POLITEHNICA UNIVERSITY OF BUCHAREST

Doctoral School of Electronics, Telecommunications and Information Technology

Decision No. ____ from DD-MM-YYYY

Ph.D. THESIS SUMMARY

Ing. Paul Liviu Aurel Diaconescu

CONTRIBUȚII LA OPTIMIZAREA ȘI APLICAȚIILE MODELULUI DEEP CONVOLUTIONAL NEURAL NETWORKS

CONTRIBUTIONS TO THE OPTIMIZATION AND APPLICATIONS OF DEEP CONVOLUTIONAL NEURAL NETWORK MODEL

BUCUREŞTI 2021

Content

List of tables

List of figures

Chapter 1

Introduction

1.1 Presentation of the field of the doctoral thesis

1.2 Scope of the doctoral thesis

1.3 Content of the doctoral thesis

The thesis is split in eight chapters in which DCNN systems are presented with image processing purposes.

Chapter 1 is an introductory chapter where the field, the scope and the content of doctoral thesis are presented.

Chapter 2 is a chapter related to Artificial Intelligence in general and Deep Learning in special, explaining essential components, their mechanisms and their interaction.

Chapter 3 presents a new method for the detection of drunkenness state from thermal images in infrared spectrum with the help of an ensemble of two DCNNs.

Chapter 4 presents the use of a DCNN with an original architecture for hyperspectral image classification.

Chapter 5 introduces the generation of artificial pixels created with Generative Adversarial Networks used for an improved classification of hyperspectral pixels.

Chapter 6 presents the coordination of DCNN training with optimization algorithms for the classification of credit requests

Chapter 7 introduces transfer learning in case of object detection in the field of urban auto traffic.

Chapter 8 presents the results, the conclusions and contributions of DCNN developments plus the future perspectives for the applications described in chapters 3 to 7.

Chapter 2

Artificial Intelligence and Machine Learning

The artificial intelligence is the intelligence demonstrated by machines and robots, an intelligence created by humans that includes the automation of human activities as planification, learning, rationing, representation, perception, and with some constraints even social intelligence and creativity.

Machine learning is the artificial intelligence field the covers the learning component and provides to researchers a class of methods including supervised and unsupervised methods.

2.1 Deep Learning – Essential concepts

Deep Learning is a supervised learning method of machine learning, inspired from human brain functionality. Similar with the way the brain learns from examples, neural networks can learn from the exposure to a high number of training elements of which characteristics are assimilated. The training elements can be images, meaningful text or sounds. To allow the learning of some complex objects or structures, deep learning methods are using a system built from layers of neurons with dynamic weights, updating to match the relevant info received. Due to the large number of layers, we say about these systems that are deep.

The applications of this methods can be the autonomous driving, elements classification, language translation, virtual assistants, disease detection, face recognition, multimedia content generation, the analysis and prognosis of events, fraud detection and even the replacement of human players in complex logical games as chess or go. These techniques are used in all these applications due to their very good results, which in some cases are even better than human results. These results are a good basis for future developments in the direction of an general artificial intelligence.

A major advantage of deep learning versus other machine learning methods is the inclusion of the training element characteristics extraction, a step that usually with other methods has to be performed separately.

2.1.1 Convolutional Neural Networks - CNN

2.1.2 Convolutional layer

2.1.3 Activation layer (Relu, Tahn, Sinusoidal)

2.1.4 Pooling layer

2.1.5 DropOut Layer

2.2 DCNN training

The training of a DCNN is a complex development, including the following components:

- Finding a dataset (for the network training) that is suitable for the proposed objective and split of its elements in training elements, training validation elements and test elements
- The creation of a proper architecture considering the objective and dataset
- Network configuration hyperparameters selection
- Weights initialization
- Selection of a proper training loss and validation loss functions
- Selection of a training optimization algorithm, that drives the way the DCNN weights are updated depending on the last epoch loss function result
- Network productization. Involves the network installation on a hardware system

Considering the complexity of each of these steps, the training can take days or even weeks or months. The training is an iterative process, good practices being established for automation of some parts of the flow.

2.2.2 Back-propagation

2.2.3 Gradient Descent and learning rate. Early stopping

2.2.4 Characteristics of training algorithms

2.2.5 Overfitting

2.2.6 Training Validation

2.2.7 Training Loss

2.2.7 Training acceleration methods

2.3 CNN applications

2.3.1 Classification

One of deep learning application is the elements classification. The classification is a process through which a DCNN learns from a set of training elements and is able to classify test elements. The classification can be unsupervised, meaning the network classify the elements using common characteristics in a number of classes defined by the researcher or the classification can be supervised, test elements being classified with the help of human feedback during the training phase.

2.3.2 Transfer learning

2.3.3 Object detection

2.4 Selection of well-known DCNN architectures

2.5 Other types of Deep Learning

2.6 Frameworks

2.7 Hardware performance optimization

Chapter 3

An ensemble of deep convolutional neural networks for drunkenness detection using thermal infrared imagery

3.1 Introduction

This paper proposes an original method for drunkenness detection using an Ensemble of two Deep Convolutional Neural Networks (DCNNs) for processing of the thermal infrared facial imagery of the subjects to be tested. We have chosen the variant of subject independent drunkenness recognition implying the procedure of building training datasets for each of the J subjects, using the images of the other (J-1) ones different of the considered subject. The main objective of our research is to obtain a correct detection rate of more than 90% for both of the drunk and sober states. The model is evaluated using the dataset of 400 thermal infrared images in infrared spectrum.

3.2 Proposed model based on the couple of concurrent DCNNs

Convolutional Neural Networks are used in various of application due to their capacity to extract features and use these features to recognize and classify objects. Variations of CNNs include layers with different primitive functions: convolution, dropout, activation (as rectifier), fully-connected, soft-max, classification and others.

