Evolved Speech-Transformer: Applying Neural Architecture Search to End-to-End Automatic Speech Recognition

Jihwan Kim, Jisung Wang[*] *Sangki Kim, and Yeha Lee*

VUNO Inc., 507, Gangnam-daero, Seocho-gu, Seoul

jhkim@vuno.co, jisung.wang@navercorp.com, sangki.kim@vuno.co, yeha.lee@vuno.co

Abstract

Neural architecture search (NAS) has been successfully applied to finding efficient, high-performance deep neural network architectures in a task-adaptive manner without extensive human intervention. This is achieved by choosing genetic, reinforcement learning, or gradient -based algorithms as automative alternatives of manual architecture design. However, a naive application of existing NAS algorithms to different tasks may result in architectures which perform sub-par to those manually designed. In this work, we show that NAS can provide efficient architectures that outperform maually designed attention -based arhitectures on speech recognition tasks, after which we named Evolved Speech-Transformer (EST). With a combination of carefully designed search space and Progressive dynamic hurdles, a genetic algorithm based, our algorithm finds a memory-efficient architecture which outperforms vanilla Transformer with reduced training time.

Index Terms: neural architecture search, end-to-end speech recognition, transformer, deep learning

1. Introduction

Neural architecture search (NAS) [1] aims at automatically finding neural network (NN) architectures which had been designed manually. Typical approaches formulate the task as an optimization problem and solve the complicated problem using genetic or reinforcement learning algorithms. Alternatively, differentiable approaches relax the search space into differentiable candidates, admitting gradient-based algorithms in solving the task. A remarkable property of NAS is its transferability across datasets; an architecture found by NAS on some dataset often performs well when evaluated on different dataset with same task.

While most successful NAS techniques have been attributed to finding architectures fit for computer vision tasks, recent interest in applying NAS to sequence-to-sequence (seq2seq) models has emerged. Such applications include finding optimal seq2seq architectures for neural machine translation, speech synthesis, and speech recognition. However, architectures found for automatic speech recognition (ASR) tasks require additional accoustic knowledge including phone sets and dictionaries. On the other hand, end-to-end ASR aims at processing and translating acoustic signals into written text using only paired acoustics, and do not require additional domain knowledge.

Successful NN-based solutions are typically based on encoder-decoder networks, with three types of manually designed deep neural networks (DNNs) mainly dominating the field: Connection temporal classification (CTC) [2, 3, 4, 5], Recurrent neural network transducer (RNN-T) [6, 7], and Listen, attend, and spell (LAS) [8] models. However, recurrent networks require excessive training time and memory, calling for alternative solutions to end-to-end ASR. With the advent of Transformer [9], architectures with multi-head feed-forward full attention and parallel-in-time computation established its grounds due to its outstanding performance while exhibiting desirable properties such as an intuitive architecture and reduced training time. On a standard ASR dataset such as Wall Street Journal (WSJ) or Librispeech [10], the Speech-Transformer [11] reached an adequate performance in only a fraction of previously required training time outperformed recurrent models.

Inspired by the Transformer blocks, Evolved Transformer (ET) [12] demonstrated substantial performance improvements to the manually designed transformer-based modules. With the support of Progressive dynamic hurdles (PDH), an evolutionary algorithm based on survival of the fittest, and a search space designed to reflect recent advances in feed-forward seq2seq models, it was able to find the model suitable for neural machine translation, which is a task that requires extensive computations.

In this work, we designed a search space apt for speech recognition and applied PDH to find a memory-efficient architecture which performs well across standard benchmark datasets with reduced training time. The model found in our experiments outperformed the original Transformer on standard ASR benchmarks: WSJ and Zeroth. Our final model achieved 0.6% performance improvement compared to Transformer with 26.1% less parameters on average. An architecture attained with the WSJ dataset using our algorithm was also found to perform well on the Zeroth dataset, confirming the superior transferability of NAS even on end-to-end ASR tasks.

