BABEŞ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Optimizing Neural Network Architectures using Computational Intelligence Methods

PhD thesis summary

PhD student: Sergiu Cosmin Nistor
Scientific supervisor: Prof. dr. Czibula Gabriela

2021

Keywords: Neural architecture search, Graph neural networks, Recurrent neural networks, Convolutional neural networks.

Contents

Thesis table of contents

Glossary

CNN convolutional neural network. 7–10, 14–18

DAG directed acyclic graph. 8, 14–16, 18

DAGRNN Directed Acyclic Graph Recurrent Neural Network. 15, 16, 18, 19

EA evolutionary algorithm. 8, 11, 12, 17, 19

ERMC Empathic Recurrent Memory Cell. 18

GERNN Graph-Encoding Recurrent Neural Network. 12, 17–19

GNN graph neural network. 10, 17–19

GRU Gated Recurrent Unit. 12, 17

IntelliSwAS Intelligent Swarm Architecture Search. 8, 14, 15, 18

LSTM Long Short-Term Memory. 12, 17

ML machine learning. 7, 8, 10, 12, 14–16, 18

NAS neural architecture search. 7, 8, 10–12, 15, 17–19

PSO Particle Swarm Optimization. 8, 9, 14, 15, 18

RNN recurrent neural network. 7–18

List of publications

The ranking of publications was performed according to the CNATDCU (National Council for the Recognition of University Degrees, Diplomas and Certificates) standards applicable for doctoral students enrolled after October 1, 2018. All rankings are listed according to the classification of journals[1] and conferences[2] in Computer Science.

Publications in Web of Science - Science Citation Index Expanded

[NC22] **Sergiu Cosmin Nistor** and Gabriela Czibula *IntelliSwAS: Optimizing Deep Neural Network Architectures using a Particle Swarm-based Approach*. Expert systems with Applications, Volume 187, January 2022, 115945 (**2020 IF=6.954**, 2020 Journal IF Quartile Q1)

Rank A, 8 points.

[NMM+21] **Sergiu Cosmin Nistor**, Mircea Moca, Darie Moldovan, Delia Beatrice Oprean and Răzvan Liviu Nistor. *Building a Twitter Sentiment Analysis System with Recurrent Neural Networks*. Sensors (2021), 21, 7, 2266, MDPI (**2020 IF=3.576**, 2020 Journal IF Quartile Q1).

Rank A, 2.66 points.

[NMN21] **Sergiu Cosmin Nistor**, Mircea Moca and Răzvan Liviu Nistor. *Discovering Novel Memory Cell Designs for Sentiment Analysis on Tweets*. Genetic Programming and Evolvable Machines (2021): 22, 2, 147-187, Springer (**2020 IF=1.714**, 2020 Journal IF Quartile Q2).

Rank B, 4 points.

[NID20] **Sergiu Cosmin Nistor**, Tudor Alexandru Ileni and Adrian Sergiu Darabant. *Automatic Development of Deep Learning Architectures for Image Segmentation*. Sustainability (2020), 12, 22, 9707, MDPI (**2020 IF=3.251**, 2020 Journal IF Quartile Q2).

Rank B, 4 points.

Publications in Web of Science - Conference Proceedings Citation Index

[NDB18] **Sergiu Cosmin Nistor**, Adrian Sergiu Darabant and Diana Borza. *Micro-Expressions Detection Based on Micro-Motions Dense Optical Flows*. 2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1-7. IEEE, 2018 (**indexed IEEE**).

Rank B - CORE2018, 4 points.

[1] https://uefiscdi.ro/premierea-rezultatelor-cercetarii-articole
[2] http://portal.core.edu.au/conf-ranks/

[BND17] Diana Borza, **Sergiu Cosmin Nistor** and Adrian Sergiu Darabant. *Towards Automatic Skin Tone Classification in Facial Images*. International Conference on Image Analysis and Processing (ICIAP), pp. 299-309. Springer, Cham, 2017.