This paper proposes to process the input facial thermal infrared image for inebriation detection using a neural system composed by an ensemble of two DCNNs. The two DCNN modules of the ensemble have 12 layers and respectively, 10 layers. The difference is made by the third convolutional layer (with an additional dropout Layer) used in only one of the networks.

Tab. 3.1 *Architecture of the DCNN modules*

Layer number	**DCNN-1**	**DCNN-2**
1	imageInputLayer	imageInputLayer
2	convolution2dLayer	convolution2dLayer
3	dropoutLayer	dropoutLayer
4	convolution2dLayer	convolution2dLayer
5	dropoutLayer	dropoutLayer
6	convolution2dLayer	fullyConnectedLayer
7	dropoutLayer	reluLayer
8	fullyConnectedLayer	fullyConnectedLayer
9	reluLayer	softmaxLayer
10	fullyConnectedLayer	classificationLayer
11	softmaxLayer	
12	classificationLayer	

The two DCNNs have been trained separately, by using only some common parameters, the rest of parameters being different, as shown in Table I. Consequently, there are two independent DCNNs with independent decisions. The final decision mechanism works like this:

(a) When both DCNNs lead to the same decision (drunk or sober), we consider that common result as final

(b) When the two DCNNs lead to different decisions, higher confidence result is selected as a final decision.

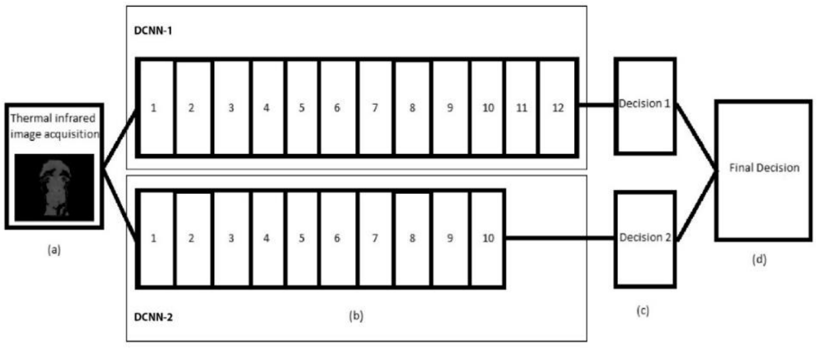

Fig. 3.1 *Fig. 1. An ensemble of two DCNNs architecture for subject independent drunkenness detection: (a) loading of the thermal infrared image; (b) the two DCNNs with 12 and 10 layers; (c) detection results for each of the two DCNNs; (d) Final decision result given by fusion of the component decisions.*

3.2 Experiments

3.2.1 Thermal image dataset

3.2.2 Training and parameter setup

3.3 Results

The results of subject independent drunkenness detection are given in Table III, both as individual performance for each subject and as an overall performance. One can deduce a minimum correct detection rate of 90% for subject number 7, a maximum rate of 97.5% for subjects numbers 2, 3, 6, 8 and 10, as well as an overall detection rate of 95.75%.

Tab. 3.3 *Subject independent drunkenness detection rate*

Index of the test subject for drunkenness detection with the neural couple of CNNs trained on the other 9 subjects	**DCNN 1**	**DCNN 2**	**Ensemble of two DCNNs**
1	90.00%	95.00%	95.00%
2	97.50%	95.00%	97.50%
3	95.00%	92.50%	97.50%
4	95.00%	95.00%	95.00%
5	92.50%	90.00%	95.00%
6	97.50%	95.00%	97.50%
7	90.00%	87.50%	90.00%
8	92.50%	90.00%	97.50%
9	92.50%	92.50%	95.00%
10	90.00%	95.00%	97.50%
Average Correct Detection Rate	93.25%	92.75%	95.75%
Standard Deviation	3.89%	3.69%	3.18%

Chapter 4

A highly configurable deep learning architecture for hyperspectral image classification

4.1 Introduction

Hyperspectral images are pictures obtained through collection and processing of information from electromagnetic spectrum. Each image pixel is a collection of responses received from a point in different wave lengths. With tens or hundreds of wave lengths and high-resolution pictures, the task of pixel classification cannot be done by humans. Machine learning methods can contain this complexity and have many subfields that are able to support such classifications. Between machine learning methods, deep learning methods are very much used these days while being constantly improved. One of their advantages is the fact that hyperspectral images do not need augmentation before the deep learning pattern understanding and classification.

4.2 Proposed method

Pixel classification can be done based on the pixel characteristics alone or also based on characteristics of the pixel and pixels around it. These options are called spectral correlation and spectral-spatial correlation. The number of neighbor pixels is typically chosen considering the pixels around each original pixel: a first layer has 8 neighbor pixels, two layers have 24 pixels, three layers have 48 pixels. Other works had already shown a better classification is achieved in case of using neighbors. Also, it was found that adding more neighbors after a certain number of neighbors, does not help the classification score.

The network architecture is based on parallel flows of convolutional layers and is presented in Fig. 4.2.

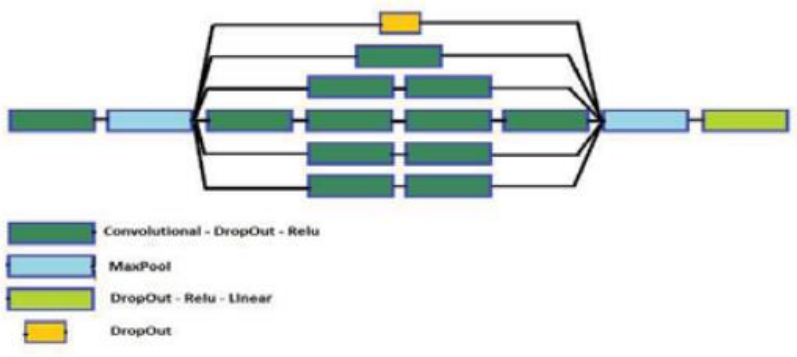

Fig. 4.2 *Network architecture*

The training mode is determined by the optimization algorithm and loss function. Our most significant candidates for optimization algorithms were Stochastic Gradient Descent, Adaptive Moment Estimation and Adagrad, while the loss functions considered were Cross Entropy Loss, Negative Log Likelihood Loss, Connectionist Temporal Classification and Multi Margin Loss.

