

APPLICATION OF THE GROWING NEURAL GAS ALGORITHM TO OPTIMIZE COMPUTATIONAL RESOURCE IN NEURAL NETWORK ARCHITECTURE

N.V. Eliseeva, V.E. Petrov, Yu.V. Kaykova, I.V. Kaykova*

Moscow State University of Technology “STANKIN”, Moscow
Russia

* Corresponding Author, e-mail: kaykova.yulia@gmail.com

Abstract: The paper is devoted to the study of methods for improving the efficiency of using and training neural networks, given their complex architectures and a large number of parameters. The author has developed a method for optimizing the number of trained parameters using the growing neural gas algorithm. Approbation results show that the proposed method, combined with convolutional layers, shows a degradation in accuracy compared to traditional convolutional networks, but still outperforms random guessing. Changing the criterion for adding new nodes based on the current error reduced the number of neurons by half. The conclusions indicate that the developed method does not compete with existing solutions, but its further improvement may lead to optimization of computational resources in neural networks.

Key words: neural network optimization, machine learning automation, growing neural gas algorithm.

1. INTRODUCTION

Modern variants of neural networks can consist of tens or hundreds of thousands of neurons arranged in multiple layers and have hundreds of millions of parameters to train. Such sizes require a huge number of computations, for which multinational corporations use supercomputers.

At the same time, there is a growing worldwide need for products based on the use of neural networks: face recognition, unmanned transportation, anti-fraud systems, etc. Real-world applications require an increase in the quality of the results produced, which leads to an even greater increase in the number of parameters to be trained. This trend leads to the need to optimize the training processes of artificial neural networks not only because of time and financial, but also environmental problems [1].

In part, this problem is solved by optimizing training algorithms that minimize a given loss function. However, when training new models, there appears the problem of choosing the architecture – the number and order of layers, as well as the number of

neurons in them, and selecting hyperparameters – learning rate, size of batches, DropOut probability, etc. In most cases, the trial and error method is used for this purpose. The model is run with different parameters, and then researchers choose the ones with which it showed the best result. This approach leads to the need for a lot of calculations that are not involved in the final results.

The goal of this research is to find a way to minimize computation to find an architecture that gives the model sufficient accuracy with less computation. The latter part is relevant given that an excessively large number of trained parameters may not only fail to give a strong gain in accuracy, but also lead to its deterioration due to such features of gradient-based optimization methods as gradient explosion and gradient decay, as well as overtraining problems. At the same time, the lack of these parameters leads to undertraining of the model.

From the researcher's point of view, when creating a new model, it is difficult to predict the strength of the influence of certain elements of the network architecture on the final result. For example, researchers from Facebook AI Research managed [2] to achieve good accuracy from networks whose architecture was created using pseudo-random generators.

New architectures strive to fulfill business requirements – to be accurate, fast to run and learn, and not computationally intensive. They are created by researchers relying on mathematical apparatus and an intuitive notion of what methods can show the best results. Thus, the human factor plays a major role.

2. LITERATURE REVIEW

Currently, technologies for automating data pipelines in machine learning are being designed and developed. This direction has been named AutoML. Within its framework, tasks are set to automatically search for the best solutions at different stages of data processing: preprocessing, feature selection, model selection, hyperparameter selection, postprocessing, and model evaluation [3].

The purpose of developing such tools is to lower the threshold of entry into the field of machine learning and make the creation of own products in the field of artificial intelligence available to people who are not experts in it. From the point of view of this research, algorithms for searching neural network architecture are of interest.

The analysis of AutoML methods [4-9] has shown that they use different tools to automatically search the network architecture and its hyperparameters. The author's conclusions based on the analysis of the methods according to the criteria significant for the study of the possibility of creating minimum-sufficient neural network architectures and minimizing the computations required for training are presented in Table 1.

