (WSJ) as a source.
For training, we used 80% of si284; for development, 10%; and for testing, another 10%. The challenge is comparable to a classification issue with 259 speakers. Feature values are supplied into a 1-layer GRU network, followed by a Softmax layer, which is applied to the output of the previous time step and optimised by minimising the negative log-likelihood throughout the training set. If just one utterance per speaker is available for training, we look at how this affects our ability to train our models. For example, a mobile device's voice application may need to rapidly adapt to user-specific characteristics with only a few input samples in order to improve user experience. Exploring such one- or few-shot learning situations is particularly intriguing since it's closer to the actual world.
With respect to transferrable speaker information, we can observe from Table 5 that APC representations are virtually always superior to all the other qualities, no matter how many speakers we train with. When it comes to one-shot learning, T-APC outperforms log Mel by almost two to one (17.6 to 8.7).

Table 1: ASR results (WER $\downarrow$) of APC with varying n during pre-training and different transfer learning approaches (Frozen vs. Fine-tuned). log Mel is the baseline that uses log Mel spectrograms as input features. The best transfer learning result is marked in bold.

Features	n					
	1	2	3	5	10	20
log Mel			18.3			
R-APC Scratch			23.2			
R-APC Frozen	17.2	15.8	15.2	16.3	17.8	20.9
R-APC Finetuned	18.2	17.6	16.9	18.2	19.7	21.7
T-APC Scratch			25.0			
T-APC Frozen	19.0	16.1	14.1	**13.7**	15.4	21.3
T-APC Finetuned	22.4	17.0	15.5	14.6	16.9	23.3

Table 2: ASR WER results with varying amounts of training data randomly sampled from `si284`. Feature extractors pre-trained with just 10 hours of LibriSpeech audio are denoted with a subscript 10.

Features	Proportion of `si284`					
	1	1/2	1/4	1/8	1/16	1/32
log Mel	18.3	24.1	33.4	44.6	66.4	87.7
CPC	20.7	28.3	38.8	50.9	69.7	88.1
R-APC	15.2	18.3	24.6	35.8	49.0	66.8
T-APC	13.7	16.4	21.3	31.4	43.0	63.2
PASE_{10}	20.8	26.6	32.8	42.1	58.8	78.6
CPC_{10}	23.4	30.0	40.1	53.5	71.3	89.3
R-APC_{10}	17.6	22.7	28.9	38.6	55.3	73.7
T-APC_{10}	18.0	23.8	31.6	43.4	61.2	80.4

Table 3: ASR WER results using different numbers of GRU layers for the encoder in the ASR seq2seq model.

Features	Number of encoder layers			
	1	2	3	4
log Mel	28.8	23.5	20.8	18.3
CPC	34.3	29.8	25.2	23.7
R-APC	26.2	20.3	17.6	15.2
T-APC	25.2	18.6	15.8	13.7
PASE_{10}	29.4	25.7	22.5	20.8
CPC_{10}	35.8	31.3	26.0	24.4
R-APC_{10}	27.6	22.3	19.6	17.6
T-APC_{10}	28.1	23.2	20.6	18.0

Table 4: Speech translation results. BLEU scores (↑) are reported. We also include the results of the cascaded system (ASR + MT) reported in [28] and the S-Transformer model reported in [29]. Only the results on the test set are available for these two approaches.

Methods	dev	test
Cascaded	-	14.6
S-Transformer	-	13.8
log Mel	12.5	12.9
CPC	12.1	12.5
R-APC	13.5	13.8
T-APC	13.7	14.3
$PASE_{10}$	12.0	12.4
CPC_{10}	11.8	12.3
$R\text{-}APC_{10}$	13.2	13.7
$T\text{-}APC_{10}$	12.8	13.4

Table 5: Speaker ID results. Accuracies (↑) are reported.

Features	Number of utterances per speaker seen in training					
	1	5	10	20	50	full (130 in avg.)
log Mel	8.7	43.7	60.4	70.5	87.4	96.1
CPC	13.0	45.5	65.8	75.9	89.3	96.5
R-APC	17.2	56.9	73.3	87.4	95.1	99.0
T-APC	17.6	58.6	74.4	87.8	96.3	99.1
$PASE_{10}$	12.5	48.6	64.8	79.6	92.6	96.7
CPC_{10}	11.7	44.9	63.2	74.6	88.3	95.8
$R\text{-}APC_{10}$	14.3	54.4	72.3	87.1	95.0	98.9
$T\text{-}APC_{10}$	13.5	49.2	70.5	82.8	92.4	98.0

V. FUTURE SCOPE AND CONCLUSION

Using a variety of different speech tasks, we show that APC is a useful generative pre-training goal for transfer learning. An RNN in [1] was shown to be less successful than a Transformer in modelling APC. APC representations consistently and, in most cases, considerably outperform log Mel spectrograms and representations learnt by other aims, such as CPC [6] and PASE [3], on ASR, speech translation, and speaker identification (SID). APC representations are shown to be the most efficient way for lowering the amount of downstream labelled data and model parameters when compared to other comparison methods. Future research might go in a lot of different areas. When training on a downstream dataset, we've found that keeping APC weights frozen is preferable than changing them. The latter, on the other hand, is more suited for transfer learning since it tailors the extracted representations to the intended task. Fine-tuning procedures [30] that are more advanced might be used. The Transformer model's backbone design may be enhanced by altering the way positional information is injected [31, 32]. Training APC on more unlabeled data may also help improve transfer learning outcomes, as shown by recent NLP research [23, 24]. APC might also be used in other speech applications, such as speech synthesis, where pre-training and transfer learning have previously been successful [33, 34, 35, 36, 37].

