AUTOKWS: KEYWORD SPOTTING WITH DIFFERENTIABLE ARCHITECTURE SEARCH

Bo Zhang, WenFeng Li, Qingyuan Li, Weiji Zhuang, Xiangxiang Chu, Yujun Wang

Xiaomi AI Lab

ABSTRACT

Smart audio devices are gated by always-on lightweight keyword spotting programs to reduce power consumption. It is however challenging to design models that have both high accuracy and low latency for accurate and fast responsiveness. Many efforts have been made to develop end-to-end neural networks, in which depthwise separable convolutions, temporal convolutions, and LSTMs are adopted as building units. Nonetheless, these networks designed with human expertise may not achieve an optimal trade-off in an expansive search space. In this paper, we propose to leverage recent advances in differentiable neural architecture search to discover more efficient networks. Our found model attains 97.2% top-1 accuracy on Google Speech Command Dataset v1.

Index Terms— Keyword spotting, neural architecture search

1. INTRODUCTION

With the fast evolution of deep learning algorithms, keyword spotting (KWS) has become the entrance of smart device terminals, such as mobile phone, smart speakers, etc. In applications, one keyword is predefined for the keyword detection system, such as OK Google, Hey Siri, Xiao Ai Tong Xue, etc. After the system detects the keyword, the audio stream is uploaded to speech recognition system. The goal of KWS is to quickly and accurately detect keywords in real-time audio stream. Therefore, it is necessary to comprehensively evaluate three performances of accuracy, the number of parameters and operations. In previous work, the algorithms for KWS are generally divided into four types: 1) Feature template matching [1, 2] that uses the DTW algorithm to match the template by calculating distance. This method can save the training time but has poor robustness. 2) Decoding method based on graph search [3] that search the best path on the decoding graph using the viterbi algorithm still has strong competitiveness so far, but the cost of computation is expensive. 3) Post-processing method [4] computes the confidence score using sliding window on the posterior probability. When the confidence score surpass the threshold, the keyword is detected. This method has limit operations and is suitable for the limit resource platform. But it is worth noting that a neural network still needs to be pre-trained for frame-level alignment. 4) Recently, many works are focused on end-to-end method including sequence-to-sequence models and end-to-end systems [5, 6, 7, 8]. The paper [5] trained a LSTM model with connectionist temporal classification (CTC) for KWS that generate lattice for search. The paper [9] trained a CNN for end-to-end system to predict whether a keyword is spotted in audio stream without decoding the audio into phoneme or word strings.

Fig. 1. Differentiable architecture search in the adapted TC-ResNet search space. A searchable layer consists of several TC-ResNet-SE blocks and a skip connection, each is associated with an architectural parameter $\alpha$ to denote its importance. The outputs are summed up after multiplying $\sigma(\alpha)$. Note $\sigma$ can either be softmax for DARTS [10] and NoisyDARTS [11], or sigmoid for FairDARTS [12].

End-to-end KWS based on CNN can notice the information of the entire window, which needs less assumptions and has large room for improvement. Therefore, this paper is based on the end-to-end method to explore the model architecture with higher performance.
2. PRIOR WORK

2.1. Manual Designed Networks for KWS

The paper [13] explores various neural network architectures for KWS on embedded hardware in terms of accuracy, the number of parameters and operations, including the traditional DNN, CNN, LSTM, CRNN [14] and MobileNetV1 [15]. The experimental results show that the traditional DNN has limit parameters and operations, but the accuracy is slightly worse; CNN has high accuracy but has large number of operations; LSTM and CRNN can balance the number of operations and parameters and achieve good accuracy; MobileNetV1 achieves the best result, because its deep separation structure can make the network layer deeper and perform better. [16] explores the application of deep residual learning and dilated convolutions to KWS. And the model Res15 achieves 95.8% accuracy and surpasses CNN [9]. [17] proposed a temporal convolutional neural network for real-time KWS on embedded hardware using 1D convolution along temporal dimension. TC-ResNet8 proposed in this paper achieves 385x speedup and the accuracy is improved 0.3% accuracy compared to the deep and complex Res15 [16]. To reduce the number of parameters, [18] proposed a new architecture grouping depthwise separable convolutions (GDSConv) that achieves 96.4% accuracy with 62k parameters.

