

COMPLEXITY CONTROLLED NEURAL NETWORKS

HIMANSHU PANT



DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY DELHI

OCTOBER 2021

© Indian Institute of Technology Delhi (IITD), New Delhi, 2021

COMPLEXITY CONTROLLED NEURAL NETWORKS

by

HIMANSHU PANT

Department of Electrical Engineering

Submitted

**in fulfilment of the requirements of the degree of Doctor of Philosophy
to the**



INDIAN INSTITUTE OF TECHNOLOGY DELHI

OCTOBER 2021

Certificate

This is to certify that the thesis entitled “**Complexity Controlled Neural Nets**”, being submitted by **Himanshu Pant** for the award of the degree of **Doctor of Philosophy** to the Department of Electrical Engineering, Indian Institute of Technology Delhi, is a record of bonafide work done by him under my supervision and guidance. The matter embodied in this thesis has not been submitted to any other University or Institute for the award of any other degree or diploma.

Dr. Jayadeva

Professor

Department of Electrical Engineering,

Indian Institute of Technology Delhi,

Hauz Khas, New Delhi - 110016,

INDIA.

Acknowledgments

I would like to thank my supervisor Prof. Jayadeva, Research Committee members Professors Santanu Chaudhary, I.N. Kar and Amit Kumar; faculty members with whom I worked and published - Professors Suresh Chandra and Amit Bhaya; friends and colleagues Sumit Soman, Mayank Sharma, Pawas Gupta; department staff members Rakesh, Yatindra, Mukesh and Ritwick; my mother, [wife](#), [kids](#) and brother for supporting me through this journey.

(Himanshu Pant)

Abstract

The Vapnik–Chervonenkis (VC) dimension of a learning machine is a key concept in the area of model complexity control. The VC dimension is a measure of the capacity of a learning machine. With rapid increases in the availability of training data, and subsequent advent of monolithic deep learning architectures, capacity control of a learning machine has become even more significant. The number of weights in many deep architectures significantly outnumbers the dataset size. The capacity of a machine directly affects its generalizability, due to which architectures with excess capacity tend to over-fit, resulting in poor test performance.

The relationship between the capacity of a learning machine and its generalizability may be best understood from Vapnik’s risk formula, which provides an upper bound on the total risk of a learning machine in terms of its empirical risk and structural risk for a given number of training samples. It shows that there exists an optimal VC dimension for which upper bound on total risk is minimum and if we increase machine capacity beyond this, bound on total risk exhibits increasing trend, leading to poor generalization. This problem is further aggravated for cases where training data is having class imbalance and for some deep learning architectures which are in general difficult to stabilize and control such as generative adversarial networks (GANs).

Several attempts has been made in literature to estimate bounds on structural risk, but they do not translate into a tractable problem to be optimized in terms of learning parameters of the machine. The recently proposed Minimal Complexity Machine (MCM) has shown that a bound on the VC dimension can be directly incorporated into the objective function of a learning machine, resulting in better generalizing capabilities and sparser solutions. The present

work focuses primarily on a minimal complexity implementation of Neural Networks, termed as the Low Complexity Neural Network (LCNN). This is done by deriving a differentiable and scalable loss function, that not only minimizes the empirical risk, but also the structural risk of a classifier. This implementation was extended to various deep learning architectures and tested on multiple benchmark datasets.

The Low Complexity loss function was also derived and implemented for Generative Models. It was shown that by introducing complexity term in Generative Adversarial Networks (GANs) and its variants, training stability and quality of generated image was improved and issue of mode collapse was avoided.

Next, Twin Neural Networks, a novel neural network architecture for handling class imbalance is described. Proposed class of machine not only provides a scalable framework for large sized datasets but also keeps structural risk at check resulting in good generalization. It was shown over several benchmark datasets with very high class imbalance that Twin Neural Networks produces better results than that of competing methods. A real world application of Twin Neural Network for classification of ElectroEncephaloGram (EEG) signals is also discussed.

Subsequent to this, a direct realization of the Minimal Complexity Machine (MCM) was implemented via a neurodynamical system. The proposed system solves coupled differential equations in a multilayer neural network setting, enabling stochastic learning and implicit kernel optimization. The resulting system converges to MCM solution providing better generalization and scalability. Numerical experiments on benchmark datasets show that the proposed approach is scalable and accurate, and learns models with improved accuracies and fewer support vectors.

The last part of the thesis takes an alternative route to Risk Minimization

via Data Augmentation. The proposed method is termed as Eigenbag, which presents framework for augmenting data at scale. This becomes important for scenarios, where the number of samples available for training are small. Eigenbag generates new data samples in a lower dimensional space via bagging and projects it back to the original dimension. A deep Learning version of Eigenbag is also discussed that uses Convolutional Autoencoder to increase training sample size at scale. The proposed method is also tested against competing methods.