Some of the considered methods have penalty mechanisms for trained models, limiting their growth. Thus, it is possible to control the size of models and neural network architectures, in particular, trying to find those that will show good accuracy with fewer parameters. In addition, most tools are available under free licenses and their source codes are published on the Internet. This allows you to modify the algorithms they use, including adding similar limitations. At the same time, almost all of the above tools rely on an empirical approach to search. During search, they create multiple models, train

them on a full or partial dataset, evaluate them, and use the resulting information to modify them. This causes some of the intermediate models that have shown low accuracy to be “forgotten”. This method is similar to manual model tuning, usually conducted using trial and error. Although the accuracy data from these models is used to find better models, such computations are redundant and eliminating the need to train many unused models could greatly improve the performance of AutoML tools.

Table 1. AutoML tools comparison table

<i>Tool</i>	<i>The basis of the method</i>	<i>Creating minimalist models</i>	<i>Necessity of multiple retraining</i>
<i>WANN</i>	<i>Genetic algorithm</i>	<i>Yes</i>	<i>Yes</i>
<i>AutoWEKA</i>	<i>Bayesian optimization</i>	<i>No</i>	<i>Yes</i>
<i>TPOT</i>	<i>Bayesian optimization</i>	<i>No</i>	<i>Yes</i>
<i>H2O</i>	<i>Grid search</i>	<i>No</i>	<i>Yes</i>
<i>Auto PyTorch</i>	<i>Bayesian optimization</i>	<i>Yes</i>	<i>Partly</i>
<i>AutoKera</i>	<i>Bayesian optimization</i>	<i>Yes</i>	<i>Partly</i>
<i>Google AutoML</i>	<i>Neural networks</i>	<i>Yes</i>	<i>Yes</i>
<i>Growing neural gas</i>	<i>Topology repetition</i>	<i>Yes</i>	<i>No</i>

Based on the results of the analysis, the author chose the growing neural gas algorithm as the basis of the proposed method due to its advantage over other considered variants in the form of the absence of the need to train and evaluate intermediate models. It should be noted that the task of optimizing neural network architectures is posed in many applied studies in the field of machine learning [10, 11].

Despite the fact that the growing neural gas is a clustering algorithm, it can be used in neural networks to solve classification and clustering problems. For this purpose, we can resort to radial basis function networks (RBF networks). In this case, the graph constructed by the algorithm is used as neurons for the network layer. There are two use cases at this stage:

1. Use only the centers of clusters. This reduces the number of elements many times. The downside is that there is no guarantee that two complexly separable classes will end up in different clusters.

2. Use all the nodes of the graph. Then all the learned topology information of the input data is used, but more computation is required. When attempting to predict a value for a new input vector, the layer calculates the distance from a point in space of appropriate size, defined by the vector coordinates, to all elements used as neurons. This information is then used for prediction.

The author's proposed final scheme for working with the growing neural gas algorithm (Figure 1) using RBF networks consists of the following steps:

I. Training:

1. Prepare the data: fixed size vectors should be given as input to the algorithm.
2. Determine the parameters of the algorithm. According to research, growing neural gas is not very sensitive to small deviations of the coefficients used, and their recommended values have been identified. The main influence is the limitation of the maximum number of nodes and the choice of node addition and stopping criteria.

3. Apply the algorithm.
4. Train the weights of the subsequent layer(s).

II. Usage:

1. Obtain an input vector of the same size as the vectors used in training.
2. Calculate the distance from the point defined by it to the graph nodes or cluster centers.
3. Pass the obtained values to the next layer(s).

