ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

Compact Neural Architecture Search for Local Climate Zones Classification

Kalifou Rene Traore[1,2] Andrés Camero[2]
Xiao Xiang Zhu[1,2]

1- Technical University of Munich, Data Science in Earth Observation
2- German Aerospace Center (DLR), Remote Sensing Technology Institute (IMF)

Abstract. State-of-the-art Computer Vision models achieve impressive performance but with an increasing complexity. Great advances have been made towards automatic model design, but accounting for model performance and low complexity is still an open challenge. In this study, we propose a neural architecture search strategy for high performance low complexity classification models, that combines an efficient search algorithm with mechanisms for reducing complexity. We tested our proposal on a real World remote sensing problem, the Local Climate Zone classification. The results show that our proposal achieves state-of-the-art performance, while being at least 91.8% more compact in terms of size and FLOPs.

1 Introduction

Modern data-driven approaches to deal with large-scale data, particularly Deep Learning (DL), require domain and data science expertise. The variety of application tasks (e.g., classification and object detection) often require designing models that are not necessarily reusable in other tasks [1]. Moreover, manually designing models is time consuming and error prone. Thus, automating the Machine Learning (ML) pipeline design (a.k.a. AutoML [2]) is much desirable.

Neural Architecture Search (NAS) [3], a sub-field of AutoML focusing on the design of ML models, has already proven to be successful for a wide variety of problems, including Computer Vision (CV) [3].

In Earth observation (EO), satellite remote sensing enables recovering contact-free large-scale information about the physical properties of the Earth from space. Thanks to ESA's Sentinel missions and NewSpace companies, petabytes of satellite data has become available, leveraging large-scale datasets that have boosted the study of tailored models, in particular for multi-spectral and Synthetic Aperture Radar (SAR) data classification [4]. However, the application of NAS for further improvements remains quite unexplored.

In this paper, we propose a NAS strategy to automatically design high-performance low complexity image classification models. Particularly, we combine a *differentiable search strategy* to optimize the elementary normal cells of a backbone architecture, with a *structural depth* and a *complexity reducing loss*. We validate our proposal on a real world problem, the *So2Sat LCZ42* [5] data set for classification of Local Climate Zones (LCZ). Our results show that the best found models are on par with state-of-the-art baselines, while accounting for at least 91.8% less FLOPs and size.

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

2 Related Work

The increase in remote sensing missions and sensors has allowed more data collection, which in turn has motivated the creation of larger EO datasets for scientific purposes [6]. In order to analyse these large data sets, the remote sensing community has embraced the use of DL models classically used in CV for similar tasks [4]. However, due to the particularities of these EO data sets (e.g., multi-modal, geo-located, time-variable data), it is necessary to develop domain-specific models [7]. Therefore, AutoML (and particularly NAS), has the potential to ease and improve the quality of the future tailored models.

Great advances have been made in the NAS field [3, 8]. However, most approaches have not yet been adopted because they require a lot of computational resources and time [3, 8]. To cope with these limitations, some authors have leveraged the *differentiable architecture search* (DARTS) [9], dramatically reducing the search budget time (GDAS [10]), without compromising the performance. On the other hand, some authors have explore NAS for low complexity models, including the design of models for constrained hardware [11, 12], compression-based approaches [13], and pruning techniques [14], among others.

3 NAS for Compactness

A Differentiable Search Strategy: Let $\alpha = \{\alpha_{cell}, \Delta\}$ be a candidate neural architecture defined by an elementary module α_{cell}, a backbone configuration Δ, and ω_{cell} and Ω_{depth} the respective weights. The objective is to find α^* that minimizes a validation loss, while identifying its optimal parameters ω_{cell*} in training:

$$\min_{(\alpha_{cell},\Delta)} \mathbb{E}_{(x',y')\sim\mathbb{D}_V} L(x^{'}, y^{'}) \text{ s.t. } \omega^*_{cell} = \operatorname{argmin}_{\omega} \mathbb{E}_{(x,y)\sim\mathbb{D}_T} L(x,y) \quad (1)$$