सार

एक लर्निंग मशीन की वापनीक-चेरवोनेनकिस (वी सी) आयाम , मॉडल जटिलता नियंत्रण के क्षेत्र में एक महत्वपूर्ण अवधारणा है। वीसी आयाम एक लर्निंग मशीन की क्षमता का भी प्रतिनिधित्व करता है | प्रशिक्षण के आंकड़ों की उपलब्धता में वृद्धि और उसके बाद डीप लर्निंग आर्किटेक्चर के आगमन के साथ , एक लर्निंग मशीन की क्षमता नियंत्रण और भी महत्वपूर्ण हो गयी है। मशीन की क्षमता सीधे इसकी सामान्यता को प्रभावित करती है , जिसके कारण अनुकूलतम मान से अधिक क्षमता वाली मशीनें ओवरफिट होने के कारण खराब परीक्षण प्रदर्शन करती हैं ।

एक मशीन की क्षमता और इसकी सामान्यता के बीच संबंध को सबसे अच्छी तरह से वापनीक के जोखिम सूत्र द्वारा समझा जा सकता है जो किसी प्रशिक्षण नमूनों की संख्या के लिए अपने अनुभवजन्य जोखिम और संरचनात्मक जोखिम के संदर्भ में एक ऊपरी सीमा प्रदान करता है। यह दर्शाता है कि एक इष्टतम वीसी आयाम मौजूद है , जिसके लिए कुल जोखिम पर ऊपरी सीमा न्यूनतम है और अगर हम इससे परे मशीन की क्षमता बढ़ाते हैं , तो कुल जोखिम की सीमा बढ़ने लगती है , जिससे सामान्यीकरण प्रभावित होता है। यह समस्या उन मामलों के लिए और बढ़ जाती है जहाँ प्रशिक्षण डेटा में कक्षा असंतुलन है और कुछ डीप लर्निंग आर्किटेक्चर के लिए जो सामान्य रूप से स्थिर और नियंत्रित करने मुश्किल हैं जैसे कि जेनरेटिव अड्वेंसरेडल नेटवर्क(GANs) |

संरचनात्मक जोखिम पर सीमा का अनुमान लगाने के लिए साहित्य में कई प्रयास किए गए हैं , लेकिन वे मशीन के सीखने के मापदंडों के संदर्भ में अनुकूलित किए जाने योग्य ट्रेक्टेबल समस्या में परिवर्तित नहीं होते हैं। हाल ही में प्रस्तावित मिनिमल कम्प्लेक्सिटी मशीन (एमसीएम) ने दिखाया है कि वीसी आयाम को सीधे एक सीखने की मशीन के उद्देश्य में शामिल किया जा सकता है जिसके परिणामस्वरूप सामान्यीकरण क्षमताओं और विरल समाधानों को बेहतर बनाया जा सकता है। वर्तमान कार्य मुख्य रूप से न्यूरल नेटवर्क के न्यूनतम जटिलता कार्यान्वयन पर केंद्रित है जिसे एक डिफ्रेंसिअबल और स्केलेबल लॉस फंक्शन को प्राप्त करके लो कम्प्लेक्सिटी न्यूरल नेटवर्क (LCNN) कहा गया है , जो न केवल एक क्लासिफायरियर के अनुभवजन्य जोखिम को कम करता है बल्कि संरचनात्मक जोखिम को भी कम करता है। इस कार्यान्वयन को विभिन्न डीप लर्निंग आर्किटेक्चर में कार्यान्वित किया था और बेंचमार्क डेटासेट पर परीक्षण किया गया। लो कम्प्लेक्सिटी लॉस फंक्शन जेनरेटिव मॉडल्स के लिए भी कार्यान्वित किया गया। यह प्रस्तुत किया गया था कि जेनरेटिव अड्वेंसरेडल नेटवर्क (GANs) और इसके वेरिएंट्स में लो कम्प्लेक्सिटी का समावेश करने से , प्रशिक्षण की स्थिरता और उत्पन्न छवि की गुणवत्ता में सुधार आया और मोड कोलैप्स के मुद्दे से भी बचा गया।

तदोपरांत , वर्ग असंतुलन से निपटने के लिए , द्विन न्यूरल नेटवर्क , एक नवीन न्यूरल नेटवर्क का वर्णन किया गया है। मशीन का प्रस्तावित वर्ग न केवल बड़े आकार के डेटासेट के लिए एक स्केलेबल फ्रेमवर्क प्रदान करता है , बल्कि संरचनात्मक रूप से जोखिम भी कम रखता है जिसके परिणामस्वरूप अच्छा सामान्यीकरण होता है। यह उच्च वर्ग असंतुलन वाले कई बेंचमार्क डेटासेट पर दिखाया गया था कि द्विन न्यूरल नेटवर्क प्रतिस्पर्धी तरीकों की तुलना में बेहतर परिणाम देता है। इलेक्ट्रोएन्सेफ्लोग्राम (EEG) संकेतों के वर्गीकरण के लिए द्विन न्यूरल नेटवर्क का एक वास्तविक अनुप्रयोग भी प्रदर्शित किया गया है।