2.2. Neural Architecture Search and Audio

Neural architecture search (NAS) [19] has already become a new paradigm of designing neural networks for many deep learning tasks. DARTS [10] tremendously reduces the search cost with weight-sharing mechanism, which adopts gradient descent for a bi-level optimization on network parameters and architectural coefficients. It is however known to be unstable to reproduce. Our recent works Fair DARTS [12] and Noisy DARTS [11] study its failure case in depth and propose multiple ways to robustify DARTS. Specifically, Fair DARTS [12] breaks the exclusive competition among parallelizing operations and imposes an auxiliary loss to push architectural coefficients towards its extremities. Noisy DARTS [11] attenuates the unfair advantage of skip connections by adding small amount of Gaussian noise to their feature maps during the optimization. Both methods have been proved effective on classification tasks.

For audio tasks, NAS has also attracted considerable attention. Apart from our previous work NASC [20] adopting our two-stage one-shot NAS approach FairNAS [21] on acoustic scene classification, DARTS [12] has been also applied to speaker recognition in AutoSpeech [22], and to speech recognition in DARTS-ASR [23]. There is also a noticeable contemporary work [24] also applying DARTS on KWS. However, due to the complex cell-based network topology, their searched networks might be limited for direct application on smart devices. Therefore, [25] adopts DARTS in a domain-specific search space for ASR. Noticeably, there are some other NAS approaches on keyword spotting, such as NASIL [26] and [27], however they are relatively costly (60 and 50 GPU hours respectively) and both generate less competitive results.

3. METHOD

In this section, we undertake an efficient differentiable neural architecture search approach for the keyword spotting task. We first design a viable search space and then we perform DARTS and our robustified variants for searching.

3.1. Search Space Design and Analysis

We design our search space on top of TC-ResNet [17] due to its outstanding performance and small memory footprint. We also introduce the squeeze-and-excitation (SE) module [28] for the TC block, see Fig. 2. To measure whether the search space is well set, we manually fix convolutional kernels as {3,5,7,9} to train each model (trained 7 times and evaluated on the test set). The results are shown in Table 1. We discover that under manual settings, the performance ranges from 95.97% to 96.98% for TC-ResNet14-SE on V1 dataset and from 96.33% to 97.22% on V2.

Specifically, for each TC block we have options of kernel sizes in {3,5,7,9}, whether to enable SE or not, and an additional skip connection. In total, we have $6^9 \approx 10M$ models to search for TC14 and $9^9$ models for TC20 (we repeat each non-downsampling blocks once to increase from 6 blocks to 9).

![Fig. 2. The searchable TC-ResNet-SE block. The kernel size $j \in \{3, 5, 7, 9\}$. (a) optional SE module, (b) normal block, (c) reduction block (stride s=2)](image-url)
3.2. Searching Algorithm

Considering the efficiency and effectiveness, we adopt DARTS \[10\], FairDARTS \[12\] and NoisyDARTS \[11\] for searching. DARTS and FairDARTS undergo a bi-level optimization process where the network weights $w$ and architectural weights $\alpha$ are updated in an interleaved order, w.r.t. to the loss on train set and validation set respectively as follows,

\[
\min_{\alpha} \mathcal{L}_{val}(w^*(\alpha), \alpha) \tag{1}
\]
\[
\text{s.t. } w^*(\alpha) = \arg \min_w \mathcal{L}_{train}(w, \alpha) \tag{2}
\]

We have shown that how to construct different operations in the same layer in Figure 2. DARTS and FairDARTS differ in the activation function $\sigma$ used for architectural parameters $\alpha$. The former uses softmax (Equation 3), and the latter goes with sigmoid (Equation 4), which essentially makes each operation independent of others. Notice FairDARTS selects operations that are above a threshold ($\sigma = 0.8$ in our case) while DARTS chooses the one with the highest $\sigma$.

\[
\text{softmax}(\alpha_j) = \frac{e^{\alpha_j}}{\sum_{i=0}^{k} e^{\alpha_i}} \tag{3}
\]
\[
\text{sigmoid}(\alpha_j) = \frac{1}{1 + e^{-\alpha_j}} \tag{4}
\]

NoisyDARTS \[11\] observes that skip connection causes performance collapse by forming up a residual structure. It thus injects Gaussian noise to perturb the fluent gradient flow. In short, it blends the skip connection’s output with a noise as follows, where the mean $\mu$ is normally set to zero (to be unbiased) and the standard deviation $\beta$ to a small value.

\[
f_{\text{skip}}(x) = x + \mathcal{N}(\mu, \beta) \tag{5}
\]

4. EXPERIMENTS

4.1. Google Speech Commands Dataset

This paper explores the end-to-end neural network architecture on the Google speech commands datasets V1 and V2. With the standard splitting setup where the dataset was divided into training set, cross validation set and test set with a ratio of 8:1:1, V1 contains 22246, 3093, 3081 and V2 has 36923, 4445, 4890. Each audio is 1 second long and contains only one keyword. There are a total of 12 classes, including 10 keywords (Yes, NO, Up, Down, Left, Right, On, Off, Stop, Go) and two extra classes: silence and unknown (sampled from the remaining 20 keywords). For the reliability of the experimental results, the audios of one person can only be divided into the same data set.