Rank B - CORE2017, 4 points.

[NMDB17] **Sergiu Cosmin Nistor**, Alexandra-Cristina Marina, Adrian Sergiu Darabant and Diana Borza. *Automatic gender recognition for "in the wild" facial images using convolutional neural networks*. 2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 287-291. IEEE, 2017.

Rank C - CORE2017, 1 point.

[Nis20] **Sergiu Cosmin Nistor**. *Multi-Staged Training of Deep Neural Networks for Micro-Expression Recognition*. 2020 IEEE 14th International Symposium on Applied Computational Intelligence and Informatics (SACI), pp. 000029-000034. IEEE, 2020.

Rank D - CORE2020, 1 point.

[Nis21] **Sergiu Cosmin Nistor**. *An Actor-Critic Approach to Neural Network Architecture Search for Facial Expressions Recognition*. 2021 17th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), IEEE, 2021, accepted for publication.

Rank D - CORE2021, 1 point.

Publications score: 29.66 points.

Introduction

The fast progress of computer science has enabled us to automate many processes. So many applications were proposed that there exist very few domains that did not yet benefit of this progress. This offers two important perspectives: what the already automated processes generate and what can be automated in the future. The first perspective refers to the data that is generated. This data can be analyzed and the results can be used to improve the quality of the automation, or to move towards new applications.

Even though we have seen such progress, there is still much to be done. Currently, the involvement of human experts is still necessary in many domains. Machine learning has helped much in providing solutions based on the large quantities of data that are available. Many applications are based on such intelligent solutions, but almost every application needs careful studying and development of the algorithm, which can only be done by human experts.

For each new application, the human experts usually select an existing architecture which has previously proven successful on other tasks. This architecture is adapted to the new use case. In order to evaluate the quality of the architecture, it is trained and tested. The results are analyzed by the human experts and adjustments are made to the architecture. The new candidate is evaluated. This repetitive process is necessary for finding an efficient solution and it requires the involvement of the human experts for conducting rigorous analyzes and making important decisions.

The research community developed an interest for automating the process of finding the optimal neural network architecture for a given task, and so the domain of neural architecture search (NAS) emerged. These algorithms automate as many parts the architecture searching process as possible. They were applied for both convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Ideally, such a solution would quickly provide the optimal architecture for a given task, without human intervention. Even though the applicability potential of this domain is great and the results that were obtained are impressive, there are still many challenges that need to be overcome.

Our previous experience required us to propose solutions for which we either handcrafted features or experiment with many architectures which we manually studied, compared and adjusted. These are classical approaches in machine learning projects and they have truly proven successful in many situations. Even so, the process could be greatly improved by automating it. Many successful handcrafted architectures were proposed and then reused in multiple new applications. While using such an architecture leads to a high chance of success, it can also be a limitation, as other options are left unexplored. NAS algorithms can be created to find efficient architectures that are not constrained to such a small number of options.

We proposed our own types of NAS algorithms. Our solutions can be used for proposing both CNN and RNN architectures. For demonstrating the quality of our algorithms, we applied them on two tasks which are both challenging and also have a high applicability: image classification and sentiment analysis on tweets.

Original contributions

We split the presentation of our original contributions based on the type of neural networks for which we search for efficient architectures: recurrent neural networks and convolutional neural networks. The methods that we proposed were designed to take advantage of the particularities of these types of networks.

Chapter 2 is dedicated to RNNs. We present a series of methods that we used for finding efficient architectures for a given task. The task that we selected is sentiment analysis on tweets, a challenging problem that can be modelled as a sequence processing task, which makes it approachable using RNNs.

The first approach that we used is an experimental one that follows the classical methodology of searching for an architecture, that has successfully been used previously. This methodology requires considerable involvement from the researchers throughout the entire search process. The results of this approach were published in [NMM+21]. We defined here the way in which we model the problem and experimented with multiple design decisions. Various preprocessing options and different architectural decision were explored. We present these experiments and their results. These results were used to develop the other, more sophisticated, approaches.