The optimization algorithm that worked best with Pavia University dataset was the Adaptive Moment Estimation. This algorithm has an adaptive learning rate.

Multi Margin Loss and Cross Entropy Loss functions had much better results than Negative Log Likelihood Loss and Connectionist Temporal Classification functions. They worked much better than the others with any of the above optimization algorithms. We concluded those two loss functions were better adapted to the type of dataset used. Finally, MultiMarginLoss loss function has been kept for the main training due to a faster run time and Cross Entropy Loss has been used for pre-training.

4.3 Experimental results

4.3.1 Dataset

In this paper, the Pavia University dataset has been used.

4.3.2 Results

Tab. 4.4 *Results of deep learning classification accuracy for 200 training elements*

#	Vecinity	Epochs	Optimization algorithm	Loss function	Correct classification
1	0	123	ADAM	CELoss	78.6%
2	8	80	ADAM	CELoss	82.4%
3	24	1014	ADAM	CELoss	85.1%
4	24	815	ADAM	MMLoss	89.8%
5	24	1262	SGD	MMLoss	84.8%
6	48	2953	ADAM	MMLoss	83.6%

The classification score is the average of class classifications. Two classes have been classified 100% correctly (Painted Metal Sheets and Shadows) and the smallest score has been obtained for Meadows (83%). The results from Table 4 have been obtained in PyTorch framework. In Matlab the classification results for the almost the same network are between 1%-2% lower compared with PyTorch.

We have used as benchmark the Support Vector Machine (SVM). Table 5 shows the accuracy results of SVM classification based on different number of training pixels and different number of neighbors.

Tab 4.5 *Accuracy results of SVM classification*

Number/Percentage of Training Pixels	**8 Neighbors**	**24 Neighbors**
4%	61.1%	61.8%
20%	77.5%	77.9%
200	51.3%	51.7%
4% + 200	76.0%	76.9%

The classification accuracy of deep learning method and SVM have been compared for the same number of training pixels and the same number of neighbors in Tab.4.6

Tab. 4.6 *Deep learning vs SVM classification accuracy*

Machine Learning Method	**Number of Training Pixels**	**Neighbors**	**Correct Classification Score**
SVM	200/class	24	51.7%
Deep Learning	200/class	24	89.8%

Chapter 5

CNN hyperspectral image classification using training sample augmentation with Generative Adversarial Networks

5.1 Introduction

5.1.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) are used for more than 1,000 applications, as human face generation, face aging, text to image translation and video prediction. New elements are generated from an existing distribution of models, keeping model features. The generation is done by an ensemble of two networks, one of them called “the Generator” and the other called “the Discriminator”. These networks are unsupervised networks and the accuracy of their work is evaluated by the number of errors they make.

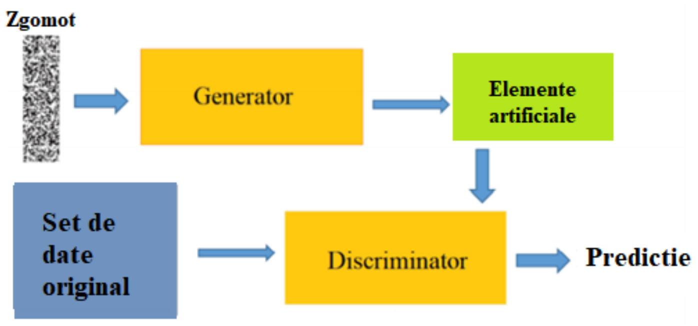

Fig. 5.1 The architecture of Generative Adversarial Networks

5.1.2 Hyperspectral Image Classification

As an aim of this paper, the authors have tried to increase the pixel classification accuracy using different state of the art deep learning training methods. The technique that worked best was that based on using more pixels for training, by the addition of those obtained with GANs. Considering the pixels cannot be directly read and understood by humans due to their complexity, generating new high quality pixels has become a challenging task. By high quality pixels we understand pixels that have the same features as the original pixels, leading to their correct classification.

5.2 Proposed Method

For the GAN model we have used a custom architecture shown in Fig. 5.3.

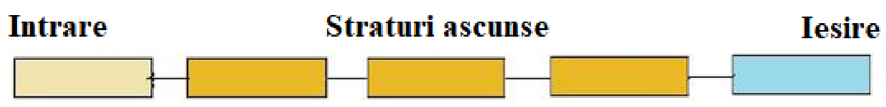

Fig. 5.3 *Generator architecture and Discriminator architecture*

The Generator includes five transposed convolutional layers, the first four of them (in beige and orange colors) being each followed by a batch normalization layer and a ReLU layer, and the last convolutional layers being followed only by a Tanh layer at the end.

The Discriminator contains five classical convolutional layers, the first four of them (in beige and orange colors) being each followed by a batch normalization layer and a LeakyRelu layer. The last convolutional layer is followed by a Sigmoid layer.