The optimized loss L is the sum of a prediction loss L_{NLL} and a weighted complexity reducing loss L_{cpl}: $L(x,y) = L_{NLL}(x,y) + C_{cpl} \cdot L_{cpl}$. Here L_{NLL} is the negative log likelihood of predicting the correct label y, given a data sample x, α_{cell}, Δ and $\omega_{cell} \in \mathbb{W}$. Both α_{cell}, Δ are respectively sampled from τ_{A} and τ_{Δ} the probability distributions over the cells and depths configurations. In practice, τ_{A} and τ_{Δ} are encoded by the matrices A_{cell}, and Ω_{depth}. Thus, the distribution over α is $\mathbb{A} = \{\mathrm{A}_{cell}\} \cup \{\Omega_{depth}\}$. To sample, we use a continuous and smooth Gumbel-Softmax function (i.e., the sampling on α is differentiable), thus we can learn τ_{A} via Gradient Descent. To tackle our Equation 1, we propose to use Algorithm 1 alternating between updating the trainable parameters of an architecture, sampled in $\mathbb{W}$, and its structural parameters sampled in $\mathbb{A}$ [10] .

Cells and Depth Search Space: All considered model α use the backbone architecture introduced in Figure 1 [10]. As in [13], each block i of α is made of a sequence of identical cells α_{cell}, of variable length δ_i.Therefore, α is parametrized by $\Delta = [\delta_1, \delta_2, ..., \delta_N]$, s.t. $\delta_i < \delta_{max}$, the set of depths for all block i. As in [10], we search for the topology of α_{cell}. Such module is defined as a directed acyclic graph $\mathbb{G}$, with an ordered sequence of $\mathbb{B}$ features (nodes). Each node is the resulting transformation of its $\mathbb{T} = 2$ preceding nodes as: $\mathrm{I_i} = f_{i,j}(\mathrm{I_j}) + f_{i,k}(\mathrm{I_k})$ s.t. $j < i$ & $k < i$ where $\mathrm{I_i}, \mathrm{I_j}, \mathrm{I_k}$ represent nodes of respective indices

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

i, j, k. $f_{i,j}$, $f_{i,k}$ represent operations sampled from a set $\mathbb{F}$. The final α_{cell} obtained by selecting the function $f_{i,j}$ between each pair of nodes i and j with the highest probability to appear.

Complexity Reducing Loss: In order to reduce the complexity of a searched α, we formulate a complexity loss to minimize [11, 13]: $L_{cpl} = \sum_{\delta_k \in \Delta} L_{cpl-cell} \cdot \delta_k$, a function of the selected depth δ_k for all k blocks. At a cell-level, it writes as $L_{cpl-cell} = \frac{1}{N_{ops}} \sum_{o_{i,j} \in \alpha_{cell}} FLOPs(o_{i,j}) \cdot \omega_{i,j}$, where N_{ops} is the total number of operations, and $\omega_{i,j}$ the probability of selecting an operation $o_{i,j}$ in α_{cell}. It is made differentiable only with respect to the cells' weights $\omega_{i,j}$.

Architecture Depth Search: We look for the optimal depth of each block in α [13]. This aspect is capture by matrix: $\Omega_{depth} = [\Omega_1, ..., \Omega_i, .., \Omega_N]$, s.t. $\Omega_i \in \mathbb{R}^{\delta_{max}}$ where δ_{max} is the maximal depth per block, and Ω_i the depth parameters for block i. These parameters are involved in the training process as follows: $f_{block_{i+1}} = \sum_{\Omega_i^j \in \Omega_i} f_{block_i} \cdot \Omega_i^j$ where f_{block_i} is the feature map resulting from block i. The final feature map is a sum of the original block's feature map weighted by the parameters Ω_i^j of all potential depth j at layer i.