The second approach is an original NAS algorithm, which we also published in [NMN21]. We proposed an evolutionary algorithm (EA) that has the purpose of discovering novel recurrent memory cells. The algorithm, with its components, is described in detail. We proposed performance estimation strategies for reducing the time required for identifying the high-quality cells. Many cells were discovered and we present the best performing ones, which we also compare to baselines based on generic-purpose cells.

The third approach dedicated to RNNs focuses on the performance estimation strategy. While the previous original contribution that we made also included such strategies, they still required long running times. Our newer strategy is to use a machine learning algorithm for the performance estimations. We propose a novel model for predicting how well a recurrent memory cell will perform based on its structure. Extensive experiments were conducted to find an algorithm that would make accurate predictions of performance. Using these algorithms, we analyzed new recurrent memory cells and we present the best one.

Chapter 3 presents our approach for discovering novel CNN architectures. For this type of networks, we decided to use image classification as the task for which we search the architectures. This original contribution was published in [NC22]. The method that we proposed is called *IntelliSwAS* and is a particle-swarm approach which we enhanced with a machine learning algorithm for fast performance estimation.

We proposed a Particle Swarm Optimization (PSO) algorithm which searches for CNN cells that can be used to build larger architectures. This algorithm is highly configurable, as we wanted to be able to find an efficient search strategy. The particles move through a search space that represents the CNN cells, for which we proposed a numerical representation. We created a simulation environment in which we could search for configurations that would produce high-performance architectures. In this simulation environment, we ran our PSO algorithms with various options for the parameters and evaluated these results.

The particles need to know which are the positions of high quality and it is desirable that this is done with as few computational resources as possible. Towards this purpose, we propose a novel machine learning algorithm specifically designed for processing data structured as directed acyclic graphs (DAGs). This algorithm we use for predicting the relative quality of the candidate architectures. We made experiments for creating this algorithm and we also compared its efficiency with a

baseline model based on work that was previously proposed in the scientific literature.

The PSO parameter configurations and the performance estimation model were used to discover novel efficient CNN cells. These cells were used to build full CNN architectures and experiments were conducted to evaluate the quality of these architectures.

Thesis structure

The first chapter of this thesis is dedicated to the work that inspired our research. We build upon this work, and due to this reason we decided to give an overview of the multiple topics that influenced us. While this review is not exhaustive and many important scientific contributions were not included, we hope to give an idea of the building blocks that we used, the progress that was made and the challenges that exist.

Chapter 2 presents our original contributions in pursuit of discovering recurrent neural network architectures.

In Chapter 3 we present our original contributions that we made for automatically searching for CNN architectures.

We draw the conclusions in the final chapter. Here we present an overview of what we presented and our results. Possible future research directions that we envision are also described.

Chapter 1

Background

This chapter is dedicated to reviewing scientific contributions that were relevant for our research. Firstly, two of our sections are dedicated to the two machine learning algorithms that we aim to find efficient architectures for: recurrent neural networks and convolutional neural networks. Moreover, our proposed solutions also employ RNNs for the search, so it is important to detail the way in which they function and what are their strengths. We afterwards offer a detailed view of the evolution of neural architecture search algorithms for creating the context in which we conducted our research and proposed our own versions of such algorithms. As neural networks can easily be represented as graphs, we considered that studying graph neural networks (GNNs) can offer us important opportunities for processing the architectures. In later chapters we will present our own versions of GNNs which we proposed for improving our NAS algorithms. A section is also dedicated to sentiment analysis on tweets, as this is the task that we chose for our architecture search solutions for RNNs. For CNNs architectures, we selected image classification, as this is an important task in the domain of computer vision and it is strongly related to the evolution of CNNs, as we also mention in the section which is dedicated to this type of networks.