The DCNN architecture includes parallel flows: different types of convolutional layers and Gated Recurrent Units (GRU) layers. All the convolutional (or GRU) layers are followed by normalization layers, dropout layers and LeakyRelu layers. The last layers are two linear layers and a LogSoftmax layer. The top five flows are using convolutional layers while the bottom five flows are using GRU layers. The colors have been used in Fig. 4 as following: Beige: convolutional layer, dropout layer, activation layer, normalization layer; Orange three times: convolutional layer, dropout layer, activation layer, normalization layer; Orange two times: convolutional layer, dropout layer, activation layer, normalization layer; Green: two times: transpose convolutional layer, dropout layer, activation layer, normalization layer; Cold Green: transpose convolutional layer, dropout layer, activation layer, normalization layer; Blue: last layer: linear layer, dropout layer, activation layer, linear layer, dropout layer and the last activation layer

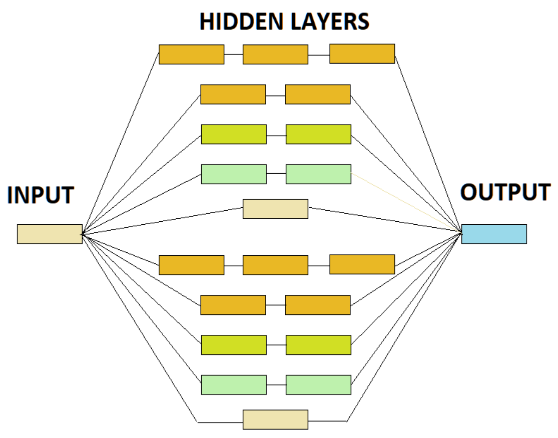

Fig. 5.4 *DCNN architecture for classification*

5.3 Experimental results

Tab. 5.1 *Pixels allocation variants for each class*

Variant of pixels allocation	**Original Pixels for Training**	**Classical augmented pixels for training**	**GANs augmented pixels for training**	**Validation pixels**	**Test pixels**
1	300	0	0	0	42,476
2	200	0	0	100	42,476
3	300	500	0	0	41,976
4	200	500	0	100	41,976
5	300	0	500	0	42,476
6	200	0	500	100	42,476

Combining augmentation with training validation, we have obtained six types of training and the best accuracy results are shown in Tab. 5.4.

Tab. 5.4 *Classification results based on depending on transformations and validation*

Variant of pixels allocation	**Pixel Allocation for Augmentation and Validation**	**Classification accuracy**
1	No Augmentation / No Validation	87.43%
2	No Augmentation / Validation	92.94%
3	Classical Augmentation / No Validation	84.14%
4	Classical Augmentation / Validation	86.76%
5	GAN Augmentation / No Validation	89.79%
6	GAN Augmentation / Validation	95.32%

Chapter 6

The classification of credit requests using DCNN model optimization

6.1 Introduction

6.1.1 Financial credit risk

This paper addresses the evaluation of credit risk associated to lending to private people and it doesn’t cover the risk associated with lending to economic entities. For the purpose of this paper, the macroeconomic context is considered stable.

6.1.2 DCNN solutions for credit risk predictions

This work aims beyond classical human try and error techniques for creating a DLNN architecture and using intuition for selecting parameters. The architecture and hyperparameters have to be the result of algorithm selection, depending on the dataset and project objective.

6.2 Proposed Method

6.2.1 Framework

6.2.2 Evaluation criteria and credit cost

6.2.3 Optimization algorithms

For our application, the following optimization algorithms had been used with the scope of minimizing the objective function:

Gradient Boosted Regression Trees (GBRT) [18] is starting an initial set of simulations based on randomly chosen hyperparameters from a given range, then it tries to find a better combination of hyperparameters by building trees from the results, then adding new branches on the most interesting nodes.

Decision Trees (DT) is characterized by the fact that the space of possible values is divided into distinct and non-overlapping regions based on hyperparameters used by the optimization function. Every observation is used to build a mean of responses for the corresponding region. This method doesn’t need the initial random runs, all runs are part of the algorithm.

Uniform Sampling (US) within the given bounds of the hyper parameters. The space was split in a number of sub-spaces given by the number of algorithm iterations.

Random Search (RS) [19] had been shown having better efficiency than Uniform Sampling

Gradient Descent (GD) optimization algorithms from Pytorch libraries as ADAM, Adagrad, SGD, Adadelta and others. These algorithms are starting with a single random point in the n-dimensional space of all solutions and based on the size of the error between the result and the objective, are moving the selection in different directions with a speed/ distance given by the algorithm used.

Bayesian Optimization (BO) is one of the most used optimization algorithms for deep learning. The a posteriori probability of a model based on existing evidence is proportional with the likelihood of that evidence multiplied by a priori probability. The main difference of this optimization model compared with the previous two ones is that all exercised points matter, the final solution depends on all of them. This optimization algorithm works as following:

- observation of a number of simulations (a priori) which are considered existing data
- prediction of a function based on existing data with a degree of uncertainty
- continuous update of the function (combining the estimated function with new evidence) by adding new real results which reduce the uncertainty.

6.2.4 DCNN architecture

The optimum architecture for our scope can be seen in Fig. 6.2.

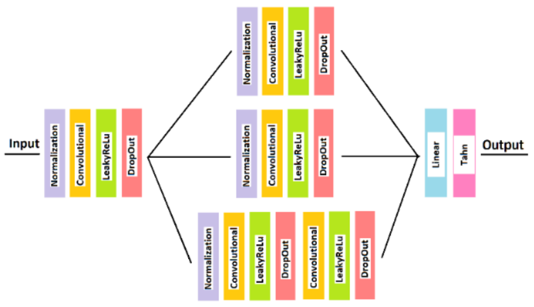

Fig. 6.2 DCNN architecture

The layers used in the optimum DLNN architecture are shown in Tab 6.2.

Tab. 6.2 *DCNN architecture*

Layer number	**Layer**
1 – 4	Batch Normalization - Conv - LeakyRelu - DropOut
5 – 8	Batch Normalization - Conv - LeakyRelu - DropOut
9 – 12	Batch Normalization - Conv - LeakyRelu - DropOut
13 – 19	Batch Normalization - Conv - LeakyRelu – DropOut - Conv - LeakyRelu - DropOut
20 – 21	Linear – Tahn

6.3 Experiments and results

6.3.1 Dataset

This paper is using the German Credit dataset [20], containing 1,000 records with 20 attributes each. Each record represents the responses offered by a credit requester and it is classified as good or bad for credit. There are 700 records classified as good and 300 records classified as bad.

