Les Cahiers du GERAD ISSN: 0711–2440

State of compact architecture search for deep neural networks

M. Shafiee, A. Hryniowski,
F. Li, Z. Q. Lin, A. Wong

G–2020–23–EIW11

April 2020

La collection *Les Cahiers du GERAD* est constituée des travaux de recherche menés par nos membres. La plupart de ces documents de travail a été soumis à des revues avec comité de révision. Lorsqu'un document est accepté et publié, le pdf original est retiré si c'est nécessaire et un lien vers l'article publié est ajouté.

Citation suggérée : M. Shafiee, A. Hryniowski, F. Li, Z. Q. Lin, A. Wong (Avril 2020). State of compact architecture search for deep neural networks, *In* C. Audet, S. Le Digabel, A. Lodi, D. Orban and V. Partovi Nia, (Eds.). Proceedings of the Edge Intelligence Workshop 2020, Montréal, Canada, 2–3 Mars, 2020, pages 71–76. Les Cahiers du GERAD G–2020–23, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2020-23-EIW11) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

The series *Les Cahiers du GERAD* consists of working papers carried out by our members. Most of these pre-prints have been submitted to peer-reviewed journals. When accepted and published, if necessary, the original pdf is removed and a link to the published article is added.

Suggested citation: M. Shafiee, A. Hryniowski, F. Li, Z. Q. Lin, A. Wong (April 2020). State of compact architecture search for deep neural networks, *In* C. Audet, S. Le Digabel, A. Lodi, D. Orban and V. Partovi Nia, (Eds.). Proceedings of the Edge Intelligence Workshop 2020, Montreal, Canada, March 2–3, 2020, pages 71–76. Les Cahiers du GERAD G–2020–23, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2020-23-EIW11) to update your reference data, if it has been published in a scientific journal.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

State of compact architecture search for deep neural networks

Mohammad Shafiee [a,b]

Andrew Hryniowski [a,b]

Francis Li, [b]

Zhong Qiu Lin [a,b]

Alexander Wong [a,b]

[a] *Waterloo Artificial Intelligence Institute, Waterloo (Ontario), Canada, N2L 3G1*

[b] *DarwinAI Corp., Waterloo (Ontario) Canada, N2V 1K4*

April 2020
Les Cahiers du GERAD
G–2020–23–EIW11

Copyright © 2020 GERAD, Shafiee, Hryniowski, Li, Lin, Wong

Les textes publiés dans la série des rapports de recherche *Les Cahiers du GERAD* n'engagent que la responsabilité de leurs auteurs. Les auteurs conservent leur droit d'auteur et leurs droits moraux sur leurs publications et les utilisateurs s'engagent à reconnaître et respecter les exigences légales associées à ces droits. Ainsi, les utilisateurs:

- Peuvent télécharger et imprimer une copie de toute publication du portail public aux fins d'étude ou de recherche privée;
- Ne peuvent pas distribuer le matériel ou l'utiliser pour une activité à but lucratif ou pour un gain commercial;
- Peuvent distribuer gratuitement l'URL identifiant la publication.

Si vous pensez que ce document enfreint le droit d'auteur, contactez-nous en fournissant des détails. Nous supprimerons immédiatement l'accès au travail et enquêterons sur votre demande.

The authors are exclusively responsible for the content of their research papers published in the series *Les Cahiers du GERAD*. Copyright and moral rights for the publications are retained by the authors and the users must commit themselves to recognize and abide the legal requirements associated with these rights. Thus, users:

- May download and print one copy of any publication from the public portal for the purpose of private study or research;
- May not further distribute the material or use it for any profit-making activity or commercial gain;
- May freely distribute the URL identifying the publication.

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Abstract: *The design of compact deep neural networks is a crucial task to enable widespread adoption of deep neural networks in the real-world, particularly for edge and mobile scenarios. Due to the time-consuming and challenging nature of manually designing compact deep neural networks, there has been significant recent interest into algorithms that automatically search for compact network architectures. A particularly interesting class of compact architecture search algorithms are those guided by baseline network architectures. In this study, we explore the current state of compact architecture search for deep neural networks through both theoretical and empirical analysis of four different state-of-the-art compact architecture search algorithms: i) group lasso regularization, ii) variational dropout, iii) MorphNet, and iv) Generative Synthesis. We examine these methods in detail based on a number of different factors such as efficiency, effectiveness, and scalability across three well-known benchmark datasets, as well as explore practical considerations.*