Chapter 2

Novel Methods for Recurrent Neural Network Architecture Search

In this chapter we present our approaches towards discovering novel recurrent neural network architectures. We present multiple solutions, each one building upon the previous. For these solutions, we selected as the task of interest sentiment analysis on tweets. This task is both useful as it is a sequence processing task, for which RNNs are an appropriate algorithm choice, and it is also a relevant task, receiving much interest in the scientific community due to its important applications. Even though we selected this task, our methods are applicable to many other problems with minimum need for modifications, as it will be explained in the following sections.

We also published these approaches in two original papers: [NMM+21, NMN21].

The first approach is to search in the classical manner of creating architectures, evaluating their quality by training them until convergence, testing them and using the insights that we obtain to propose new architectures. For this, we looked at the architecture from a macro perspective, by using two existing RNN memory cells and adjusting multiple design decisions of how to use them in a full RNN architecture. We proposed and previously presented this approach in [NMM+21]. This phase allowed us to asses the performance of different architectural decisions. We experimented with multiple architectures based on the concepts presented in Chapter 1. Our solution first converts the text into a numerical representation. Afterwards, preprocessing is applied on the input. The information is passed to one or more recurrent layers for processing. Optionally, an attention mechanism is then applied. The final classification is done by a feed-forward layer, which outputs the scores for each considered sentiment, positive or negative, the larger score deciding the class. The input text is processed character-by-character and no part of the tweet text is filtered out. This decision was made to take into account all available information when analyzing the tweet. Other works [KWM11, PP10] use a dictionary of words. By processing only alphanumeric characters and grouping them into words, information like emoticons, emojis, and hashtags is lost. Even if these features are treated separately, by using a predefined dictionary of words, creatively spelled words would be lost because they are not in the dictionary, or they would require to be corrected, which could introduce errors in the text and distort the result.

The second approach is a neural architecture search one. We developed an evolutionary algorithm which we use for searching for RNN architectures. We previously published this work in [NMN21]. For this approach, we looked at the architectures from a micro perspective by searching for new RNN memory cells. Full architectures are built based on these cells and templates that we created based on the insights gained from the classical search that we conducted in the previous approach. Our method is a full EA with all components, which are described in detail. For the fitness function, we evaluate

the quality of the RNN architectures using multiple techniques, but no machine learning. Using the results obtained in the previous approach, we selected the larger architecture design and we decided to change the architecture of the hidden units, by proposing alternatives to to generic-purpose LSTM [HS97] and GRU [CVMG+14]. We used three different tasks to discover and evaluate the designs. We conducted experiments and the results show that the best obtained designs surpass the baselines—which are the most popular cells, LSTM and GRU. During the discovery process we evaluated roughly 17000 cell designs. The selected winning candidate outperformed the others for the overall sentiment analysis problem, hence showing generality. We made the winner selection by using the cumulated accuracies on all three considered tasks.

The algorithm used in our solution is an evolutionary algorithm. This type of algorithm simulates the evolution of species. A population of individuals is used, each individual representing a possible solution to the problem.

In our problem space, an individual represents a candidate RNN memory cell. The structure of each individual is given by their genes. To form new solutions, the genes of individuals are combined and then mutated. The individuals are ranked by their fitness and the top-performers are favored for passing their genes. The components of our EA are:

- Individual representation;
- Selection operator;
- Crossover operator;
- Mutation operator;
- Population initialization;
- Fitness evaluation.

Our third approach is centered around a more sophisticated performance estimation strategy based on a machine learning algorithm. Again we search for RNN memory cells, but we speed up the process by proposing a novel algorithm for predicting the quality of our designs, algorithm that runs considerably faster than the previous methods that we employed for this purpose. While experimenting with the previous approach, we observed that evaluating the performance of a candidate architecture takes a large amount of time, even when using the performance estimation strategies that we proposed. In order to reduce this time, we propose a graph-oriented machine learning algorithm which we use for estimating the performance of a recurrent memory cell on a given task. We present the architecture of the estimation algorithm, discussing each component. We include this new algorithm into a full NAS method which we describe in detail. Using this algorithm, we were able to evaluate one million recurrent memory cell architectures and we discovered novel designs that obtain good performances on sentiment analysis. We describe the discovered design that obtains the best performances.