6.3.2 Performance and optimum hyperparameters

6.3.3 Experimental results

Six optimization algorithms have been used to find the optimum hyperparameters for the DLNN training. In general, the approach was to start with large ranges for each parameter and after several rounds of observations of simulations done with each algorithm, the ranges have been reduced. For example, for learning rate parameter, the search started in the range of (10-8, 0.9) and it has been reduced to (10-3, 0.1) after observations showed that learning did not happen for the lowest part of the initial range.

As ranges have been reduced due to good results from an algorithm, the other algorithms benefited from the reduction of the ranges as well. For this reason, we can say that using more than one optimization algorithms can help with the speed up of the overall search. Some of the algorithms proved better with initial exploration (Random Search, Uniform Sampling) while others proved better at exploitation (Gradient Boosted Decision Trees and Gradient Descent Optimization). The algorithms better at explorations have shown also a better stability in the response to different initial network values, considering similar strategies for generating these values as presented earlier in this paper.

Considering the large number of the optimized hyperparameters (fourteen), we have deduced the minimum number of runs of any algorithm in a given simulation should be at least around 1,000 in order to obtain results with less than 25% accuracy difference from the best result.

The operations of the optimization algorithm have been completely run on CPU, while the DLNN training has been run on GPUs. Some of the algorithms required sequences of operations taking place before the hyperparameters selection at each run and this increased the simulation time but overall, this overhead has been small (less than 1% of time spent in a simulation).

Random Search optimization algorithm has been the easiest algorithm to use. It just received the list of ranges for searching parameters and randomly tried different combinations of parameters. It was fast for obtaining acceptable results and it is recommended to be used during early development of the DLNNs.

Uniform Sampling optimization algorithm proved to be very dependent on the distance between two samples, due to the multiple local minimums and considering the large number of parameters, it showed a very low speed for converging to good results. Even after finding good results, it did not take them into account (as expected) and went into long intervals of non-interesting results. As the Random Search optimization algorithm, Uniform Sampling algorithm can be a choice for the initial stages of DLNN development.

Gradient Boosted Regression Trees optimization algorithm presented a relative fast convergence to acceptable results then, it took a long time to obtain very good results. A computation overhead had been observed while the algorithm calculated the new leaves. The algorithm is deciding which leaves are more interesting for growing and we have remarked a dependence with the quality of the initial random runs. We have remarked better results for exploitation and less good results if exploration was needed, as in case of unfortunate initial weights.

Decision Trees optimization algorithm has been quite slow to converge to good results in most of the simulations. This algorithm works better for finding solutions in more plane n-dimensional spaces, where it can theoretically be faster than other algorithms.

Gradient Descent optimization algorithm has good results when used on already reduced parameters ranges. However, with the initial, larger ranges, the results were unsatisfactory. We found would be a good decision to run it after another algorithm. The algorithm instance we have used missed a random set of initial random runs, to automatically start from a convenient set of parameters. This enhancement would probably make it much better overall.

The best results were obtained using a Bayesian Optimization algorithm. We have remarked consistent results with both larger and smaller parameter ranges and finally we have obtained the best results with it. This optimization algorithm is quite fast in converging to good credit cost results compared with the other algorithms, considering the number of runs. It brings a computational overhead produced when it calculates the new parameters values and this overhead increases as the algorithm has more previous runs to take into consideration. One significant advantage of BO algorithm was the fact is highly configurable in terms of exploration vs exploitation, that makes it a good choice at different phases of searching for a DLNN hyperparameters.

Tab. 6.6 *Classification precision results based on DCNN for 220 training elements*

#	**Epochs**	**Optimization**	**Initial LR**	**DL Optimization/ DL Loss Type**	**Credit Cost**	**MAR**
1	86	GDO	0.001	ADAM / CEL	137	7%
2	20	DT	0.057	Adagrad / NLL	170	11%
3	85	BO	0.170	AdamW / MMLoss	**103**	**3%**
4	18	GBRT	0.050	Adam / NLL	152	10%
5	142	RS	0.756	AdamW / MMLoss	167	11%
6	267	US	0.065	AdamW / MMLoss	155	9%

Chapter 7

DCNN techniques used for automated driving in urban areas

7.1 Introduction

Building on previous research directed to machine learning for pedestrian detection and neural networks for autonomous navigation, in this paper we are adding an additional contribution to the automotive field developments by introducing a novel technique of mapping object detection technologies to an automotive environment.

7.2 Transfer learning

7.3 YOLOv5 DCNN

7.4 Proposed method

7.4.1 Transfer learning method

7.4.2 Evaluation criteria

The object detection mechanism is based on two components: object identification for a given image and prediction of its coordinates. They can be measured using the indices of Recall (how well you find all the positives) and Precision (how accurate are your coordinates predictions).

For the purpose of measuring the DLNN performance of object detection, we have used the mean Average Precision (mAP@0.5) as a main criterion as this includes the components of both Recall and Precision. The index mAP@0.5 is the mean Average Precision (calculated as the area under the Precision vs Recall curve) at IoU > 0.50, where IoU is Intersection over Union (calculated as the intersection between the ground-truth bounding box of an object and the corresponding predicted bounding box). A value for IoU higher than 0.5 is agreed to correspond to a good detection of an object. However, the index mAP@0.5 is calculated for all objects and this is a measure of the DLNN capabilities, not a measure of how well each object has been detected.

