Accelerating Neural Architecture Search for Natural Language Processing with Knowledge Distillation and Earth Mover's Distance

Jianquan Li*
Faculty of Artificial Intelligence and Big Data, Yibin University
lijianquan2@ultrapower.com.cn

Xiaokang Liu*
Beijing Ultrapower Software Co.,Ltd.
liuxiaokang1@ultrapower.com.cn

Sheng Zhang*
National University of Defense Technology
zhangsheng@nudt.edu.cn

Min Yang†
SIAT, Chinese Academy of Sciences
min.yang@siat.ac.cn

Ruifeng Xu
Harbin Institute of Technology (Shenzhen)
xuruifeng@hit.edu.cn

Fengqing Qin
Faculty of Artificial Intelligence and Big Data, Yibin University
qinfengqing@163.com

ABSTRACT

Recent AI research has witnessed increasing interests in automatically designing the architecture of deep neural networks, which is coined as neural architecture search (NAS). The automatically searched network architectures via NAS methods have outperformed manually designed architectures on some NLP tasks. However, training a large number of model configurations for efficient NAS is computationally expensive, creating a substantial barrier for applying NAS methods in real-life applications. In this paper, we propose to accelerate neural architecture search for natural language processing based on knowledge distillation (called KD-NAS). Specifically, instead of searching the optimal network architecture on the validation set conditioned on the optimal network weights on the training set, we learn the optimal network by minimizing the knowledge loss transferred from a pre-trained teacher network to the searching network based on Earth Mover's Distance (EMD). Experiments on five datasets show that our method achieves promising performance compared to strong competitors on both accuracy and searching speed. For reproducibility, we submit the code at: https://github.com/lxk00/KD-NAS-EMD.

CCS CONCEPTS

• **Computing methodologies** → **Neural architecture search**.

KEYWORDS

Neural architecture search, knowledge distillation, earth mover's distance

*Both authors contributed equally to this research.
†Min Yang is corresponding author.

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3463017

ACM Reference Format:
Jianquan Li, Xiaokang Liu, Sheng Zhang, Min Yang, Ruifeng Xu, and Fengqing Qin. 2021. Accelerating Neural Architecture Search for Natural Language Processing with Knowledge Distillation and Earth Mover's Distance. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3404835.3463017

1 INTRODUCTION

Deep neural networks (DNNs) have enabled remarkable progress over the last years in a variety of natural language processing (NLP) tasks. One crucial aspect for this progress is to manually design a DNN architecture that effectively captures the syntax and semantics of texts. However, the design of the network architecture often relies heavily on the researchers' prior knowledge and tedious trials. In recent years, NAS has emerged, which aims to design a network architecture with the best performance automatically with as little human intervention as possible [28]. So far, the automatically searched network architectures via NAS have outperformed manually designed architectures on some NLP tasks [16, 23].

Despite its remarkable empirical performance, most NAS methods are computationally expensive. In particular, obtaining a state-of-the-art architecture with reinforcement learning (RL) [2, 17, 28] and evolutionary algorithms [14, 20] often consumes hundreds of GPU days or even more computing resources. For example, a popular reinforcement learning based NAS method [28] on CIFAR-10 requires 2000 GPU days, while the evolution method [18] on CIFAR-10 requires 3150 GPU days. This huge amount of calculation creates a substantial barrier to the wide application of NAS in NLP. Therefore, it is critical to accelerate the search of the network architecture while maintaining the promising accuracy.

Recently, there are many works being proposed to reduce the amount of calculation and accelerate the architecture search [15, 17], of which the gradient-based NAS methods [4, 13, 15, 24] are considered to be able to provide a practical way. The main idea is to formulate gradient-based methods as a bilevel optimization problem [5], where the optimal neural architecture on the validation set depends on the optimal network weights on the training set. For example, DARTS [15] is considered a typical work of the gradient-based NSA, which continuously relaxes the originally discrete search space and

makes it possible to use gradients to efficiently optimize the search space of the architecture based on bilevel optimization.