Algorithm 1: Searching for α_{cell} and Δ

Input: Two disjoint sets $\mathbb{D}_T$ and $\mathbb{D}_V$, randomly initialized $\mathbb{A}$ and $\mathbb{W}$, batch size;
while *not at convergence* **do**
  Sample a batch $\mathbb{D}_t = \{(x_i, y_i)\}_{i=1}^n$ from $\mathbb{D}_T$
  Calculate $L_T = \sum_{i=1}^n L(x_i, y_i)$
  Update $\mathbb{W}$ by gradient descent: $\mathbb{W} = \mathbb{W} - \nabla_{\mathbb{W}} L_T$
  Sample a batch $\mathbb{D}_v = \{(x_i, y_i)\}_{i=1}^n$ from $\mathbb{D}_V$
  Calculate $L_V = \sum_{i=1}^n L(x_i, y_i)$
  Update $\mathbb{A}$ by gradient descent: $\mathbb{A} = \mathbb{A} - \nabla_{\mathbb{A}} L_V$
end
Derive the final architecture $\alpha = \{\alpha_{cell}, \Delta\}$ from $\mathbb{A}$;
Optimize $\alpha = \{\alpha_{cell}, \Delta\}$ on the whole training set for future inference on the test set.

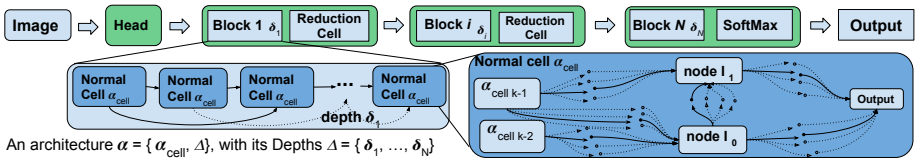

Fig. 1: An architecture α, with a reduction cell is fixed according to [10].

4 Experimental Setup

Data: The So2Sat LCZ42 classification benchmark [5] consist of samples from 42 urban areas (plus 10 smaller areas for validation and test) across all continents. Including 400,673 co-registered Sentinel-1 SAR (8 real-valued bands) and Sentinel-2 multi-spectral (10 real valued bands) image patches and its respective labels. Each patch is a 32x32 pixel image (320x320 meters spatial resolution). The data set defines 17 LCZ classes, ten urban classes and seven vegetation ones.

Data distribution: We consider two distributions using Sentinel-2 and the labels, exclusively. In $d1$, the training-set consist of samples from 42 cities (352,366 patches). Validation (24,188 patches) and test (24,199 patches) contain samples

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

from 10 different cities, respectively taken from east and west part of each city. In $d2$, train and test sets are randomly draw from $d1$ training-set (80% and 20% respectively). Thus, both have the same data distribution. At search time, the train set is split into two halves: $\mathbb{D}_T$ (training) and $\mathbb{D}_V$ (validation).

Hyperparameters: We set the number of channels in the first layer 16, the number of nodes in α_{cell} to 4, the number of incoming cells to 2, and the initial depth δ in all blocks to 3. The max depth per block is set as 5. We use SGD (105 epochs) for training, and we set the learning rate to 0.025, weight decay to 3e−4, annealed to 1e−3, and momentum of 0.9. The Gumbel-Softmax sampler temperature is set to 10 (linearly reduced to 1). Also, we set $C_{cpl} = 0.01$, to enable balancing the optimization of both L_{NLL} and L_{cpl}. We observed that larger values tend prevent the convergence of the overall objective L. Additionally, following [10], we set $\mathbb{F}$ as: the (1) identity, (2) zeroize, (3) 3x3 depth-wise separate convolutions, (4) 3x3 dilated depth-wise separate convolutions, (5) 5x5 depth-wise separate convolutions, (6) 5x5 dilated depth-wise separate convolutions, (7) 3x3 average pooling, (8) 3x3 max pooling. The stride is set to 1 for normal cells, and to 2 for reduction cells.

Implementation: We used GDAS [10] and TAS [14] source code as basis, and *ResNet08*, *ResNet32* and *ResNet110* models (available on the same repository) as baselines. The experiment were run on a Nvidia V100 GPU.

5 Results

Performance in $d1$: Table 1 presents the mean performance on test data (4 independent runs). S stands for Single Cell Search, C for Complexity Loss, D for Depth Search, and '+' indicates the combination of these settings. The individual effect of Depth Search **(S + D)** and the Complexity Aware Loss **(S + C)** is positive in improving performance on both OA, AA, and Kappa metrics over Single Cell Search **(S)**. The combination **(S + C + D)** does not accumulate their benefice. The model **(S + D)** brings the most improvements while keeping the complexity low over, compared to **(S)**. When compared to manually designed baselines (Table 1 - left), **(S + D)** is a TOP-2 low complexity model (FLOPs and Size), a TOP-2 performer for two accuracy metrics (OA and Kappa), while best on the AA metric. The best selected model **(S + D)** outperforms manually designed baselines of similar complexity. Also, all selected models are performing well in test on $d1$ while being selecting on data distribution $d2$.