1 Introduction

Designing compact deep neural network architectures for edge scenarios is very time-consuming and human insight about optimized macro- and micro-architectures is limited in terms of design granularity. To address this need, a number of algorithmic approaches have been proposed in literature for designing compact neural network architecture. For example, a combination of pruning, quantization and coding techniques have been leveraged to address the storage requirement of a deep neural network. Low-rank matrix factorization is another technique which was applied to approximate the filter structures and convolutional kernels in a deep neural network. Structural learning approach during the training of a model is another trick to learn the filter shape and depth during the training process. Compact architectural search was also addressed via variational Bayesian algorithms. One are of great recent interest is neural architecture search (NAS) [1], leading to a variety of strategies such as evolutionary algorithms, reinforcement learning methods, or Bayesian optimization. For example, Shafiee et al. [6] formulated the compact network architectural search problem via an evolutionary synthesis framework so-called EvoNet. Such methods can be designed to target compact network architectures to great potential effect, which we will refer to as compact architecture search approaches.

In this study, we explore the current state of compact architecture search for deep neural networks by conducting both theoretical and empirical analysis on four different state-of-the-art compact architecture search algorithms across three different datasets based on accuracy, computational complexity, and architectural complexity. This provides us with a detailed perspective of where the field stands in this area of research.

2 Theoretical analysis

A key criteria when selecting the set of algorithms to study for a better understanding of the current state of compact architecture search is scalability. Here, we focus on exploring state-of-the-art compact architecture search algorithms that, guided by baseline network architectures, produce compact neural network architectures with smaller yet structured topologies that tend well to parallel computing on modern processor architectures as well as provide lower effective memory footprint during inference. Furthermore, such approaches have been demonstrated to scale well in producing compact deep neural networks that possess strong modelling accuracy for more complex problems and data.

2.1 Group Lasso regularization

Group Lasso can be used to learn structured sparsity in an efficient way. For compact architecture search purposes, it was applied [7] to regularize network structures such as filters or channels during

training via the following loss function:

$$\mathcal{L}(W) = E_D(W) + \lambda_g \sum_{l=1}^{L} R_g(W^{(l)}) \tag{1}$$

where W represents the set of weights in the network and $E_D(\cdot)$ is the loss of the network on the data D. $R_g(\cdot)$ formulates the group lasso for each layer which enforces the structure sparsity during the training and λ_g is the regularization factor.

2.2 Variational dropout

The variational dropout technique adds a Gaussian stochastic factor on each activation during the training step and the mean and standard deviation of this distribution are learned during the training. The values of these parameters reach to zero for those weights that have no impact on the network output which promotes the sparsity. Therefore, the network weights at the end of training are sparse. However the produced sparsity is unstructured and cannot provide any speed up inference. Here we extend upon this approach and drop those filters (i.e., a set of weights in a layer which produces one activation channel at the output) which are more sparse than a pre-defined threshold to generate the final network architecture. Setting up proper parameters and initialization of the parameters is very important to produce the best results.

2.3 MorphNet

MorphNet [3] performs an iterative shrink-and-expand approach to automatically design the structure of a deep neural network. It utilizes weight regularization on network activations to sparsify the network in the shrinking step, followed by a uniform multiplicative approach on all layers in the expand step. More specifically, a penalty term is applied on the loss function during the training in the shrinking step as follows:

$$\begin{aligned} \mathcal{L}(W) =& E_D(W) + \lambda \mathcal{F}(O_{1:M}) \;\; s.t. \\ & \mathcal{F}(O_{1:M}) = \sum_{L=1}^{M+1} \mathcal{F}(\text{layer } L). \end{aligned} \tag{2}$$

The $\mathcal{F}(\cdot)$ is a regularizer on neurons which can induce some of the neurons to be zeroed out and can be formulated as:

$$\mathcal{F}(\text{layer } L) = C \sum_{i=0}^{I_L - 1} A_{L,j} \sum_{j=0}^{O_L - 1} B_{L,j} \tag{3}$$

where $A_{L,j}$ and $B_{L,j}$ are indicator functions that encode whether the input i or output j are alive or zeroed out. L_1 norm on variables of batch normalization is utilized to provide tractable learning via gradient descent. The batch-norm is applied to each layer which means that each neuron has a particular variable in the batch-norm; setting this variable to zero effectively disables the neuron. The MorphNet method needs hyper-parameter tuning; Tuning the parameters in this method plays a vital rule to produce reasonable results.