Although previous gradient-based NAS methods reduce the inference time and achieve excellent performances on a variety of tasks [24, 25], they still suffer from certain inherent challenges. First, most gradient-based methods, such as DARTS [15], adopt second-order approximation to solve bilevel optimization. However, the evaluation based on the undertrained network parameters cannot correctly rank the candidate models. In particular, the architecture that achieves the best performance at early training stage cannot defend its top ranking when convergence is obtained. Second, previous gradient-based NAS methods search the optimal network architecture on the validation set conditioned on the optimal network weights on the training set. Specifically, in the search stage, a NAS algorithm is employed to find the architecture with the highest validation accuracy. However, as revealed in [3, 11], the validation accuracy of the model is not predictive to its true performance.

In this paper, we exploit knowledge distillation to accelerate neural architecture search for natural language processing based on Earth Mover's Distance (denoted as KD-NAS). Specifically, instead of searching the optimal architecture on the validation set conditioned on the optimal network weights on the training set, we search the optimal network architecture by minimizing the knowledge loss transferred from a pre-trained teacher network to the searching network based on Earth Mover's Distance (EMD). Each searching layer can adaptively learn from different hidden layers of the teacher model for different NLP tasks. In addition, we introduce an AddNorm operation to relieve the collapse problem in gradient-based NAS methods, which normalizes the NAS framework after concatenating the hidden nodes at each layer.

We summarize our contributions as follows. (1) To our knowledge, we are the first to exploit knowledge distillation to accelerate NAS based on EMD. We use the hidden representations of the pre-trained teacher model to supervise our architecture search via many-to-many layer mapping, where each intermediate searching layer has the chance to learn from all the intermediate layers of the teacher model. (2) Experiments show that KD-NAS consistently outperforms strong NAS methods across multiple NLP tasks. In particular, on the RTE [22] language inference task, KD-NAS achieves 54.93% accuracy with 0.09 GPU hours, which is 35.9 times faster than DARTS [15] with 0.5% higher accuracy and 8.1 times faster than SNAS [24] with 0.6% higher accuracy.

2 METHODOLOGY

2.1 Overview

The goal of this work is to automatically search an optimal student architecture based on a large pre-trained teacher model. As illustrated in Figure 1, we leverage knowledge distillation to accelerate NAS for NLP tasks, which learns the optimal student network containing a stack of searched cells by minimizing the knowledge loss transferred from the pre-trained teacher network to the searching network based on Earth Mover's Distance (EMD).

2.2 Teacher Model

We adopt BERT as the teacher model due to its superior performance on various NLP tasks. The architecture of BERT is a multi-layer bidirectional Transformer encoder, which is composed of a stack of N identical blocks. Each block consists of two sub-layers, where the first layer is a multi-head self-attention mechanism and the second layer is a simple fully connected feed-forward network. The residual connection is applied to each of the two continuous sub-layers, followed by layer normalization [1].

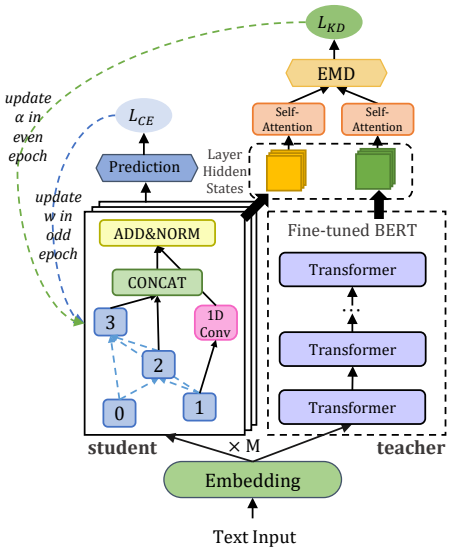

Figure 1: The overall framework of our method.

2.3 Student Model

We learn the student network using NAS techniques rather than assigning a fixed structure in advance by humans. In this paper, we propose a micro search strategy to find an optimal network architecture from the search space formed by the operation sets.