Our Graph-Encoding Recurrent Neural Network (GERNN) takes as input graphs which are composed of a set of nodes and a set of edges between the nodes. Each node may have any number of properties, but all nodes must have the same set of properties. Similarly, edges may have their own set of properties. Graphs may be directed or undirected. The output of our GERNN depends on the problem to be solved, as our algorithm can be used for both classification and regression problems.

Our GERNN is an algorithm composed of recurrent neural networks and attention mechanisms and due to this, the inputs must have numerical representations. Each node and each edge is converted into a feature vector which describes the properties associated to the node / edge. The numerical

representation of the graph is a list of node-vectors and edge-vectors. This sequential representation is also imposed by the RNNs composing the algorithm, as RNNs are designed for sequence processing [LBE15].

After preprocessing, the nodes are processed by a specific RNN. A second RNN is used to process the adges. Between the two RNNs there is an attention mechanism that is used to better select which are the most important node-processing results for edge-processing. Two other attention mechanisms are used to aggregate the node-processing results and edge-processing results into a feature vector from which the final prediction is made.

The presentations in this chapter are based on the original papers that we previously published [NMM+21, NMN21].

Chapter 3

Novel Methods for Convolutional Neural Network Architecture Search

In this chapter we present our original method of automatically searching for convolutional neural network architectures using a Particle Swarm Optimization approach. We published our work in [NC22].

The contribution of this work is twofold. Firstly, we propose a novel machine learning model designed to process CNN architectures and estimate their performance. The proposed model extends RNNs such that they are better suited to process data structured as directed acyclic graphs [FSZ14], not just sequential data. DAGs are graphs with directed edges (the data flows in only one direction) and no cycles inside the graph (no vertex can be a child of itself).

As CNN architectures have such a DAG structure, computational units being represented as vertices and the data flow represented as directed edges, our model can be efficiently used to analyze them and make predictions. We use this model to speed up the search. We call our approach Intelligent Swarm Architecture Search (*IntelliSwAS*), as we use PSO for searching for CNN architectures, but we also enhance our algorithm with the machine learning model. The way in which we use our model for performance estimation is also novel. We do not employ the usual approach of estimating the performance of a given architecture, but we consider pairs of architectures and we predict the relative quality of the two candidates.

For the second contribution of this work, we experiment with our *IntelliSwAS* approach and we evaluate the performance of the resulting CNN architectures. Comparisons to the image classification state-of-the-art methods and statistical analysis were conducted and we observed and reported that our solution surpasses many other options, showing that our approach provides a significant improvement.

During the design and implementation of our proposed approach, we were guided by the following research questions:

RQ1 How to extend recurrent neural network models such that they could be better suited to process data structured as DAGs?

RQ2 Could such a DAG-oriented machine learning model be used to predict the performance of a full convolutional neural network architecture based on the structure of the main building block of this CNN?

RQ3 To what extent can this machine learning model, in combination with a particle swarm approach, be able to discover high-performance CNN architectures for image classification?

RQ4 Does *IntelliSwAS* significantly improve the image classification performance compared to existing related work?

For answering RQ1, we propose an extension to RNNs which we describe in detail. The proposed model is used to answer RQ2. Our approach is different that the usual one taken in NAS algorithms. While other solutions estimate what the quality of a single architecture might be, we propose a model that compares two candidate architectures. To this purpose, we experiment with estimating the relative quality of various CNN architectures and evaluate the ability of our model to solve this problem. We present these experiments and also a comparison of the model with an existing alternative. In relation to RQ3, we propose our *IntelliSwAS* algorithm. A statistical analysis is then performed for answering RQ4 and highlighting the performance of *IntelliSwAS* with respect to existing image classification models. We discovered novel CNN architectures which we also describe.