2.4 Generative synthesis

The overall goal of the Generative Synthesis method [8] as a compact architecture search method is to learn a generator $\mathcal{G}$ that can synthesize deep neural networks $\{N_s | s \in S\}$, given a seed set S, maximize a universal performance function $\mathcal{U}$ while satisfying quantitative human-specified design constraints and performance targets, as defined by an indicator function $\mathbb{1}_r(\cdot)$. This learning of generative machines

for synthesizing deep neural networks can be formulated as the following constrained optimization problem:

$$\mathcal{G} = \max_{\mathcal{G}} \ \mathcal{U}\Big(\mathcal{G}(s)\Big) \ \ s.t. \tag{4}$$
$$\mathbb{1}_r\Big(\mathcal{G}(s)\Big) = 1, \forall s \in S.$$

Given the intractability of solving this problem, an approximate solution $\mathcal{G}$ to the constrained optimization problem posed is achieved by leveraging the interplay between a generator-inquisitor pair that work together in a synergistic manner to obtain not only improved insights about deep neural networks (via an inquisitor) as well as learn to synthesize compact deep neural networks (via a generator) in a cyclical and iterative manner, taking into consideration human-specified design and operational requirements (size, accuracy tolerance, performance targets, hardware-level requirements such as channel multiplicity and data precision, etc.) when synthesizing progressively more compact deep neural networks until performance targets and operational requirements are satisfied.

3 Empirical analysis

In this study, we further empirically examine the current state of compact architecture search algorithms for producing deep neural networks. The trade-offs between size, speed, and accuracy are measured by conducting several empirical experiments using well-known benchmark datasets. Based on the experimental results, we study the different performance characteristics of the compact deep neural networks produced by the tested compact architecture search algorithms to gain better insights into the effectiveness as well as the search behaviours of the tested algorithms.

3.1 Datasets & networks

The four state-of-the-art compact network architecture search methods studied here are evaluated using CIFAR-10, CIFAR-100 [4], and ImageNet 64x64 [2], a downsampled variant of ImageNet with over 1.28M training images across 1000 classes. In this study, a 32 layer ResNet architecture is used as the baseline architecture for CIFAR-10 while a deeper 50 layer ResNet architecture is used for CIFAR-100 given that they are more difficult tasks. For ImageNet 64x64, two baseline architectures were used for evaluation: i) ResNet-50, and ii) InceptionV3. For the CIFAR-10, CIFAR-100 and ImageNet 64×64 evaluations, the FLOPs target for all four of the tested methods is set to one-third of the respective baseline network architectures to quantitatively investigate the efficacy of the four methods at producing highly compact deep neural networks with low computational costs.

3.2 Results & discussion

The experimental results for CIFAR-10 and CIFAR-100 are shown in Table 1, while the results for ImageNet 64x64 are shown in Table 2.

CIFAR-10/CIFAR-100: As seen in Table 1, the compact deep neural networks produced by each of the tested compact architecture search algorithms have drastically different neural network architectures with very different performance tradeoffs to meet the ∼48 and ∼74 MFLOPs (1/3 of the baseline) performance target for CIFAR-10 and CIFAR-100.

Conducting a close examination of the performance characteristics of the compact deep neural networks produced using the four tested compact architecture search algorithms allowed for the observation of several interesting insights:

- While all produced networks have similar FLOPs as a result of the FLOPs target, several of the produced neural networks have significant more parameters than others. For CIFAR-10, the deep neural network produced via the Group Lasso approach had noticeably more parameters

than the other three methods (e.g., ∼33% higher than Variational Dropout). For CIFAR-100, the deep neural networks produced by MorphNet and Variational Dropout have more than twice the number of parameters as that produced using Group Lasso and Generative Synthesis.

- While all tested methods have the same FLOPs performance target (1/3 FLOPs of baseline), none of the methods produce networks that hit the target exactly. The deep neural network produced by Generative Synthesis had the lowest number of parameters (by around 3% and 7% lower than others for CIFAR-10 and CIFAR-100, respectively).
- Of the methods, Variational Dropout produced the network with the lowest modeling accuracy. MorphNet and Generative Synthesis produced networks with better modeling accuracy than the Group Lasso and Variational Dropout methods, with Generative Synthesis producing the network with the highest accuracy amongst the tested methods for the given FLOPs target. In fact, the network produced by Generative Synthesis achieved the same accuracy as the baseline architecture in the CIFAR-10 case. which illustrates that it was able to find the most balanced trade-off between size, speed, and accuracy.