Method	*ResNet08*	*ResNet32*	*ResNet110*	*S*	*S+C*	*S+D*	*S+C+D*
OA	60.25	64.85	67.60	64.05	64.25	64.96	64.00
AA	47.86	52.24	53.34	50.39	51.63	54.02	52.20
Kappa	0.565	0.615	0.645	0.61	0.61	0.62	0.61
Size (MB)	0.08	0.27	1.73	0.224	0.165	0.14	0.130
FLOPs (M)	13.53	41.85	253.89	30.37	24.10	20.05	18.42
Block's Depth	NA	NA	NA	3x3	3x3	1x3	1x3

Table 1: Mean performance benchmark in test for setting $d1$.

Performance in $d2$: Table 2 presents the mean performance for 4 independent runs on $d2$. The individual effect of Depth Search **(S + D)** and the Complexity Aware Loss **(S + C)** are non beneficial over Single Cell Search **(S)**

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

in test. Combining with **(S + C + D)** does not improve performances neither. However, overall performance of all models are very good, suggesting that the $d2$ data distribution is much easier to fit than $d1$. The best model found using the proposed approach is **(S + D)**, with 98% OA, 91.44% AA and 0.98 Kappa.

Method	*ResNet08*	*ResNet32*	*ResNet110*	*S*	*S+C*	*S+D*	*S+C+D*
OA	96.18	96.89	98.59	98.15	97.81	98.00	97.74
AA	88.91	89.75	91.91	91.61	91.12	91.44	91.08
Kappa	0.958	0.966	0.984	0.980	0.975	0.980	0.975
Size (MB)	0.08	0.27	1.73	0.224	0.165	0.14	0.130
FLOPs (M)	13.53	41.85	253.89	30.37	24.10	20.05	18.42
Block's Depth	NA	NA	NA	3x3	3x3	1x3	1x3

Table 2: Mean performance benchmark in test for setting $d2$.

Compared to modern classification baselines (Table 2 - ResNets), the model **(S + D)** is TOP-2 in performance and is very close (0.59% - OA, 0.47% - AA and 0.004 - Kappa) to ResNet110 (TOP-1), while it is nearly 92% smaller (FLOPs and Size) than ResNet110. Also, it outperforms similar complexity competitors.

Confusion: Figure 2 depicts the confusion matrix of (S + D) on $d1$ (left) and $d2$ (right). The confidence increases from blue to yellow. In both cases, the confusion is higher for urban classes (1 to 10). This resemblance suggests that the 42 cities selected on $d1$ and the 10 introduced on $d2$ are similar. Also, the confusion for the vegetation classes (11 to 17) is lower on $d2$ than on $d1$. This difference could be explained by the vegetation disparity around the globe.

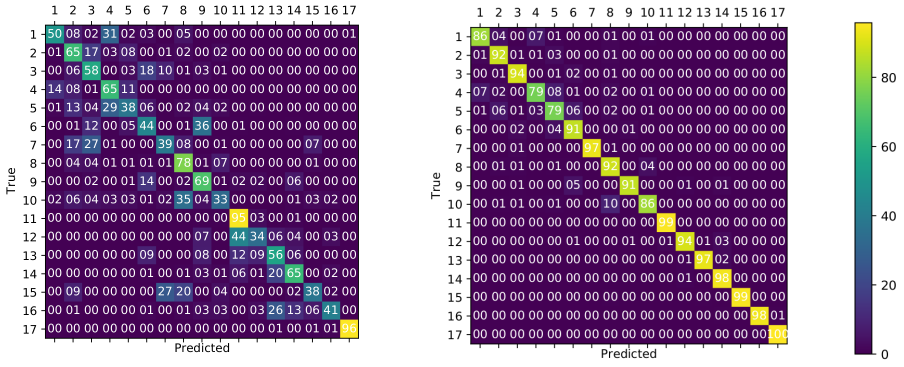

Fig. 2: Confusion matrix for model (S + D) on test $d1$ (left) and $d2$ (right).