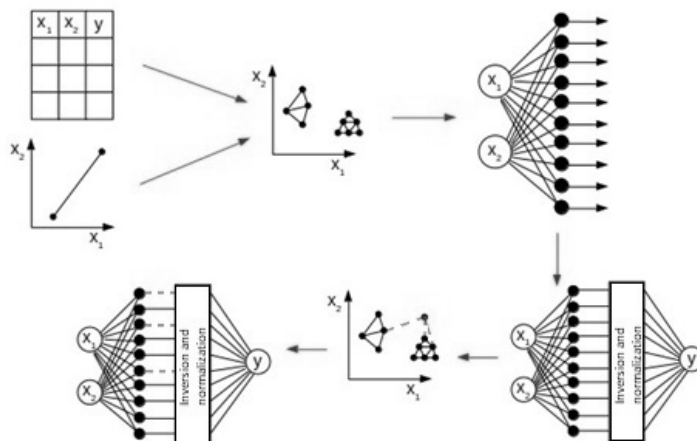


Figure 1. Scheme of using growing neural gas in a neural network

The output of the radial-basis layer yields non-normalized positive values, where 0 means that this neuron is located directly at the point specified at the input. Therefore, it makes sense to normalize and invert the values so that neurons close to the input give larger numbers than those distant from it.

3. DESCRIPTION OF THE PROBLEM AND PROPOSED SOLUTION

As part of the research for this paper, we decided to develop a method for applying the growing neural gas algorithm to the task of image classification. The task involves creating a model that determines the class label for an image depending on the object in the image. Classical full-link neural networks are able to solve this task to some extent, but encounter problems when the object appears rotated or reflected relative to how it looked in the training sample. To combat this, convolutional neural networks are typically used, which analyze small regions of an image and compose an image of the object in it from the resulting data. Therefore, it was decided to create a method that acts similarly to the image convolution operation. As a replacement for the convolution filter-core system, similar structures generated by a growing neural gas should serve.

In convolutional neural networks, applying a filter to an image region determines the degree of their similarity. In the case of a growing neural gas, an image region can be represented by a point in space with a number of dimensions equal to the number of pixels in it. The point will be at a position determined by the brightness of the colors

stored in the pixel data. The degree of similarity between two such points can be determined by the distance between them. The convolutional neural networks build an image of an object based on the data on the presence or absence of regions in the image that are similar or completely correspond to the specified filters. In this case, the trained parameters are the weights in the filters themselves.

Growing neural gas in combination with RBF-networks can build a similar system by first reconstructing the topology of the training set using the previously considered points, and then building the object image based on the remoteness of the input image regions from certain learned points. In this case, unlike convolutional networks, the growing neural gas does not require determining the number of filters to be used before training begins. Instead, having started training with two random “filters”, it increases their number according to the data set and the criterion of adding a new node.

Thus, it is assumed that the growing neural gas is able to build classifiers for images with as many trained parameters as needed for a given training dataset, without requiring to determine this number empirically. Given a dataset with images of width and height w pixels with three channels (according to the RGB color model), the luminance of each of which is expressed by the number $[0, 255]$, the assumed learning algorithm (Figures 2 and 3):

1. Normalize the brightness of the channels for each pixel: translate them from the interval $[0, 255]$ to $[0,1]$.
2. Initialize the growing neural gas algorithm with two points in 9-dimensional space.
3. For each image:
 - 3.1. Sliding window partition the image into overlapping regions of 3×3 pixels.
 - 3.2. For each region:
 - 3.2.1. Add up the channel brightness for each pixel.
 - 3.2.2. “Straighten” the resulting 3×3 matrix into a vector with dimensionality 9.
 - 3.2.3 Use the resulting vector as an example for training a growing neural gas.
4. Repeat step 3 until the training of the growing neural gas is completed. The result should be a graph consisting of points in 9-dimensional space, which are distributed according to the distribution of pixel group values in the input set.
5. Initialize the RBF layer with the nodes of the resulting graph. The 9-dimensional location vectors of the graph points of the growing neural gas are used as weights for the neurons of the RBF network. In this way, the RBF network obtains neurons that cover the topology of the training set. An input vector close to one of the points of the resulting growing neural gas graph will give a high signal at the output of the corresponding neuron after inverting.
6. Train the following layers of the neural network:
 - 6.1. For each image:
 - 6.1.1. Sliding window partition the image into overlapping regions of 3×3 pixels.
 - 6.1.2. For each region:
 - 6.1.2.1. Add up the channel luminance for each pixel.
 - 6.1.2.2. “Straighten” the resulting 3×3 matrix into a vector with dimensionality 9.
 - 6.1.2.3. Calculate the RBF layer output for the resulting vector.