4.2. Searching

For searching settings, we follow DARTS \[10\] with minor modifications. We train the supernet with a batch size of 128 for 50 epochs. We set a learning rate of 0.1 for the optimizer of network weight and 3e-2 for architectural optimizer. We use additive Gaussian noise with $\beta=0.1$ for NoisyDARTS experiments on V1 and $\beta=0.3$ for V2. The searching takes about 2 hours on a single V100 GPU.

4.3. Single Network Training

The input of the neural network is a 40-dimensional MFCC with better effect and closer to the characteristics of human ear. We follow the exact same setting as TC-ResNet \[17\] for training all the models. It takes nearly a hour on a single V100 GPU.

4.4. Searching Results

After searching with the adopted methods, we fully train each model from scratch. The results are shown in Table 2 and Table 3. The searching and evaluation are proxyless either on V1 or V2 dataset. We refrain from using techniques like SpecAugment \[29\] and Multi-head attention \[30\] to have fair comparison with other prior arts. Notice that with these tricks, the performance can be further boosted.

| Method               | Params | $x+$ | Avg. acc (%) | Best |
|----------------------|--------|------|--------------|------|
| TC-ResNet-14 \[17\]  | 305K   | 13.4M| 96.49±0.18*  | 96.7* |
| CENet-GCN-40 \[31\]  | 72.3K  | 16.18M| 96.8          | 97.0 |
| MHAir-RNN \[40\]     | 743K   | -    | 96.63±0.22    | 97.2†|
| Random-TK14          | 196K   | 8.8M | 96.58±0.15    | 96.8 |
| DARTS-TK14           | 93K    | 4.9M | 96.70±0.11    | 96.9 |
| FairDARTS-TK14       | 188K   | 10.6M| 96.79±0.30    | 97.2 |
| NoisyDARTS-TK14      | 109K   | 6.3M |              |      |

Table 2. Comparison with state-of-the-art lightweight models on Google Speech Command Dataset v1. *: rerun from their source code. †: w/ SpecAugment
| Method                  | Params | ×+ | Avg. acc (%) | Best   |
|------------------------|--------|----|--------------|--------|
| TC-ResNet-14 [17]      | 305K   |    | 96.79±0.18*  | 97.03* |
| NoisyDARTS-TC14        | 107K   |    | 96.95±0.14   | 97.32  |

Table 3. Comparison with state-of-the-art lightweight models on Google Speech Command Dataset v2. *: rerun from their source code.

5. CONCLUSION

This paper exploits differentiable neural architecture search on the keyword spotting task. Due to its strict hardware constraints and high performance requirement, we designed an efficient and applicable search space from TC-ResNet with minor modifications. We investigated DARTS and its two variants FairDARTS and NoisyDARTS for searching. The found architectures achieve state-of-the-art results on the standard Google Speech Command Dataset. We hope this effort could give a new paradigm for the future network design on KWS.

6. REFERENCES

[1] Guoguo Chen, Carolina Parada, and Tara N Sainath, “Query-by-example keyword spotting using long short-term memory networks,” in 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2015, pp. 5236–5240.

[2] Timothy J Hazen, Wade Shen, and Christopher White, “Query-by-example spoken term detection using phonetic posteriorgram templates,” in 2009 IEEE Workshop on Automatic Speech Recognition & Understanding. IEEE, 2009, pp. 421–426.

[3] Minhua Wu, Sankaran Panchapagesan, Ming Sun, Jiacheng Gu, Ryan Thomas, Shiv Naga Prasad Vataldevuni, Bjorn Hoffmeister, and Arindam Mandal, “Monophone-based background modeling for two-stage on-device wake word detection,” in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018, pp. 5494–5498.

[4] Guoguo Chen, Carolina Parada, and Georg Heigold, “Small-footprint keyword spotting using deep neural networks,” in 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2014, pp. 4087–4091.

[5] Yimeng Zhuang, Xuanakl Chang, Yanmin Qian, and Kai Yu, “Unrestricted vocabulary keyword spotting using lstm-etc,” in Interspeech, 2016, pp. 938–942.