Our algorithm searches for CNN cells. We build full CNN architectures by stacking multiple instances of the same cell. A fixed template describes how the cell instances are organized.

Each cell has a fixed number of computational nodes. Each node takes the input from other nodes, concatenates it on the depth dimension and applies an operation on this data to produce its output. The computational nodes, and so, the cells, take as input a volume of size $H \times W \times C$ and produce an output volume of size $H \times W \times C'$. As it can be observed, the cells do not affect the height and the width of the input, but the number of channels can be changed, depending on the number of the filters. Even though the nodes are independent computational units, we decided that all nodes of a cell will have the same number of filters for homogeneity. Each node has an associated operation that it applies.

The cells can be represented as DAGs. Each cell has an input node, an output node and intermediary nodes. We consider an ordering of the nodes and to preserve the DAG property (not having any cycles), each node may have connections only to the nodes that are before it. So, the first intermediary node can only be connected to the input node, the second intermediary node can be connected to the input node and to the first intermediary node and so on.

A cell can be represented by its adjacency matrix and a list of operation identifiers (one for each node). As the input node does not need description, we can ignore its line from the adjacency matrix. Also, no node of the current cell will take as input the value from the output node, so the column for this node can be ignored. We consider 8 possible operations for our nodes, so each operation identifier can be represented on 3 bits. Because we imposed that nodes may take the input only from the nodes previous to them in the ordering, the adjacency matrix will have only zeros above the main diagonal. In this way, we can represent a cell as a sequence of bits, that can be converted to a decimal number.

We search for the best CNN architecture using PSO. Our algorithm is customizable through a set of parameters. A population of particles searches through the possible cells of size N. Each algorithm run employs a constant population size, which is the number of particles which will search for efficient CNN cells. Each particle has a position, which is an integer number and travels through the search space with a velocity. Each position encodes a CNN cell and the conversion is done as we previously explained. N is a parameter of our search and we run our PSO searches for different values of this parameter.

We propose *Directed Acyclic Graph Recurrent Neural Network (DAGRNN)*, our answer to **RQ1**. In the general case, our machine learning model is used to process data structured as DAGs. In the particular case of *IntelliSwAS*, we use DAGRNN as a performance estimation strategy. Most performance estimation strategies presented in the scientific literature deal with a single architecture. This architecture is taken as input and its estimated performance is the output. Our approach is different. We take two architectures and use our model to estimate their relative quality.

We need a machine learning algorithm capable of comparing the performances of two CNN architectures without training and testing these CNN. Our algorithm bases its prediction on the structure of the CNNs. The two CNNs that need to be compared are based on two CNN cells. We created a variation of RNNs to better represent and process the DAG structure of the cells.

Our DAGRNN model will process the graph node by node. While RNNs take into consideration the internal state at the previous time step, DAGRNNs take into consideration the internal states at all the nodes which have an edge that connects to the currently processed node. As we need these internal states to be available when processing the current node, DAGRNNs use a topological ordering. Even though multiple topological orderings may exist for the same DAG, the computations done by DAGRNNs are the same for each such ordering.

The presentation in this chapter is based on the original papers that we published [NC22, NID20, Nis21].

Conclusions and Future Work

Intelligent algorithms helped us in automating many processes across a wide variety of domains. Even though so many applications were found to such algorithms, they still have a great potential of solving many other problems. The challenge is to design the algorithm that can efficiently solve the task. For this, the classical approach requires human experts to continuously propose and evaluate candidate neural network architectures until a satisfactory result is obtained.

In this thesis we focused on the domain of neural architecture search, which has the purpose of automating the proposal of an efficient algorithm for a given task. By using artificial neural networks, NAS automatically experiments with multiple candidate architectures towards the purpose of finding the most suitable one.