2. Method

2.1. Transformer

The Transformer is based on the encoder-decoder architecture: the encoder transforms a feature sequence $\mathbf{x} = (x_1, ..., x_T)$ to a hidden representation $\mathbf{h} = (h_1, ..., h_L)$. Given h, the decoder then generates an output sequence $\mathbf{y} = (y_1, ..., y_S)$ one character at a time. At each step, the decoder consumes the previously emitted characters as additional inputs when emitting the next character. However, as a no-recurrence seq2seq model, the Transformer differs from recurrent seq2seq models mainly on two aspects: Firstly, both the encoder and decoder are composed of multi-head attention and position-wise 1D-convolutional networks rather than RNNs. Secondly, the encoder outputs h are attended by each decoder block respectively, as shown in Figure 1, replacing the one-step intermediary attention of recurrent seq2seq models.

An attention function maps a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the

[*]Currently working at NAVER Corp.

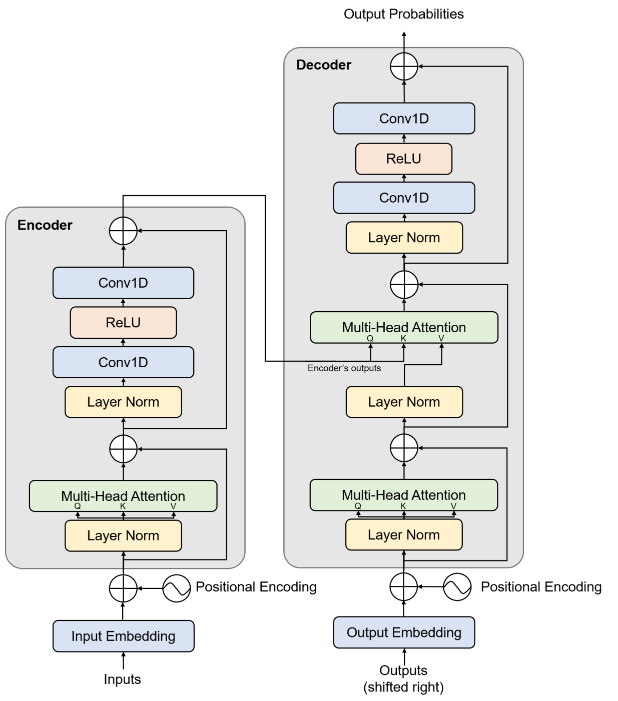


Figure 1: *Architecture of Transformer*

values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. Scaled dot-product attention is adopted as the basic attention function in the Transformer which describes (1).

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}}) \tag{1}$$

Where the dimension of query Q and key K are the same, which are d_k, and dimension of value V is d_v.

Instead of performing a single attention function, the Transformer employs the multi-head attention (MHA) which projects the queries, keys and values h times with different, learned linear projections to d_k, d_k and d_v dimensions. On each of these projected versions of queries, keys and values, the basic attention function is performed in parallel, yielding dv-dimensional output values. These are concatenated and projected again, resulting in the final values. The equations can be represented as below.

$$head_i = Attention(QW_i{}^Q, KW_i{}^K, VW_i{}^V) \tag{2}$$

$$MultiHead(Q,K,V) = Concat(head_1, ..., head_h)W^O \tag{3}$$

Where the projections are parameter matrices $W_i{}^Q \in \mathbf{R}^{d_{model} \times d_k}$, $W_i{}^K \in \mathbf{R}^{d_{model} \times d_k}$, $W_i{}^V \in \mathbf{R}^{d_{model} \times d_v}$, $W_i{}^O \in \mathbf{R}^{hd_v \times d_{model}}$, where h is the number of heads, and d_{model} is the model dimension.