7.4.3 Evolving parameters with a genetic algorithm

In the search for the proper hyperparameters values for BDD100K dataset, we have used a default mechanism included in YOLOv5 release, called “Hyperparameter evolution” and we have done customizations over this mechanism.

Hyperparametric evolution lets the researcher to define a number of tries and a number of epochs per try. Then, after each training iteration of the defined number of epochs, the Genetic Algorithm updates the hyperparameters and another run is started until the defined number of tries is reached.

7.4.4 Software architecture

YOLOv5s and YOLOv5s6 (the smallest variants, release 5 [18]) DLNNs are written in Python and run with Pytorch framework. We have done our simulations under Ubuntu Linux 20.04, using Python 3.9 (beta) compiler and NVIDIA CUDA 11.2.

7.5 Experiments and results

7.5.1 Datasets

Common Objects in Context (COCO) 2017 is the default dataset used for the development of YOLOv5 DLNN. The dataset includes 80 types of objects in 1.5 million object instances.

We have used transfer learning to recognize objects from BDD100K dataset. This is a Large-scale Diverse Driving Video Database from Berkeley, including 100,000 images with good context diversity including multiple cities, multiple weathers, multiple times of day (including low light, low visibility) multiple scenes types (including overlapped objects).

7.5.2 Transformation of annotations

7.5.3 Experimental results

After trying a set of techniques for the most efficient transfer learning, the proper technique for our case has been the following:

As the objective has been having a high mAP@0.5, we have tried to modify the YOLOv5 fitness function with different values. Against intuition, the best results haven't been achieved when having mAP@0.5 very high and the other objectives very low, instead the best results have been obtained when finding an equilibrium between them.

[Precision, Recall, mAP@0.5, mAP@0.5:0.95] = [0.2, 0.2, 0.4, 0.4]

A trained (trained with COCO) version of YOLOv5 DLNN has been used for a large number of training iterations (training with BDD100K) with the only scope of finding the optimum hyperparameters for training the network. This has been achieved by selecting the hyperparameters that allowed the fastest progression of mAP@0.5 after 6 epochs, then using the winner set of hyperparameters for more evolving in simulations with 50 epochs. After finding the most efficient hyperparameters in the 50 epochs contest, we have run a training iteration with enough epochs to obtain a high mAP@0.5.

Examples of images with detected objects marked, are presented in Tab 7.3

Tab. 7.3 *Results*

Evaluation Indicator	mAP@0.5	Training time	Average detection time per image
Reference: **Original YOLOv5s,** trained with COCO, image size 640 / **Original YOLOv5s6**, tested with COCO, image size 1280	55.4 / 61.9	N/A	0.010s / 0.024s
YOLOv5s6 trained with BDD100K for 100 epochs, image size 1280, tested with BDD100K	62.6	65.5 hours	0.024s
YOLOv5s trained with BDD100K for 100 epochs, image size 640, tested with BDD100K	59.6	18.1 hours	0.010s
YOLOv5s trained with BDD100K for 100 epochs, image size 1280, tested with BDD100K	68.7	65.2 hours	0.024s
YOLOv5s trained with BDD100K for 100 epochs with 24 layers frozen, image size 640, tested with BDD100K	19.6	17.3 hours	0.010s
YOLOv5s trained with BDD100K for 100 epochs, using ADAM instead of SGD optimization algorithm, image size 640, tested with BDD100K	30.2	17.5 hours	0.010s
Results from other papers [19] for object detection on BDD100K dataset	45.7	N/A	N/A
Results from other papers [20] for object detection with YOLO (from scratch, without transfer learning) on BDD100K dataset	18.6	N/A	N/A

We have obtained better object detection results for BDD100K dataset compared with the best results obtained by YOLOv5 with COCO dataset. The most probable explanation is that the number of object classes is lower in case of BDD100K dataset than in case of COCO. However, for a specialized application as autonomous driving in this case, the results can be considered as very good.

Chapter 8

Conclusions

8.1 Obtained results

8.2 Original contributions

8.2.1 Contributions to the use of an ensemble of deep convolutional neural networks for drunkenness detection using thermal infrared imagery

The contribution related use of an ensemble of deep convolutional neural networks for drunkenness detection using thermal infrared imagery are the following:

- better drunkenness detection performance: 2.5% more compared with the case of using a single DCNN

- the method allows both feature selection and classification, thus avoiding the necessity to use old cumbersome and time-consuming techniques for feature selection.

These contributions have been presented in the paper:

V. E. Neagoe, **P. Diaconescu**, "An Ensemble of Deep Convolutional Neural Networks for Drunkenness Detection Using Thermal Infrared Facial Imagery," Proc. 13th International Conference on Communications (COMM2020), Bucharest, Romania, Electr Netowork, June 19-20, 2020, pp. 147-150, WOS:000612723900026

8.2.2 Contributions to the use of a highly configurable deep learning architecture for hyperspectral image classification

In Chapter 4 we have presented a method to adapt an architecture and hyperparameters to a dataset and to a chosen objective.

We have compared 2 frameworks (Matlab and Pytorch) showing the impact of changing the training optimization algorithm, impact of training loss, impact of considering the vicinities and of number of epochs over the percent of correct classification for hyperspectral pixels. We have achieved a very good classification result as well.

These contributions have been presented in the paper:

P. Diaconescu, V. E. Neagoe, "A Higly Configurable Deep Learning Architecture for Hyperspectral Image Classification," Proc. IEEE *13th International Symposium on Applied Computational Intelligence And Informatics (SACI 2019),* Timisoara, ROMANIA, May 29-31, 2019, pp.197-200, WOS:000610436600035

8.2.3 Contributions to a CNN hyperspectral image classification using training sample augmentation with Generative Adversarial Networks

The contribution presented in chapter 5 consists in use Generative Adversarial Networks for the enhancement of DCNN classification results through generation of artificial pixels creation. GAN implementation is original and is adapted to the type of elements used (spectral pixels with many bands and different vicinities). The contribution is significant also because the original pixels acquisition is very expensive and needs a long and complex preparation.