We presented multiple original solutions that we proposed in this domain. Our solutions were grouped into two main categories based on the type of neural networks that we searched for: recurrent neural networks and convolutional neural networks. Even though some components of a NAS algorithm can be adapted to be used for both types of algorithms, these neural networks have different particularities that we wished to consider in our solutions.

For RNNs, we presented three approaches. As we needed a task on which to work, we selected sentiment analysis on tweets.

The first approach is an experimental study in which we experimented with multiple architectural decisions for building an RNN for sentiment analysis on tweets. We concluded that the best preprocessing is, by far, the use of one-hot vectors. We observed that there is no significant difference in the results when using LSTM or GRU, the two most popular memory cell designs. The attention mechanisms that we employed at the network level did not produce improvements to our method. We also observed what were the numbers of layers and numbers of units that produced the best performances.

Using the insights provided by the first approach, we proposed a novel NAS algorithm for searching for new memory cells that would produce efficient RNNs for sentiment analysis on tweets. Our proposed solution is a evolutionary algorithm that we specifically designed to work with RNN memory cells. To ensure the generality of the solutions, we used three tasks on which we tested the memory cells. Besides the one based on a large dataset, we also used other two tasks based on a smaller dataset to speed up the search, but also to see if cells discovered for one task would perform well on another. We obtained highly performant designs, equaling and even surpassing the baselines which used the classical LSTM or GRU cells. We evaluated roughly 17000 designs, from which we selected the best performing ones and discussed them in detail.

For the third approach, we proposed a more sophisticated performance estimation strategy. The solution that we proposed is a novel graph neural network, which we named Graph-Encoding Recurrent Neural Network. We defined it and used for estimating the quality of recurrent memory cells.

Based on the dataset of pairs of RNN cells and corresponding accuracies that we built, we trained multiple GERNN architectures. The GERNNs were compared based on the results and the various design decisions were discussed. We used the best-performing ones to search for new RNN cells.

Using GERNN, we evaluated one million new architectures that we generated, we discovered

novel recurrent memory cells and we described and presented the one that obtains the best performances, which we called ERMC.

Even though the newly discovered cell did not surpass the existing alternatives that we considered, it obtained similar results, proving that our methodology can lead to successful new RNN cell designs. Our GERNN proved to help in quickly finding new designs, but at this point we only paired it with random search. Combining our performance estimation algorithm with a more complex search strategy should produce even better results.

Using our experiments, we demonstrated that GERNN is able to accurately estimate the quality of an RNN on a given task based on the structure of the cell that is used. Moreover, GERNN is able to make this estimation much faster that other performance estimation strategies helping to improve the running times of neural architecture search algorithms. GERNN is generic enough to make predictions for RNNs on any task and generic enough to be included in other NAS algorithms.

The second type of neural networks that we focused our attention towards were convolutional neural networks. We proposed, described and analyzed *IntelliSwAS*, a method of discovering efficient deep CNN architectures for image classification. For searching, we used PSO. Our particles explore the search space which contains candidate CNN cells and communicate among them to be able to converge towards efficient solutions.

IntelliSwAS enhances PSO with a machine learning model that speeds up the search: Directed Acyclic Graph Recurrent Neural Network. We designed this model to process the specific structure of CNN cells, which are naturally represented as directed acyclic graphs. We performed extensive experiments for validating *IntelliSwAS*. We compared the ability of DAGRNN to predict the relative quality of two cells to another model previously introduced in the literature. DAGRNN showed it is clearly superior in generalizing, which was expected due to its good fit with the structure of the data to be processed.

We searched for the best configurations for our version of the PSO algorithm and we analyzed the best resulting configurations. Then, we used these configurations to perform the actual search of the cells. Of the discovered cells, we considered only the ones with the best performance for their respective sizes. These high-quality cells were then integrated into a larger CNN architecture which we tested on multiple image classification datasets with great success. We performed statistical analysis to prove that our results truly improve upon the results reported in the scientific literature.