6 Conclusions and Future Work

In this paper, we proposed a differentiable search strategy combined with two mechanisms to reduce the model complexity for finding high performing and low complexity classification models. We tested our proposal on a real World problem, the So2Sat LCZ42, using two different data distributions ($d1$ and $d2$). In test, our approach achieves state-of-the-art performance (ResNet-110), while being 91.8% smaller in both FLOPS and size, and outperforms all baselines of similar complexity (ResNet-08). However, in test, these cumulative improvements are not consistent, in particular on the data distribution $d2$. As future

ESANN 2021 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 6-8 October 2021, i6doc.com publ., ISBN 978287587082-7.
Available from http://www.i6doc.com/en/.

work, we propose to investigate how to improve model selection in validation to get the most of fully retrained models in test.

Acknowledgments

Authors acknowledge support by the European Research Council (ERC) under the EU Horizon 2020 (ERC-2016-StG-714087, *So2Sat*), the Helmholtz Association through the Helmholtz AI (ZT-I-PF-5-01), the German Federal Ministry of Education and Research (BMBF) international future AI lab "AI4EO" (01DD20001), and DeToL.

References

[1] Krizhevsky Alex, Sutskever Ilya, and E. Hinton Geoffrey. Imagenet classification with deep convolutional neural networks. *NIPS*, 2012.

[2] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors. *Automated Machine Learning - Methods, Systems, Challenges*. Springer, 2019.

[3] Thomas Elsken, Jan Hendrik Metzen, Frank Hutter, et al. Neural architecture search: A survey. *JMLR*, 20(55):1–21, 2019.

[4] Xiao Xiang Zhu, Devis Tuia, Lichao Mou, Gui-Song Xia, Liangpei Zhang, Feng Xu, and Friedrich Fraundorfer. Deep learning in remote sensing: A comprehensive review and list of resources. *IEEE Geosci. Remote Sens.*, 5(4):8–36, 2017.

[5] Xiao Xiang Zhu, Jingliang Hu, Chunping Qiu, Yilei Shi, Jian Kang, Lichao Mou, Hossein Bagheri, Matthias Haberle, Yuansheng Hua, Rong Huang, et al. So2sat lcz42: A benchmark data set for the classification of global local climate zones [software and data sets]. *IEEE Geosci. Remote Sens.*, 8(3):76–89, 2020.

[6] John E Ball, Derek T Anderson, and Chee Seng Chan. Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. *J. Appl. Remote Sens.*, 11(4):042609, 2017.

[7] Qingpeng Li, Lichao Mou, Qizhi Xu, Yun Zhang, and X. X. Zhu. R3-net: A deep network for multi-oriented vehicledetection in aerial images and videos. *IEEE Tran. Geosci. Remote Sens.*, 2019.

[8] Varun Kumar Ojha, Ajith Abraham, and Václav Snášel. Metaheuristic design of feedforward neural networks: A review of two decades of research. *Eng. Appl. Artif. Intell.*, 60(January):97–116, 2017.

[9] Liu Hanxiao, Simonyan Karen, and Yang Yiming. Darts: Differentiable architecture search. *ICLR*, 2019.

[10] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. *CVPR*, 2019.

[11] Cai Han, Zhu Ligeng, and Han Song. Proxilessnas: Direct neural architecture search on target task and hardware. *ICLR*, 2019.

[12] Xie Sirui, Zheng Hehui, Liu Chunxiao, and Lin Liang. Snas: Stochastic neural architecture search. *ICLR*, 2019.

[13] Chen Daoyuan, Li Yaliang, Qiu Minghui, Wang Zhen, Li Bofang, Ding Bolin, Deng Hongbo, Huang Jun, Lin Wei, and Zhou Jingren. Adabert: Task-adaptive bert compression with differentiableneural architecture search. *Arxiv*, 2019.

[14] Xuanyi Dong and Yi Yang. Network pruning via transformable architecture search. In *NeurIPS*, 2019.