2.3.1 Search Space. We modularize the large search space of the deep model into cells to reduce its complexity. In the training phase, we only need to search for a few cell structures and then repeatedly stack such cells to form the final student architecture, following the DARTS model [15]. Specifically, each searched cell is represented as a directed acyclic graph consisting of an ordered sequence of M nodes, where each node x_i denotes a latent state $\mathbf{h}_i$. We represent each edge from node i to node j as $o_{i,j}$, indicating the operation to transform node i into node j. We compute each intermediate node x_j based on all of its predecessors in the directed acyclic graph:

$$\bar{o}_{i,j}(x) = \sum_{o \in O} \frac{\exp(\alpha^o_{i,j})}{\sum_{o' \in O} \exp(\alpha^{o'}_{i,j})} \cdot o(x) \tag{1}$$

$$x_j = \sum_{i<j} o_{i,j}(x_i) \tag{2}$$

where $o(\cdot) \in O$ denotes a operation from a set of candidate operations O (e.g., convolution, max pooling, skip, zero). $\alpha^o_{i,j}$ is a $|O|$-dimensional vector representing network weights of the architecture, which is trainable during the training phase. Here, we use the softmax operation over all possible operations to make the search space continuous. In this way, the task of neural architecture search reduces to learn each continuous variable $\alpha^o_{i,j}$.

Inspired by the success of residual connection and layer normalization in many deep neural networks, we also employ a residual connection to each of the two continuous cells, followed by layer normalization [1]. The output hidden states H_k of the k-th cell can be computed as:

$$H_k = \text{LayerNorm}([x_2^k; \ldots; x_M^k] + \sigma(H_{k-1})), \tag{3}$$

where σ is a convolutional neural network with unit kernel, which makes the size of input and output be the same. x_i^k defines the i-th node of the k-th cell. M denotes the total number of nodes in each cell. The introduction of this LayerNorm operation can greatly reduce the collapse of the model.

After the architecture search process, we can obtain a discrete architecture by replacing each weight parameter $\alpha_{i,j}$ with the most likely operation, i.e., $o_{i,j} = \text{argmax}_{o \in O} \alpha_{i,j}^o$. We assume there is a topological order among M intermediate nodes, the search space $\mathcal{A}$ is thus formalized as:

$$\mathcal{A} = [o_{0,1}, o_{0,2}, o_{1,2}, \ldots, o_{i,j}, \ldots, o_{M-1,M}] \tag{4}$$

2.3.2 Operation Set. In this work, we adopt the lightweight CNN-based operations as candidates given that they have shown both competitive accuracy and superior efficiency in various NLP tasks. The candidate operation set O is composed of four kinds of operations: *convolution*, *pooling*, *skip* and *zero* operation. The *convolution* operations include the 1D dilated convolution and standard convolution (without dilation) with kernel size $\{3, 5, 7\}$. The *pooling* operations include max pooling and average pooling with kernel size 3. The *skip* operation is leveraged to construct residual connections. The *zero* operation helps to forget the past knowledge.

2.4 Knowledge Distillation for NAS

Instead of searching the optimal architecture on the validation set conditioned on the optimal network weight on the training set as in DARTS [15], we learn the optimal network architecture by minimizing the knowledge loss transferred from the pre-trained teacher network to the searching student network. Since there are different kinds of hidden layers in teacher and student models, conventional one-to-one layer mapping algorithms [8, 9, 27] cannot be applied directly. Inspired by [10], we leverage Earth Mover's Distance (EMD) for knowledge distillation, which enables each intermediate layer of student to learn from any other intermediate layers of the teacher model.