The architecture of the Transformer is shown in Figure 1, which stacks MHA and 1D-convolution layers for both the encoder and decoder. The encoder is composed of a stack of N identical layers. Each layer has two types of sub-layers. The first is a MHA, and the second is 1D-convolution. Residual connections are followed by end of typical sub-layers. The decoder is similar to the encoder except inserting a sub-layer to perform a MHA over the output of the encoder stack. To prevent leftward information flow and preserve the auto-regressive

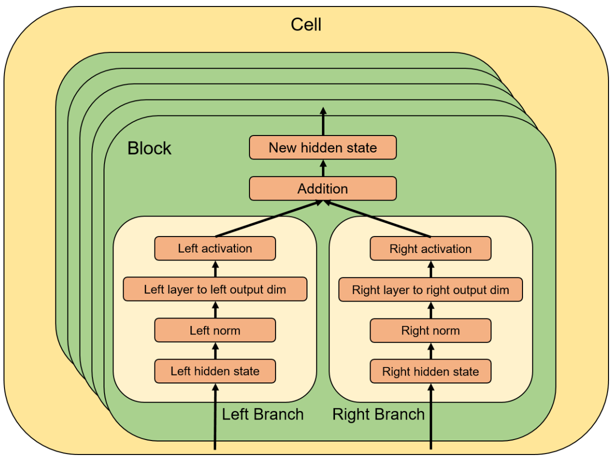


Figure 2: *Cell Search Space*

property in the decoder, the self-attention sub-layers in the decoder mask out all values corresponding to illegal connections. In addition, positional encodings [9] are added to the input at the bottoms of these encoder and decoder stacks, which inject some information about the relative or absolute position in the sequence to make use of the order of the sequence.

In the Speech-Transformer, speech feature such as spectrogram or Mel-frequency cepstral coefficients (MFCC), is put as an input value, and text as an output.

2.2. Search Space

Our search space design is mainly inspired by NASNet [13] for its simplicity and efficiency. Every candidate architecture is composed of an encoder and decoder cell, where each cell comprises N_e and N_d blocks respectively as depicted in Figure 2. A single block contains two branches taking different hidden states as respective inputs and adds the output of hidden states.

Each branch is categorized into five major components: input, normalization, layer, output dimension d_{ff}, and activation function. Dimension mismatch which may occur when branch outputs have different relative output dimensions and is accounted for by applying padding before adding. The Transformer can thus be found in principle, using a NAS algorithm, as it is equivalent to a composition of encoder and decoder cells with 14 stacked blocks each.

To reduce the search space which directly affects training time, we imposed an additional constraint that candidate models must have parameters amounting less than 3M, the number of parameters in our baseline model.

2.3. Progressive dynamic hurdles

Progressive Dynamic Hurdles (PDH) is an instantiation of evolutionary architecture search proposed in [14]. Initially, c child models are trained for s_0 iterations and are evaluated on a validation set to obtain each model's performances $p_0 = (p_{0,1}, \ldots, p_{0,c})$. Models which achieve performance $p_{0,i} \geq \bar{p}_0$ are then grouped into a hurdle H_0, where $\bar{p}_0 = \frac{1}{c}\sum_{i=1}^{c} p_{0,i}$ is the mean performance over the c models. This process is then repeated, where at each step $t \geq 1$, models in the previous hurdle H_{t-1} are trained for s_t additional iterations and $c - |H_{t-1}|$ new models are trained for s_0 iterations, assigning models which outperform the mean performance $\bar{p}_t$ a hurdle

H_t and discarding those that performed poorly. PDH is terminated after training T steps when the total number of iterations $\sum_{t=0}^{T} s_t$ hits a predefined maximum number of steps S, in other words, PDH stops when $\sum_{t=0}^{T} s_t \geq S$.

3. Experiments

3.1. Datasets

We evaluated the performance of the proposed algorithm against a strong baseline on two ASR datasets: Wall Street Journal (WSJ), which contains 80 hours of clean read English speech and English scripts written using several non-linguistic symbols, and Zeroth [15], an open source project for Korean speech recognition which was developed by a Kaldi [16]-based Korean ASR open source project called Zeroth project, consisting of 95.7 hours of Korean speech data and Korean scripts with no other non-linguistic symbols. All experiments are conducted using 80-dimensional log-Mel filterbank features, computed with a 25ms window and shifted every 10ms, and were normalized using sample mean and standard deviations computed on each training partition.