6.1.2.4. Invert the output so that the neuron outputs take values from 0 to 1, where one corresponds to the smallest possible output of the RBF-network neuron, 0, and zero corresponds to the largest possible output of the RBF-network neuron.

6.1.2.5. Write the result in the corresponding row of the output tensor.

6.1.3. Transfer the tensor obtained at step 2 to subsequent layers for training.

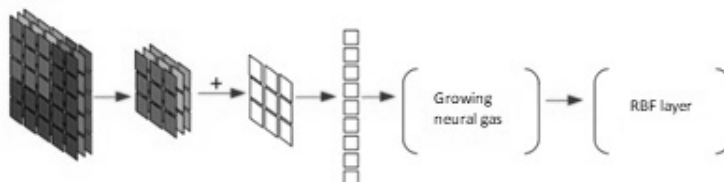


Figure 2. Scheme of the first stage of the learning algorithm

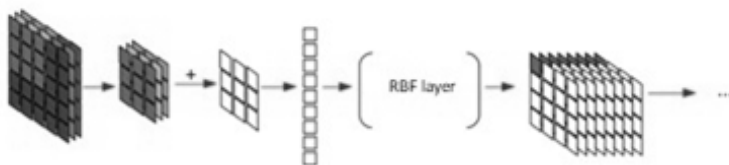


Figure 3. Scheme of the second stage of the learning algorithm

Thus, we obtain a layer that takes as input an image tensor of dimension $W \times H \times C$, where W and H are the width and height of the image, and C is the number of color channels (3 in the case of RGB images), the elements of which represent the brightness of a channel in a given pixel of the image. The output of the layer is a rank 3 tensor, whose first two dimensions are responsible for the groups of pixels of the input image obtained by the sliding window, and whose third dimension is responsible for the inverted output values of the RBF network neurons. On the axis of the third dimension, whose dimensionality is equal to the number of neurons in the RBF-network, the inverted “distance” of the group of pixels corresponding to the position in the first two dimensions from each of the nodes of the graph of the growing neural gas is marked.

The resulting method is partly similar to the operation of convolutional layers, but instead of trained filters, points in a 9-dimensional space are used, located in it with the help of the growing neural gas. In this case, the number of points is not set before training, but is built up in the process.

4. PRESENTATION OF RESULTS

For the experiment we used the well-known Cifar10 dataset, which contains training and test samples of 50,000 and 10,000 32×32 pixel color images and labels for 10 classes: airplane, car, bird, cat, deer, dog, frog, horse, ship, truck. The current world record for class prediction accuracy for this dataset is 99.7%.

The following are the results of Cifar10 image classification using growing neural gas and RBF networks using sliding window method (Table 2).

Table 2. Results of experiments with different approaches

<i>Model</i>	<i>Accuracy on test sample</i>
<i>3 layers created by the growing neural gas</i>	<i>10%</i>
<i>Model selected by the AutoKeras tool</i>	<i>81%</i>
<i>2 convolutional layers</i>	<i>65%</i>
<i>1 neural gas layer and 2 convolutional layers</i>	<i>52%</i>
<i>1 neural gas layer with modified node addition criterion and 2 convolutional ones</i>	<i>54%</i>

The following conclusions can be drawn from the results of this experiment:

1. Growing neural gas combined with convolutional layers showed a degradation in accuracy compared to the conventional convolutional network.
2. The obtained accuracy is higher than random guessing (10%), so the method retains some information and can be improved.
3. Changing the criterion of adding new nodes affects the accuracy.
4. Using a criterion for adding a new node that depends on the current error rate halved the number of neurons used.

Thus, the author has developed a method for optimizing the architecture of neural networks based on the growing neural gas algorithm for the possibility of creating minimalistic architectures and refusing to train a neural network by trial and error. The author's innovative approach consists in the idea of determining the optimal neural network architecture by approximating the training set topology with the growing neural gas algorithm.