Another contribution is the use of Bayes optimization for automated selection of hyperparameters.

These contributions have been presented in the paper:

V. E. Neagoe, **P. Diaconescu**, "CNN Hyperspectral Image Classification Using Training Sample Augmentation with Generative Adversarial Networks," Proc. 13th International Conference on Communications (COMM2020), Bucharest, Romania, Electr Network, June 19-20, 2020, pp.515-519, WOS:000612723900091

8.2.4 Contributions to the classification of credit requests using DCNN model optimization

The use of optimization algorithms in Chapter 6 brings significant contributions to the easiness of utilization by financial institutions that can obtain better results in a much cheaper way.

We have found an optimum selection of:

- Any of 6 types of optimization algorithms for finding optimum architectures and hyperparameters
- Any of 9 types of gradient optimization algorithms for network training
- Any of 8 activation layers
- Any of 4 loss functions
- Early stopping of training
- Mixed precision training

We have shown that financial institutions can use modern methods integrated together for classification of the credit risk.

These contributions have been presented in the paper:

P. Diaconescu, V. E. Neagoe, "Credit Scoring Using Deep Learning Driven by Optimization Algorithms," *Proc. 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI-2020)*, Bucharest, Romania, June 25-27, 2020, pp.1-6, WOS:000627393500021

8.2.5 Contributions to the DCNN techniques used for automated driving in urban areas

In chapter 7 we have shown that mapping a world class DCNN and a specialized dataset, one can obtain very good results very fast if proper transfer learning is done.

Depending on selected objective, the update of the YOLOv5 fitness function could have a positive effect. Our object detection score is the highest score obtained for object detection performance with the Berkeley Diverse Driving Video Database.

The code for automated transformation of image annotations is Open- Source and can be reused [21], reducing the cost of a similar approach.

These contributions have been presented in the paper:

[5] **P. Diaconescu**, V. Neagoe, „A Deep Learning Approach to Autonomous Driving in Urban Environment", accepted by Scientific Bulletin of the Politehnica University of Bucharest, Series C, Electrical Engineering and Computer Science, to be indexed in Web of Science

8.3 List of original contributions

[1] **P. Diaconescu**, V. E. Neagoe, "A Higly Configurable Deep Learning Architecture for Hyperspectral Image Classification," Proc. IEEE *13th International Symposium on Applied Computational Intelligence And Informatics (SACI 2019),* Timisoara, ROMANIA, May 29-31, 2019, pp.197-200, WOS:000610436600035

[2] V. E. Neagoe, **P. Diaconescu**, "An Ensemble of Deep Convolutional Neural Networks for Drunkenness Detection Using Thermal Infrared Facial Imagery," Proc. 13th International Conference on Communications (COMM2020), Bucharest, Romania, Electr Netowork, June 19-20, 2020, pp. 147-150, WOS:000612723900026

[3] V. E. Neagoe, **P. Diaconescu**, "CNN Hyperspectral Image Classification Using Training Sample Augmentation with Generative Adversarial Networks," Proc. 13th International Conference on Communications (COMM2020), Bucharest, Romania, Electr Netowork, June 19-20, 2020, pp.515-519, WOS:000612723900091

[4] **P. Diaconescu**, V. E. Neagoe, "Credit Scoring Using Deep Learning Driven by Optimization Algorithms," *Proc. 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI-2020)*, Bucharest, Romania, June 25-27, 2020, pp.1-6, WOS:000627393500021

[5] **P. Diaconescu**, V. Neagoe, „A Deep Learning Approach to Autonomous Driving in Urban Environment", accepted by Scientific Bulletin of the Politehnica University of Bucharest, Series C, Electrical Engineering and Computer Science, to be indexed in Web of Science

8.4 Perspectives of further development

Based on the work done so far, more developments can be initiated.

Related to the DCNN ensemble for the drunkenness state detection proposed in the chapter “An ensemble of deep convolutional neural networks for drunkenness detection using thermal infrared imagery” one can test the change of the DCNN used with other original networks or with state-of-the-art networks created after our paper has been published.

In the field of hyperspectral image classification, the DCNN proposed in the chapter “A highly configurable deep learning architecture for hyperspectral image classification” can be enhanced with the help of optimization algorithms that can automatically choose the optimum hyperparameters and optimum DCNN architecture. Other hyperspectral maps can also be used. Related to generation of artificial training pixels, other Generative Artificial Networks can be chosen and more (and longer) simulations can be used, because the simulation number has been limited by the long simulation runs.

The chapter regarding “Classification of credit requests using DCNN model optimization” could benefit trough introduction of different datasets and new optimization algorithms. A goal would be the automated selection of the optimization algorithm depending if dataset and final objective.

In the field of context recognition for autonomous driving, more measures for quality of object detection and error analysis could be added.

Bibliography

[1] A. Krizhevsky, Ilia, Sutskever, G. E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, 2012, https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] D. Ciresan, U. Meier, and J. Schmidhuber. *Multi-column deep neural networks for image classification*. Arxiv preprint arXiv:1202.2745, 2012.

[3] B. Hunicka, H. Laurell, H. Bergman, *Psychosocial characteristics of drunk drivers assessed by addiction severity index, prediction of relapse*, Scandinavian Journal of Public Health, vol. 38, no 1, pp. 71–77, 2010.

[4] D.A. Socolinsky, A. Selinger, J.D. Neuheisel, *Face recognition with visible and thermal infrared imagery,* Computer Vision and Image Understanding, vol. 91, pp. 72-114, 2003.

[5] V.E. Neagoe, A.D. Ropot, *Concurrent self-organizing maps for pattern classification*, Proc. First IEEE International Conference on Cognitive Informatics, p. 304-312, 2002.