Formally, let $\mathbf{H}_i^T$ and $\mathbf{H}_j^S$ be hidden states of the i-th teacher layer and the j-th student layer, respectively. Different from the standard knowledge distillation where the teacher and student networks share similar layer structures, our search cells are composed of convolution, pooling and skip, zero operations, making it difficult to directly calculate the difference between hidden layers of teacher and student models. In this paper, we employ self-attention mechanism to learn the self-attention matrices $\mathbf{A}_i^T$ and $\mathbf{A}_j^S$ capturing the global document features of teacher hidden layer $\mathbf{H}_i^T$ and student hidden layer $\mathbf{H}_j^S$ in a shared space:

$$\mathbf{A}_i^T = softmax(\frac{\mathbf{H}_i^T(\mathbf{H}_i^T)^\top}{\sqrt{l^T}}), \quad \mathbf{A}_j^S = softmax(\frac{\mathbf{H}_j^S(\mathbf{H}_j^S)^\top}{\sqrt{l^S}}), \tag{5}$$

Table 1: The statistics of datasets.

Dataset	Task	Metrics	#Train	#Dev
SST-2	sentiment classification	ACC	67350	873
MRPC	textual similarity	F1	4077	409
WikiQA	question answering	MAP/MRR	20359	2733
RTE	language inference	ACC	2490	277
TREC	semantic matching	ACC	5452	500

Table 2: Experimental results on five datasets.

	MRPC	SST	WikiQA	RTE	TREC	AVE
VDCNN	81.51	83.77	66.46/66.88	54.03	90.46	75.29
Transformer	81.23	82.68	67.1/67.82	53.61	90.60	75.12
DARTS	81.77	85.53	68.20/68.91	54.63	92.70	76.64
SNAS	81.54	85.43	68.18/68.86	54.57	92.83	76.58
PC-DARTS	81.78	85.81	68.43/68.96	54.87	93.23	76.88
ENAS	81.46	83.60	67.49/68.03	54.27	90.80	75.58
Ours	**82.00**	**86.10**	**69.55/70.50**	**54.93**	**94.33**	**77.48**

where l^T and l^S represent the dimensions of hidden representations $\mathbf{H}_i^T$ and $\mathbf{H}_j^S$, respectively.

We define a "ground" distance matrix $\mathbf{D} = \left[d_{ij}\right]$, where d_{ij} indicates the cost of transferring knowledge from $\mathbf{H}_i^T$ to $\mathbf{H}_j^S$. We employ the mean square error (MSE) to calculate the distance d_{ij} as follows:

$$d_{ij} = \text{MSE}(\mathbf{A}_j^S, \mathbf{A}_i^T) \tag{6}$$

Then, we learn a mapping flow matrix $\mathbf{F} = [f_{ij}]$ to minimize the cumulative cost for transferring knowledge from $\mathbf{H}^T$ to $\mathbf{H}^S$, where f_{ij} denotes the mapping flow between $\mathbf{H}_i^T$ and $\mathbf{H}_j^S$. Once the optimal mapping flow $\mathbf{F}$ is learned, we can define the Earth Mover's Distance as the work normalized by the total flow:

$$\text{EMD}(\mathbf{H}^S, \mathbf{H}^T) = \sum_{i=1}^{P} \sum_{j=1}^{N} f_{ij} d_{ij} \tag{7}$$

where P and N denote the number of hidden layers in the student and teacher networks, respectively. The objective function $\mathcal{L}_{KD}$ for EMD based knowledge distillation can be defined as:

$$\mathcal{L}_{KD} = \text{EMD}(\mathbf{H}^S, \mathbf{H}^T) \tag{8}$$

2.5 Model Optimization

Different from previous NAS methods that search the optimal architecture on the validation set conditioned on the optimal network weight on the training set, we first pre-train the teacher model and then search for the student structure under the supervision of the pre-trained teacher model. Specifically, we combine the knowledge distillation loss $\mathcal{L}_{\text{KD}}$ w.r.t the pre-trained teacher model and the cross-entropy loss ($\mathcal{L}_{CE}$) w.r.t ground-truth labels from the training data to assist the searching process. The model optimization is defined as follows:

$$\begin{aligned} &\min_{\alpha} \quad \mathcal{L}_{\text{KD}}(w^*(\alpha), \alpha) \\ &\text{s.t.} \quad w^*(\alpha) = \arg\min_{w} \mathcal{L}_{\text{KD}}(w, \alpha) \end{aligned} \tag{9}$$

where α represents the searching architecture and w indicates the weights. During the optimization process, we first fix the architecture α and update the weights w according to the cross-entropy

Table 3: The comparison of running speed of different models. #ep denotes the number of epochs for searching the best architecture; ep/s denotes the average running time in seconds per epoch; total denotes the total running time in hour.