Table 1: Results for CIFAR-10 and CIFAR-100 across four tested algorithms. Best results in bold

	CIFAR-10			CIFAR-100		
Methods	**MFLOPs**	**#Parameters**	**Top-1(%)**	**MFLOPs**	**#Parameters**	**Top-1(%)**
Baseline	144.369	487,754	91.6	223.946	769,476	67.8
Group Lasso [7]	46.282	225,564	86.7	74.852	268,609	61.6
MorphNet [3]	46.280	153,179	88.8	74.995	435,396	63.3
Variational Dropout [5]	46.046	**151,238**	84.9	74.410	563,197	57.9
Generative Synthesis [8]	**44.658**	152,668	**91.6**	**70.800**	**214,692**	**64.8**

Table 2: Results for ImageNet 64x64 using two baseline architectures. Best results in bold

Architectures	ResNet-50			InceptionV3		
Methods	**MFLOPs**	**#Parameters**	**Top-1(%)**	**MFLOPs**	**#Parameters**	**Top-1(%)**
Baseline	2,629.21	25,593,400	60.0	7,515.7	22,895,000	61.8
Group Lasso [7]	896.71	9,079,334	50.6	2506.7	8,160,687	57.6
MorphNet [3]	824.91	9,294,711	52.9	2490.6	9,489,201	58.5
Variational Dropout [5]	873.29	8,850,781	43.3	2,508.0	8,160,687	53.9
Generative Synthesis [8]	**802.14**	**3,293,420**	**57.8**	**2,340.3**	**5,257,960**	**62.3**

ImageNet 64x64: As seen in Table 2, similar to the CIFAR-10/CIFAR-100 experiments, the compact deep neural networks produced by each of the tested compact architecture search algorithms produce drastically different neural network architectures with very different performance tradeoffs to meet the ∼876 MFLOPs and ∼2505 MFLOPs (1/3 of the baseline) performance targets for ResNet-50 and InceptionV3, respectively.

Experimental results show several interesting insights regarding the performance characteristics of the compact deep neural networks produced using the four compact architecture search algorithms:

- The MorphNet approach produced the largest networks for both cases (in terms of number of parameters). For example, MorphNet produced a network architecture that has ∼2.8× the number of parameters as Generative Synthesis for the ResNet-50 case.
- While all methods have the same FLOPs performance target (1/3 FLOPs of baseline), results demonstrate that, as with the previous experiment, none of the methods produce networks that hit the target exactly. However, Generative Synthesis was able to achieve lower than expected FLOPs (by ∼7% lower for both cases) in this experiment as well.
- As with CIFAR-10 and CIFAR-100, MorphNet and Generative Synthesis produced networks with better modeling accuracy than the Group Lasso and Variational Dropout methods. In both cases, Generative Synthesis produced neural networks that had the highest accuracy and lowest number of parameters amongst the tested methods for the given FLOPs target (e.g., ∼4.9%

higher accuracy than MorphNet in the ResNet-50 case) on ImageNet 64x64. In fact, the neural network produced by Generative Synthesis outperformed the baseline architecture in terms of accuracy by 0.5% in the InceptionV3 case. This illustrates that it was able to find the most balanced trade-off between size, speed, and accuracy.

4 Practical considerations

In order to take advantage of the aforementioned compact architecture search algorithms, there are a number of practical challenges unique to each method that must be considered to achieve good performance in the resulting deep neural networks. One of the biggest challenges to effectively leveraging the Group Lasso regularization technique as a compact architecture search algorithm is to identify the optimal set of hyperparameters, with one of the most crucial hyperparameters being a proper and optimal regularization factor λ_g during the training process. Much like Group Lasso, Variational Dropout also suffers from the *curse of hyper-parameterization*. Selecting the correct regularization strength is key for allowing the model to learn. Too high a rate and a model will not learn at all, too low a rate and the model is not fully learning what parameters should be dropped. MorphNet follows an iterative process of sparsifying a network and linearly growing the network back up. The hyper-parameters that control each step in a given iteration can differ from previous iteration. Selecting the correct rate to sparsify a network is performed via trial and error. Determining when to stop sparsifying a network during any given iteration can be unclear. Generative Synthesis employs an iterative process to learn to generate compact deep neural networks that meet operational requirements and desired performance targets. As such, while Generative Synthesis will iterate until it hits the desired performance targets, a practical challenge with Generative Synthesis is that the number of iterations can vary depending on the complexity of the neural network architecture, the desired performance targets, as well as the complexity of the underlying data.

References

[1] Bowen Baker and et al. Designing neural network architectures using reinforcement learning. 2016.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. IEEE, 2009.

[3] Ariel Gordon and et al. Morphnet: Fast & simple resource-constrained structure learning of deep networks. 2018.

[4] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[5] Dmitry Molchanov and et al. Variational dropout sparsifies deep neural networks. 2017.

[6] Mohammad Javad Shafiee and et al. Deep learning with darwin: evolutionary synthesis of deep neural networks. 2018.

[7] Wei Wen and et al. Learning structured sparsity in deep neural networks. 2016.

[8] Alexander Wong and et al. Ferminets: Learning generative machines to generate efficient neural networks via generative synthesis. 2018.