5. CONCLUSION

In the process of the research, a method of searching the parameters of the neural network architecture was developed, which complicates the model as the requirement and does not use trial and error method in the process. But the obtained results showed that the described approach does not provide an accuracy advantage over traditional image classification tools. However, it allows to create minimalistic classification models by choosing the size of the resulting model based on an algorithmic decision, without requiring multiple restarts of training and evaluation of intermediate models, which distinguishes it from other AutoML tools.

The experiments showed that the method has potential: the accuracy of the hybrid model turned out to be noticeably higher than the accuracy of random guessing. This means that while processing the data, this method stores and possibly outputs new information about the input data. It can probably be used for real-world problems after improvement and correction of possible errors. In addition, the expected conclusion was obtained that changing the criterion of adding new nodes is able to positively influence the quality of the resulting model. Based on this, it can be said that further research of the method can lead to a new tool for automatic search of minimum-sufficient neural networks engaged in image classification.

REFERENCES

- [1] Strubell E., Ganesh A., McCallum A. Energy and policy considerations for deep learning in NLP. *arXiv preprint arXiv:1906.02243*, 2019, pp. 1-6.
- [2] Xie S., Kirillov A., Girshick R., He K. Exploring randomly wired neural networks for image recognition. *arXiv preprint arXiv: 1904.01569*, 2019, pp. 1-10.
- [3] Starodub A.V., Eliseeva N.V. Algorithm for analysing the activity of changes in the parameters of an artificial neural network. Selected *Scientific Proceedings of the Twentieth International Scientific and Practical Conference “Quality Management”*, Moscow, Russia, March 2021, pp. 314-318 (In Russian).
- [4] Kotthoff L. et al. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *Journal of Machine Learning Research*, vol.17, 2016, pp. 1-5.
- [5] Zimmer L., Lindauer M., Hutter F. Auto-PyTorch Tabular: Multi fidelity MetaLearning for efficient and robust AutoDL. *arXiv preprint arXiv: 2006.13799*, 2020, pp. 1-15.
- [6] Falkner S., Klein A., Hutter F. BOHB: Robust and efficient hyperparameter optimization at scale. *arXiv preprint arXiv:1807.01774*, 2018, pp. 1-19.
- [7] Jim H., Song Q., Hu X. Auto-Keras: An efficient neural architecture search system. *arXiv preprint arXiv:1806.10282*, 2019, pp. 1-11.
- [8] Riveiro M., Ventocilla E. Visual growing neural gas for exploratory data analysis. *10th International Conference on Information Visualization Theory and Applications*, Czechia, Prague, February 2019, pp. 58-71.
- [9] Lella L., Di Giorgio A., Dragoni A. Length of stay prediction and analysis through a growing neural gas model. *Proceedings of the 4th International Workshop on Artificial Intelligence and Assistive Medicine*, Pavia, Italy, June 2015, pp. 11-21.
- [10] Yoana Ivanova. Optimizing simulation models of an artificial neural network for digital recognition. *International Journal on Information Technologies and Security*, vol.13, no.4, 2021, pp. 59-70.
- [11] Shahapurkar A., Rodd S.F. Efficient feature aware machine learning model for detecting fraudulent transaction in streaming environment. *International Journal on Information Technologies and Security*, vol. 14, no. 3, 2022, pp. 3-14.

Information about the authors:

Natalia Vladimirovna Eliseeva – PhD, associate professor of MSUT “STANKIN”, areas of scientific research - information technologies, computer networks, system analysis, optimization

Valiriy Evgenivech Petrov – PhD, associate professor of MSUT “STANKIN”, areas of scientific research - information technologies, quality management, system analysis, lean manufacturing

Yulia Viktorovna Kaykova – postgraduate student of MSUT “STANKIN”, areas of scientific research - information technology, systems analysis, neural networks, optimization

Irina Viktorovna Kaykova – postgraduate student of MSUT “STANKIN”, areas of scientific research - information technologies, system analysis, quality management

Manuscript received on 14 April 2025