[6] V.E. Neagoe, A.P. Barar, N. Sebe, P. Robitu, *A deep learning approach for subject independent emotion recognition from facial expressions*, Proc. 1st International Conference on Image Processing and Pattern Recognition (IPPR13), pp. 93-98, 2013.

[7] V.E. Neagoe, S.V. Carata, *Subject independent drunkenness detection using pulse-coupled neural network segmentation of thermal infrared facial imagery*, Int. J Math. Comput. Methods, pp. 305-312, 2016.

[8] V.E. Neagoe, S.V. Carata, *Drunkenness diagnosis using a neural network-based approach for analysis of facial images in the thermal infrared spectrum,* Proc. 6th IEEE International Conference on E Health and Bioengineering (EHB), pp. 165-168, 2017.

[9] B. Chen, G. Polatkan, G. Sapiro, D. Blei, D. Dunson, L. Carin, *Deep learning with hierarchical convolutional factor analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no 8, pp. 1887-1901, 2013.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, A. Sutskever, R. Salakhutdinov, *Dropout: A simple way to prevent neural networks from overfitting*, Journal of Machine Learning Research, vol. 15, pp. 1929-1958, 2014.

[11] Y. LeCun, Y. Bengio, G. Hinton, *Deep learning*, Nature, pp. 436-444, 2015.

[12] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*, Massachusetts: MIT Press, 2016.

[13] Q. Gao, S. Lim, *Hyperspectral Image Classification Using Convolutional Neural Networks and Multiple Feature Learning*, Remote Sensing Journal, 2018

[14] V.E. Neagoe, V. Chirilă-Berbentea, *A Novel Approach for Semi-supervised Classification of Remote Sensing Images using a Clustering-based Selection of Training Data according to their GMM Responsibilities*, Proceedings of 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS2017), Fort Worth, Texas, USA, July 23–28, 2017, pp. 4730-4733

[15] V. E. Neagoe, A. D. Ciotec, *A New Approach for Accurate Classification of Hyperspectral Images Using Virtual Sample Generation by Concurrent Self-Organizing Maps,* Proc. IEEE International Geoscience & Remote Sensing Symposium (IGARSS2013), Melbourne, Australia, 21-26 July, 2013, pp. 1031-1034

[16] M. Bejiga, F. Melgani, *An adversarial approach to cross-sensor hyperspectral data classification,* Proc. Int. Geoscience Remote Sens. Symp. (IGARSS 2018), Valencia, Spain, pp. 3583-3586, July 22-27, 2018.

[17] S. Fang, D. Quan, S. Wang, L. Zhang, L. Zhou, *A two-branch network with semi-supervised learning for hyperspectral classification*, Proc. Int. Geoscience Remote Sens. Symp. (IGARSS 2018), Valencia, Spain, pp. 3868 – 3871, July 22-27, 2018.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair et al., *Generative adversarial nets,* arXiv:1406.2661,

[19] V.E. Neagoe, A.D. Ciotec, L. Bruzzone, *A weakly-supervised change detection technique for SAR images based on deep learning and synthetic training data generated by an ensemble of self-organizing maps,* Proc. Int. Geoscience Remote Sens. Symp. (IGARSS 2019), Yokohana, Japan, pp.1669-1672, July 28-August 2, 2019.

[20] V. E. Neagoe, R. Iatan, I.F. Iatan, *A nonlinear neuro-fuzzy model for prediction of daily exchange rates*, Proc. 5th International Symposium on Soft Computing for Industry, held at the World Automation Congress (WAC'04), Seville, June 28-July 1, 2004, published in : Soft Computing with Industrial Applications, Vol. 17, Book Series: TSI Press, pp. 573-578

[21] T. Damrongsakmethee, V. Neagoe, *A Neural NARX Approach for Exchange Rate Forecasting,* in Proc Electronics, Computers and Artificial Intelligence (ECAI), 11th edition, Pitesti, 2019, pp. 1-6

[22] V. Neagoe, A. Ciotec, G. Cucu, *Deep convolutional neural networks versus multilayer perceptron for financial prediction*, in Proc 2018 International Conference on Communications (COMM), 14-16 June 2018, pp 201-206

[23] J. Bergstra, D, Yamins, D. Cox, *Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures*, Proc. of the 30th International Conference on Machine Learning (ICML) 2013

[24] J. Bergstra, Y. Bengio, *Random Search for Hyper-Parameter Optimization*, Journal of Machine Learning Research 13, pp. 281-305, 2012

[25] A. M. Turing, *Computing Machinery and Intelligence*, Mind, Volumul 49, pp. 433-460, Octombrie 1950

[26] Dheekonda, Raja & Panda, Sampad & Khan, Md & Hasan, Mohammad & Anwar, Sohel *Object Detection from a Vehicle Using Deep Learning Network and Future Integration with Multi-Sensor Fusion Algorithm*. 10.4271/2017-01-0117.

[27] A.D. Ciotec, V. E. Neagoe, A. P. Bărar, *Concurrent Self-Organizing Maps for Pedestrian Detection in Thermal Imagery*, Scientific Bulletin of the Polytechnic University of Bucharest, Series C, Vol. 75, Iss. 4, 2013, ISSN 2286-3540.

[28] V. Neagoe, M. Vâlcu, and B. Sabac, *A Neural Approach for Detection of Road Direction in Autonomous Navigation,* in: Computational Intelligence, Theory and Applications, (ed. B. Reusch), Elsevier, Berlin-New York, 1999, pp. 324-333.

[29] V.E. Neagoe, C.T. Tudoran, *A Neural Machine Vision Model for Road Detection in Autonomous Navigation*, Scientific Bulletin of the Politehnica University of Bucharest, Series C - Electrical Engineering, No 2, 2011, pp. 167-178.