For the WSJ dataset, all models were trained on si-284, validated on dev '93, and tested on eval '92. Tokens in the WSJ dataset consist of 26 lowercase alphabets, apostrophe, period/space, noise, and end-of-sequence characters totaling 31 classes.

The Zeroth dataset includes only a training and test set, so models were validated using the test set. Zeroth contains a morpheme-based segmenter called morfessor [17] as well as transcribed audio datasets. We used the morphologically segmented text by using given morfessor model. Text tokens in the Zeroth data were extracted in unigram units through sentencepiece [18] totaling 6,000 tokens, as the letters of the Korean languages are rarely split into sub-character level in ASR preprocess.

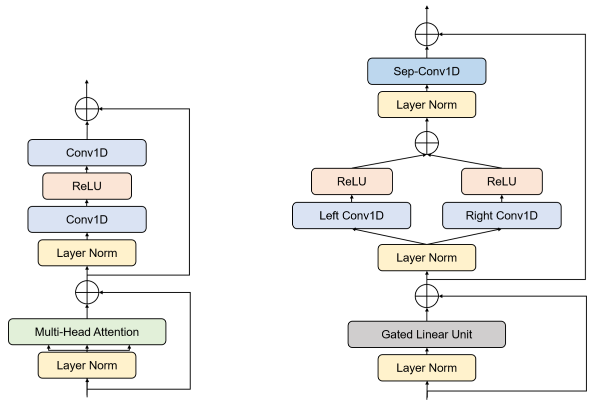
3.2. Implementation

Our algorithm was evaluated against Speech-Transformer proposed in [11] using the implementation provided by [19]. Because the branching structure within each block results in more trainable parameters than the baseline Transformer, the number of encoder and decoder blocks was set to half of that used for the original Transformer. All models were trained using the Adam optimizer [20] with hyperparameters $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$ and learning rate schedule $kd_{att}^{-0.5} \min\left(n^{-0.5}, nw^{-1.5}\right)$, where n is the number of Adam steps taken, $d_{att} = 256$ is the number of parameters in the model's attention module, $w = 25,000$ is a warmup parameter, and k is a tunable parameter initially set to 10 later decayed to 1 when the validation performance saturated. Predictions were decoded using beam search with a beam size of 10 and length penalty $\alpha = 1.0$. The model which attained the best validation accuracy was then selected for testing.

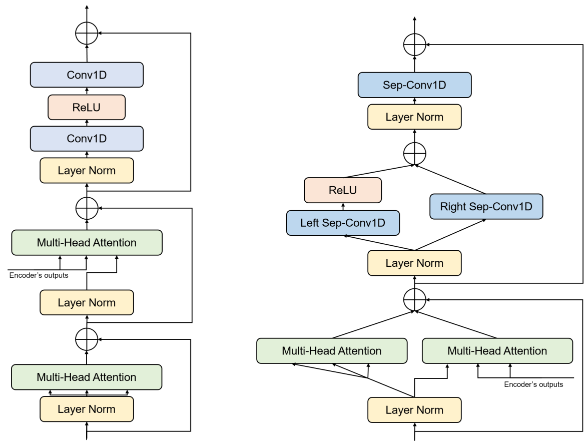
4. Results

4.1. Evolved Speech-Transformer

Among the 150 candidate models considered by our NAS algorithm, the model which reached the highest validation accuracy is hereon referred to as Evolved Speech-Transformer (EST). The resulting encoder and decoder architectures attained with WSJ are illustrated in Figures 3a and 3b, respectively.



(a) *Encoder: Multi-head attention in Transformer is replaced by Gated linear unit, and the outermost convolutional layer with a wide depth-wise separable convolutional layer.*



(b) *Decoder: EST's decoder is considerably different than Transformer's. In particular, the serial multi-head attention layer in Transformer is replaced with a branch structure, and the convolutional layer is replaced with wide depth-wise separable convolutions.*

Figure 3: ***(left)*** *Transformer and* ***(right)*** *Evolved Speech-Transformer (EST) architectures attained with the WSJ dataset.*

In the case of the original Speech-Transformer's encoder, six blocks consist of two structures on the left. However, for EST, the first three blocks form the right architecture, and the latter three form the left. As for the decoder, similarly, in the original, eight blocks are composed of two structures on the left, and in EST, the first four are formed on the right, and the latter four are formed on the left.