Model	MRPC			SST			WikiQA			RTE			TREC		
	#ep	ep/s	total/h	#ep	ep/s	total/h	#ep	ep/s	total/h	#ep	ep/s	total/h	#ep	ep/s	total/h
DARTS	59	292	4.79	70	5220	101.50	63	1581	27.67	57	204	3.23	51	432	6.12
SNAS	57	69	1.09	66	1270	23.28	36	372	3.72	56	47	0.73	59	101	1.66
PC-DARTS	44	100	1.22	69	1711	32.79	52	517	7.46	54	67	1.01	49	148	2.01
Ours	10	74	0.21	9	1306	3.27	7	392	0.76	6	51	0.09	12	109	0.36

loss on the training set. Then, we fix the weights w and update the architecture α according to the distillation loss. By alternatively updating the architecture α and weight parameters w, the optimal architecture can be obtained through the gradient descent algorithm.

3 EXPERIMENTAL SETUP

Datasets. To evaluate the effectiveness of our method, we conduct extensive experiments on five benchmark datasets, including SST-2 [19] for sentiment classification, MRPC [7] for textual similarity calculation, WikiQA [26] for question answering, RTE [22] for language inference, TREC [12] for semantic matching. The statistics of these five datasets are provided in Table 1.

Implementation Details. The number of layers in the student model is set to be 4. The number of inner nodes in each search cell is set to be 3. The learning rate for the operation parameters w is chosen from {2e-2, 1e-3, 1e-4}, while the learning rate for the structure parameters α is chosen from {1e-3, 5e-4, 1e-4}. The size of the hidden state in the student model is set to be 128. We optimize the model parameters with SGD optimizer. The batch size is set to be 32 in both searching and re-training stages. The max length of the input sequence is set to be 128. We adopt pre-trained BERT to initialize the word embeddings.

Baselines. We compare our method with several state-of-the-art NAS approaches, including DARTS [15], PC-DARTS [25], SNAS [24], ENAS [17]. We adopt the same candidate operations for all the baseline models. In addition, we also compare our model with two manually designed networks, including VDCNN [6] and 12-layer Transformers [21].

4 EXPERIMENTAL RESULTS

4.1 Main Results

To ensure the stability of our KD-NAS model, we run KD-NAS three times, and report the average results in Table 2. From the results, we can observe that our method outperforms the compared baselines on all the five datasets. In particular, on the TREC dataset, the accuracy of our method increase by 1.17% over the best baseline (e.g., PC-DARTS).

Since the aim of this work is to accelerate neural architecture search with knowledge distillation, we also report the running time of different models in Table 3. Specifically, we report the number of epochs for different approaches to obtain the best architecture and the average time for each epoch in searching step. Here, we do not provide the running time of ENAS since it pre-defines the number of steps within each epoch, while the other methods depend on the size of training data. From the results, we can observe that our method is at least 17x faster than the popular DARTS model, while maintaining slightly better performances. This is because our KD guided NAS method simplifies the searching target with the help of the teacher model.

Table 4: Ablation test results on MRPC, SST and RTE.

Model	MRPC		SST		RTE	
	ACC	#ep	ACC	#ep	ACC	#ep
Full Model	82.00	10	86.10	9	54.93	6
- w/o EMD	81.77	41	85.26	41	54.45	13
- w/o Add&Norm	81.93	15	85.23	14	54.57	10

4.2 Ablation Study

To better understand the impact of different components of our method on effectiveness and searching speed, we perform ablation study in terms of removing EMD (denoted as w/o EMD) and the residual connection followed by layer normalization (denoted as w/o Add&Norm). We summarize the experimental results in Table 4. We observe that each component contributes to both the accuracy and the speed of our method. In particular, the EMD-based knowledge distillation algorithm greatly reduces the number of epochs required for learning the best architecture. The improvement of the residual connection and layer normalization is also significant. It is no surprise that combining all the factors achieves the best performance on all the experimental datasets.