[6] Kartik Audhkkhasi, Andrew Rosenberg, Abhinav Sthty, Bhuvana Ramabhadran, and Brian Kingsbury, “End-asr-free keyword search from speech,” IEEE Journal of Selected Topics in Signal Processing, vol. 11, no. 8, pp. 1351–1359, 2017.

[7] Raziel Alvarez and Hyun-Jin Park, “End-to-end streaming keyword spotting,” in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 6336–6340.

[8] Yanzhang He, Rohit Prabhavalkar, Kanishka Rao, Wei Li, Anton Bakhtin, and Ian McGraw, “Streaming small-footprint keyword spotting using sequence-to-sequence models,” in 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). IEEE, 2017, pp. 474–481.

[9] Tara N Sainath and Carolina Parada, “Convolutional neural networks for small-footprint keyword spotting,” in Sixteenth Annual Conference of the International Speech Communication Association, 2015.

[10] Hanxiao Liu, Karen Simonyan, and Yiming Yang, “DARTS: Differentiable architecture search,” arXiv preprint arXiv:1806.09055, 2018.

[11] Xiangxiang Chu, Bo Zhang, and Xudong Li, “Noisy differentiable architecture search,” arXiv preprint arXiv:2005.03566, 2020.

[12] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li, “Fair DARTS: Eliminating unfair advantages in differentiable architecture search,” in 16th European Conference on Computer Vision, 2020.

[13] Yundong Zhang, Naveen Suda, Liangzhenn Lai, and Vikas Chandra, “Hello edge: Keyword spotting on microcontrollers,” arXiv preprint arXiv:1711.07128, 2017.

[14] Sercan O Arik, Markus Kliegl, Rewon Child, Joel Hestness, Andrew Gibiansky, Chris Fougner, Ryan Prenger, and Adam Coates, “Convolutional recurrent neural networks for small-footprint keyword spotting,” arXiv preprint arXiv:1703.05390, 2017.

[15] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” arXiv preprint arXiv:1704.04861, 2017.

[16] Raphael Tang and Jimmy Lin, “Deep residual learning for small-footprint keyword spotting,” in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2018, pp. 5484–5488.

[17] Seungwoo Choi, Seokjun Seo, Beomjun Shin, Hyeongmin Byun, Martin Kersner, Beomsu Kim, Dongyoung Kim, and Sungjoo Ha, “Temporal convolution for
real-time keyword spotting on mobile devices,” *arXiv preprint arXiv:1904.03814*, 2019.

[18] Simon Mittermaier, Ludwig Kürzinger, Bernd Waschneck, and Gerhard Rigoll, “Small-footprint keyword spotting on raw audio data with sinc-convolutions,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7454–7458.

[19] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le, “Learning transferable architectures for scalable image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[20] Jixiang Li, Chuming Liang, Bo Zhang, Zhao Wang, Fei Xiang, and Xiangxiang Chu, “Neural architecture search on acoustic scene classification,” *arXiv preprint arXiv:1912.12825*, 2019.

[21] Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li, “Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search,” *arXiv preprint arXiv:1907.01845*, 2019.

[22] Shaojin Ding, Tianlong Chen, Xinyu Gong, Weiwei Zha, and Zhangyang Wang, “Autospeech: Neural architecture search for speaker recognition,” *arXiv preprint arXiv:2005.03215*, 2020.

[23] Yi-Chen Chen, Jui-Yang Hsu, Cheng-Kuang Lee, and Hung-yi Lee, “Darts-asr: Differentiable architecture search for multilingual speech recognition and adaptation,” *arXiv preprint arXiv:2005.07029*, 2020.

[24] Tong Mo, Yakun Yu, Mohammad Salameh, Di Niu, and Shangling Jui, “Neural architecture search for keyword spotting,” *arXiv preprint arXiv:2009.00165*, 2020.

[25] Shoukang Hu, Xurong Xie, Shansong Liu, Mengzhe Geng, Xunying Liu, and Helen Meng, “Neural architecture search for speech recognition,” *arXiv preprint arXiv:2007.08818*, 2020.

[26] Farzaneh S Fard, Arash Rad, and Vikrant Singh Tomar, “Nasil: Neural architecture search with imitation learning,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3972–3976.

[27] Hanna Mazzawi, Xavi Gonzalvo, Aleks Kracun, Prashant Sridhar, Niranjan Subrahmanya, Ignacio Lopez-Moreno, Hyun-Jin Park, and Patrick Violette, “Improving keyword spotting and language identification via neural architecture search at scale,” in *INTERSPEECH*, 2019, pp. 1278–1282.