In contrast to the Transformer's encoder block, EST replaces the first three blocks with the 1D-convolution layer with a depthwise separable convolution layer with kernel size 9×1, and the MHA with a gated linear unit (GLU)[21]. The left and right branches both use a convolutional layer followed by ReLU activation, differing only in their kernel sizes 3×1 and 1×1 respectively, with the right branch having 4 times more filters than the left branch.

In the case of the decoder, EST replaces the first four blocks with branching structures. The first branch consists of two MHAs, one with twice the number of heads of the original Transformer decoder, the other receives the output of the encoder as a query and key, and the number of heads is the same as the original. The second branch has two separable convolu-

Table 1: *Performances of Transformer and Evolved Speech-Transformer (EST) on WSJ eval '92.*

Model	N_e	N_d	d_{ff}	# Params	WER (%)
Base Model	3	1	1024	8.8M	15.8
EST	3	1	1024	7.9M	14.9
Base Model	4	2	1024	12.5M	14.7
EST	4	2	1024	10.9M	14.2
Base Model	6	3	1024	17.7M	14.0
EST	6	3	1024	15.4M	13.4
Base Model	6	3	2048	27.2M	13.5
EST	6	3	2048	20.1M	**12.9**

Table 2: *Performances of Transformer and Evolved Speech-Transformer (EST) on the Zeroth test set. EST (WSJ) indicates the model attained with the WSJ dataset and retrained on Zeroth.*

Model	N_e	N_d	d_{ff}	# Params	WER (%)
Base Model	3	1	1024	8.8M	5.4
EST	3	1	1024	7.6M	4.9
EST (WSJ)	3	1	1024	7.9M	4.8
Base Model	4	2	1024	12.5M	4.8
EST	4	2	1024	10.4M	4.4
EST (WSJ)	4	2	1024	10.9M	4.4
Base Model	6	3	1024	17.7M	4.3
EST	6	3	1024	14.6M	3.9
EST (WSJ)	6	3	1024	15.4M	4.1
Base Model	6	3	2048	27.2M	3.9
EST	6	3	2048	19.1M	**3.7**
EST (WSJ)	6	3	2048	20.1M	3.8

tion layers, and each kernel size is 11×1 and 7×1, with output dimension is double and half the input dimension respectively. These processed states are finally passed through a 7×1 separable convolution layer with number of filters equal to the input dimension.

4.2. Performance

For a fair comparison between the baseline and EST, we report their word error rates (WERs) with the number of encoder and decoder blocks N_e and N_d as variables, using identical output dimensions $d_{ff} \in \{1024, 2048\}$. As shown in Tables 1, EST consistently outperforms the base model by an absolute margin of 0.6% on average while maintaining fewer parameters. In terms of memory occupation, EST has $\sim$ 13% fewer parameters than Transformer when the output dimension is set to 1024, and $\sim$ 26% fewer parameters when $d_{ff} = 2048$. This observation demonstrates how EST is particularly efficient when the feature dimension is large.

In the case of the EST found by the Zeroth dataset, the parameters were slightly different from those of the WSJ. The output size of the left convolution layer of the encoder was half of that of WSJ, and the number of MHA heads of the decoder was designed with half of the WSJ.

As shown in Tables 2, EST consistently outperforms the baseline by an absolute margin of 0.4% on average while maintaining fewer parameters. In terms of memory, or number of parameters, EST has $\sim$ 18% fewer parameters than Transformer when the output dimension is set to 1024, and $\sim$ 30% fewer