REFERENCES

[1] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass, “An unsupervised autoregressive model for speech representation learning,” in Interspeech, 2019.

[2] Jan Chorowski, Ron Weiss, Samy Bengio, and A¨aron van den Oord, “Unsupervised speech representation learning using wavenet autoencoders,” IEEE/ACM TASLP, vol. 27, no. 12, pp. 2041–2053, 2019.

[3] Santiago Pascual, Mirco Ravanelli, Joan Serr`a, Antonio Bonafonte, and Yoshua Bengio, “Learning problem-agnostic speech representations from multiple self-supervised tasks,” in Interspeech, 2019.

[4] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, “Wav2vec: Unsupervised pre-training for speech recognition,” in Interspeech, 2019.

[5] Herman Kamper, “Truly unsupervised acoustic word embeddings using weak top-down constraints in encoder-decoder models,” in ICASSP, 2019.

[6] Aaron van den Oord, Yazhe Li, and Oriol Vinyals, “Representation learning with contrastive predictive coding,” arXiv preprint arXiv:1807.03748, 2018.

[7] Benjamin Milde and Chris Biemann, “Unspeech: Unsupervised speech context embeddings,” in Interspeech, 2018.

[8] Yu-An Chung and James Glass, “Speech2vec: A sequenceto- sequence framework for learning word embeddings from speech,” in Interspeech, 2018.

[9] Yi-Chen Chen, Sung-Feng Huang, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee, “Phonetic-and-semantic embedding of spoken words with applications in spoken content retrieval,” in SLT, 2018.

[10] Wei-Ning Hsu, Yu Zhang, and James Glass, “Unsupervised learning of disentangled and interpretable representations from sequential data,” in NIPS, 2017.

[11] Tom´asˇ Mikolov, Martin Karafia´t, Luka´sˇ Burget, Jan Cˇ ernocky`, and Sanjeev Khudanpur, “Recurrent neural network based language model,” in Interspeech, 2010.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, et al., “Attention is all you need,” in NIPS, 2017.

[13] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” in SSST, 2014.

[14] Peter Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, et al., “Generating wikipedia by summarizing long sequences,” in ICLR, 2018.

[15] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, “Improving language understanding by generative pre-training,” Tech. Rep., OpenAI, 2018.

[16] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, “Librispeech: An ASR corpus based on public domain audio books,” in ICASSP, 2015.

[17] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, “Deep contextualized word representations,” in NAACL-HLT, 2018.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in CVPR, 2016.

[19] Dan Hendrycks and Kevin Gimpel, “Gaussian error linear units (gelus),” arXiv preprint arXiv:1606.08415, 2016.

[20] Diederik Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” in ICLR, 2015.

[21] Douglas Paul and Janet Baker, “The design for the wall street journal-based CSR corpus,” in Speech and Natural Language Workshop, 1992.

[22] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, “Attention-based models for speech recognition,” in NIPS, 2015.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, “BERT: Pre-training of deep bidirectional Transformers for language understanding,” in NAACL-HLT, 2019.

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, et al., “RoBERTa: A robustly optimized BERT pretraining approach,” arXiv preprint arXiv:1907.11692, 2019.

[25] Ali Kocabiyikoglu, Laurent Besacier, and Olivier Kraif, “Augmenting Librispeech with French translations: A multimodal corpus for direct speech translation evaluation,” in LREC, 2018.

[26] Yu-An Chung, Wei-Hung Weng, Schrasing Tong, and James Glass, “Towards unsupervised speech-to-text translation,” in ICASSP, 2019.

[27] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, “BLEU: A method for automatic evaluation of machine translation,” in ACL, 2002.
[28] Alexandre B´erard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin, “End-to-end automatic speech translation of audiobooks,” in ICASSP, 2018.
[29] Mattia Di Gangi, Matteo Negri, and Marco Turchi, “Adapting Transformer to end-to-end spoken language translation,” in Interspeech, 2019.
[30] Jeremy Howard and Sebastian Ruder, “Universal language model fine-tuning for text classification,” in ACL, 2018.
[31] Abdelrahman Mohamed, Dmytro Okhonko, and Luke Zettlemoyer, “Transformers with convolutional context for ASR,” arXiv preprint arXiv:1904.11660, 2019.
[32] Kazuki Irie, Albert Zeyer, Ralf Schl¨uter, and Hermann Ney, “Language modeling with deep Transformers,” in Interspeech, 2019.
[33] Yu-An Chung, Yuxuan Wang, Wei-Ning Hsu, Yu Zhang, and RJ Skerry-Ryan, “Semi-supervised training for improving data efficiency in end-to-end speech synthesis,” in ICASSP, 2019.
[34] Yuan-Jui Chen, Tao Tu, Cheng-Chieh Yeh, and Hung-Yi Lee, “End-to-end text-to-speech for low-resource languages by cross-lingual transfer learning,” in Interspeech, 2019.
[35] Tomoki Hayashi, Shinji Watanabe, Tomoki Toda, Kazuya Takeda, Shubham Toshniwal, and Karen Livescu, “Pre-trained text embeddings for enhanced text-to-speech synthesis,” in Interspeech, 2019.
[36] Wei Fang, Yu-An Chung, and James Glass, “Towards transfer learning for end-to-end speech synthesis from deep pre-trained language models,” Tech. Rep., Massachusetts Institute of Technology, 2019.
[37] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, et al., “Transfer learning from speaker verification to multispeaker text-to-speech synthesis,” in NeurIPS, 2018.