5 CONCLUSION

In this paper, we propose a novel NAS method for natural language processing with knowledge distillation, which automatically searches the optimal network architecture by minimizing the knowledge loss transferred from a pre-trained teacher network to the searching network based on Earth Mover's Distance(EMD). Experiments on five NLP tasks demonstrate that our method achieves considerably better performance than strong baselines while significantly accelerating searching speed.

ACKNOWLEDGEMENT

This parer was partially supported by Key projects of Sichuan Provincial Department of Education (No.18ZA0538), Key projects of Yibin science and Technology Bureau (2017ZSF009-9), Research projects of Yibin University (2020YY02), National Natural Science Foundation of China (No. 61906185), Youth Innovation Promotion Association of CAS China (No. 2020357), Shenzhen Basic Research Foundation (No. JCYJ20200109113441941), Shenzhen Science and Technology Innovation Program (Grant No. KQTD20190929172835662).

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[2] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. 2016. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167* (2016).

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

[4] Xiangxiang Chu, Tianbao Zhou, Bo Zhang, and Jixiang Li. 2020. Fair darts: Eliminating unfair advantages in differentiable architecture search. In *European Conference on Computer Vision*. Springer, 465–480.

[5] Benoît Colson, Patrice Marcotte, and Gilles Savard. 2007. An overview of bilevel optimization. *Annals of operations research* 153, 1 (2007), 235–256.

[6] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very Deep Convolutional Networks for Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, 1107–1116.

[7] William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *stat* 1050 (2015), 9.

[9] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351* (2019).

[10] Jianquan Li, Xiaokang Liu, Honghong Zhao, Ruifeng Xu, Min Yang, and Yaohong Jin. 2020. BERT-EMD: Many-to-Many Layer Mapping for BERT Compression with Earth Mover's Distance. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 3009–3018.

[11] Xiang Li, Chen Lin, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. 2020. Improving one-shot nas by suppressing the posterior fading. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13836–13845.

[12] Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

[13] Hanwen Liang, Shifeng Zhang, Jiacheng Sun, Xingqiu He, Weiran Huang, Kechen Zhuang, and Zhenguo Li. 2019. Darts+: Improved differentiable architecture search with early stopping. *arXiv preprint arXiv:1909.06035* (2019).

[14] Jason Liang, Elliot Meyerson, and Risto Miikkulainen. 2018. Evolutionary architecture search for deep multitask networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 466–473.

[15] Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. DARTS: Differentiable Architecture Search. In *International Conference on Learning Representations*.

[16] Ansel MacLaughlin, Jwala Dhamala, Anoop Kumar, Sriram Venkatapathy, Ragav Venkatesan, and Rahul Gupta. 2020. Evaluating the Effectiveness of Efficient Neural Architecture Search for Sentence-Pair Tasks. *arXiv preprint arXiv:2010.04249* (2020).

[17] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. 2018. Efficient Neural Architecture Search via Parameters Sharing. In *International Conference on Machine Learning*. 4095–4104.

[18] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. 2019. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 33. 4780–4789.

[19] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. 1631–1642.

[20] Gerard Jacques van Wyk and Anna Sergeevna Bosman. 2019. Evolutionary neural architecture search for image restoration. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[22] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461* (2018).

[23] Yujing Wang, Yaming Yang, Yiren Chen, Jing Bai, Ce Zhang, Guinan Su, Xiaoyu Kou, Yunhai Tong, Mao Yang, and Lidong Zhou. 2020. TextNAS: A Neural Architecture Search Space Tailored for Text Representation.. In *AAAI*. 9242–9249.

[24] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. 2018. SNAS: stochastic neural architecture search. In *International Conference on Learning Representations*.

[25] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. 2019. Pc-darts: Partial channel connections for memory-efficient architecture search. In *International Conference on Learning Representations*.

[26] Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 2013–2018.

[27] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4320–4328.

[28] Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016).