Future research directions

Our work can be extended in multiple directions. *IntelliSwAS* proved to be efficient in finding high-quality CNN cells, however these cells needed to be manually integrated into larger CNN architectures. Similarly, the architecture template that we used for RNNs was selected by manual search. This is a limitation of our methods, which is given by the exponentially large search space that would need to be explored in order to discover full CNN or RNN architectures. We plan to further explore various methods that could extend our solutions to searching for full architectures and find high-quality results in a reasonable amount of time.

Considering the two new GNN types that we proposed, GERNN and DAGRNN, these two could be integrated into different NAS algorithms or even used together. DAGRNN takes advantage of the DAG structure of CNNs, but there is no reason not to also experiment with GERNN in this context. DAGRNN should perform better, but such experiments would indeed be interesting. Using DAGRNN for RNNs cannot be done in its current form, as RNNs are not DAGs, but a possible extension could be created towards this purpose.

As we currently just used GERNN together with random search, a possible future direction is to include it in a NAS algorithm with a more sophisticated search strategy.

The GNNs that we proposed could also be used for more than performance estimation. We could include them also in the search strategy. A possible idea in this direction would be replacing the crossover operator in our evolutionary algorithm with a method based on GERNN that would take the two parents and automatically combine them into a successful offspring.

In the context of extending the usage of our GNNs, we could even explore applications in domains other than NAS. There is much data structured as graphs and some applications that process such data might benefit from GERNN and DAGRNN.

Summary Bibliography

[BND17] Diana Borza, Sergiu Cosmin Nistor, and Adrian Sergiu Darabant. Towards automatic skin tone classification in facial images. In *International Conference on Image Analysis and Processing*, pages 299–309. Springer, 2017.

[CVMG+14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[FSZ14] Ronja Foraita, Jacob Spallek, and Hajo Zeeb. *Directed Acyclic Graphs*, pages 1481–1517. Springer New York, New York, NY, 2014.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[KWM11] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg! In *Fifth International AAAI conference on weblogs and social media*, 2011.

[LBE15] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[NC22] Sergiu Cosmin Nistor and Gabriela Czibula. IntelliSwAS: Optimizing deep neural network architectures using a particle swarm-based approach. *Expert Systems with Applications*, 187:115945, 2022.

[NDB18] Sergiu Cosmin Nistor, Adrian Sergiu Darabant, and Diana Borza. Micro-expressions detection based on micro-motions dense optical flows. In *2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–7. IEEE, 2018.

[NID20] Sergiu Cosmin Nistor, Tudor Alexandru Ileni, and Adrian Sergiu Dărăbant. Automatic development of deep learning architectures for image segmentation. *Sustainability*, 12(22):9707, nov 2020.

[Nis20] Sergiu Cosmin Nistor. Multi-staged training of deep neural networks for micro-expression recognition. In *2020 IEEE 14th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 000029–000034. IEEE, 2020.

[Nis21] Sergiu Cosmin Nistor. An actor-critic approach to neural network architecture search for facial expressions recognition. In *20121 17th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, 2021.

[NMDB17] Sergiu Cosmin Nistor, Alexandra-Cristina Marina, Adrian Sergiu Darabant, and Diana Borza. Automatic gender recognition for "in the wild" facial images using convolutional neural networks. In *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 287–291. IEEE, 2017.

[NMM+21] Sergiu Cosmin Nistor, Mircea Moca, Darie Moldovan, Delia Beatrice Oprean, and Răzvan Liviu Nistor. Building a twitter sentiment analysis system with recurrent neural networks. *Sensors*, 21(7):2266, 2021.

[NMN21] Sergiu Cosmin Nistor, Mircea Moca, and Răzvan Liviu Nistor. Discovering novel memory cell designs for sentiment analysis on tweets. *Genetic Programming and Evolvable Machines*, 22(2):147–187, Jun 2021.

[PP10] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326, 2010.