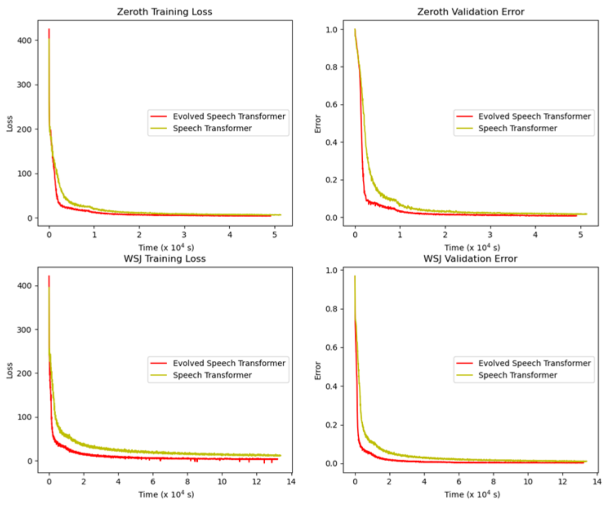


Figure 4: *Training time comparisons for Transformer and EST.*

parameters when $d_{ff} = 2048$.

We trained a model on Zeroth with the architecture attained via WSJ, and validated its higher performance compared to the baseline. This implies that an EST architecture earned from a specific speech corpus has potential of transferability to another dataset, in this case even with the different language.

Another benefit of EST over Transformer is its significantly reduced training time; Figure 4 shows Transformer's and EST's training loss and validation accuracies for first 100 epochs. In particular, training EST took 49K seconds on the Zeroth dataset in comparison to Transformer which took 51K seconds, saving about 4% of training time. For WSJ, Transformer stopped improving after 134K seconds whereas EST reached convergence in 132K seconds.

5. Conclusion

In this work, we constructed a new architecture of Speech-Transformer named Evolved Speech-Transformer by adapting search space and progressive dynamic hurdles, which work well for neural architecture search in automatically finding deep neural networks suitable for automatic speech recognition. The model we found outperforms Speech-Transformer across two datasets and various hyperparameters while maintaining fewer parameters. On WSJ and Zeroth dataset, our architecture converged with smaller training costs than the original model and achieved lower WER, which shows the efficiency and effectiveness of the neural architecture search. In addition, the potential of cross-dataset and multilingual transferability was verified by the result that the architecture extracted from a specific dataset shows high performance even when retrained to another dataset. We noticed how the EST optimized for speech recognition exhibited architectural similarities to the ET attained with language tasks. In fact, the EST's architecture was similar to the ET found in [12] which also has wide depth-wise separable convolutions and GLU. The only major architectural difference between EST and ET is that EST does not use a Swish activation function. We need to investigate further whether this result is due to the commonality such as sequential between the dataset used for NMT and for ASR. For future work, we will try to figure out whether this architecture comes out similarly for various seq2seq learning tasks such as speech synthesis [22].

6. References

[1] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[2] K. Audhkhasi *et al.*, "Building competitive direct acoustics-to-word models for english conversational speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4759–4763.

[3] A. Graves *et al.*, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369–376.

[4] K. Suyoun, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4835–4839.

[5] T. Hori *et al.*, "Advances in joint ctc-attention based end-to-end speech recognition with a deep cnn encoder and rnn-lm," *arXiv preprint arXiv:1706.02737*, 2017.

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

[7] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, 2014, pp. 1764–1772.

[8] W. Chan *et al.*, "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.

[9] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[10] V. Panayotov *et al.*, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[11] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5884–5888.

[12] D. R. So, C. Liang, and Q. V. Le, "The evolved transformer," *arXiv preprint arXiv:1901.11117*, 2019.

[13] B. Zoph *et al.*, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[14] E. Real *et al.*, "Regularized evolution for image classifier architecture search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 4780–4789.

[15] "Zeroth korean," *http://www.openslr.org/40/.*

[16] D. Povey *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[17] S. Virpioja *et al.*, "Morfessor 2.0: Python implementation and extensions for morfessor baseline," 2013.

[18] T. Kudo and R. John, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

[19] S. Watanabe *et al.*, "Espnet: End-to-end speech processing toolkit," *arXiv preprint arXiv:1804.00015*, 2018.

[20] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] Y. N. Dauphin *et al.*, "Language modeling with gated convolutional networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 933–941.

[22] N. Li *et al.*, "Neural speech synthesis with transformer network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6706–6713.