इसके बाद , न्यूरोडायनामिकल प्रणाली के माध्यम से मिनिमल कम्प्लेक्सिटी मशीन (एमसीएम) को लागू किया गया है। प्रस्तावित प्रणाली एक बहुपरत न्यूरल नेटवर्क सेटिंग में युग्मित समीकरणों को हल करती है और स्टोकास्टिक सीखने और अंतर्निहित कर्नेल अनुकूलन को सक्षम करती है। परिणामस्वरूप प्रणाली एम सी एम समाधान की ओर अभिसरित होकर बेहतर सामान्यीकरण और स्केलेबिलिटी प्रदान करती है। बेंचमार्क डेटासेट पर संख्यात्मक प्रयोग बताते हैं कि प्रस्तावित दृष्टिकोण स्केलेबल और सटीक है , और बेहतर सटीकता और कम सपोर्ट वेक्टर्स के साथ मॉडल सीखता है।

थीसिस का अंतिम भाग डेटा ऑप्टिमाइजेशन तकनीक के माध्यम से जोखिम न्यूनतमकरण के लिए एक वैकल्पिक मार्ग लेता है। प्रस्तावित विधि को आएगॉनबैग कहा गया है , जो बड़े पैमाने पर डेटा को बढ़ाने के लिए रूपरेखा प्रस्तुत करता है। यह उन परिदृश्यों के लिए और भी महत्वपूर्ण हो जाता है , जहाँ प्रशिक्षण के लिए उपलब्ध नमूनों की संख्या बहुत कम है। आएगॉनबैग एक कम आयामी स्थान में बैगिंग के माध्यम से नए डेटा नमूने उत्पन्न करता है और इसे मूल आयाम पर प्रक्षेपित करता है। आएगॉनबैग

का एक डीप लर्निंग संस्करण भी प्रस्तुत किया गया है , जो बड़े पैमाने पर प्रशिक्षण के नमूने को बढ़ाने के लिए कॉवोलूशनल ऑटोकेनडर का उपयोग करता है। प्रतिस्पर्धी तरीकों के खिलाफ प्रस्तावित विधि का परीक्षण भी किया गया है।

Contents

Certificate	i
Acknowledgements	ii
Abstract	iii
List of Figures	x
List of Tables	xv
1 Introduction	1
1.1 Scope and Objectives	1
1.2 Minimal Complexity Machines	3
1.3 Twin Support Vector Machines	4
1.4 Data Augmentation	5
1.5 Organization of the Thesis	5
1.6 Concluding Remarks	7
2 Low Complexity Neural Networks	9
2.1 Introduction	9
2.2 Related Work	11
2.3 The Low Complexity Neural Network	13

2.3.1	Bound on Neural Network Parameters	21
2.3.2	Application of the VC Bound on Hidden Layers	22
2.4	Experiments	24
2.4.1	Experiments on UCI datasets	24
2.4.2	Test Accuracies on UCI Datasets	26
2.4.3	Training Time on UCI Datasets	26
2.5	Conclusion	31
3	LCNN for Deep Learning	32
3.1	Deep Learning Architectures	33
3.1.1	CNN	33
3.1.2	DBN	34
3.1.3	Sparse and Denoising Autoencoders	34
3.2	Experimental Results	35
3.2.1	LCNN-CNN	35
3.2.2	LCNN-DBN	35
3.2.3	LCNN-Denoising Autoencoders	37
3.2.4	Denoising Autoencoders Feature Visualization	38
3.2.5	Experiments on large datasets	38
3.2.6	Gradient analysis for LCNN	47
3.3	Conclusion	47
4	LCNN For Generative Models	50
4.1	Introduction	50
4.2	Low Complexity Neural Network	53
4.3	GANs with LCNN	55
4.3.1	LCNN-GAN	56
4.4	Experiments	57

4.4.1	Results on LCNN-GAN	57
4.4.2	Results on LCNN-DCGAN	60
4.4.3	CIFAR-10	61
4.4.4	SVHN	64
4.4.5	Quantitative Evaluation of LCNN-GAN models	65
4.5	Conclusion	67
5	Twin Neural Networks	68
5.1	Introduction	68
5.2	The Twin Support Vector Machine	73
5.3	The Twin Neural Network Formulation	76
5.4	Twin Neural Network for Multi-class Datasets	82
5.5	Experiments and Discussion	84
5.5.1	Results on UCI datasets	85
5.5.2	Results on highly unbalanced datasets	88
5.5.3	Results on multi-class datasets	94
5.6	Experiments on scalability	97
5.7	Benchmarking Twin Neural Networks	99
5.7.1	Weighted Lagrangian Twin SVM (WLTWSVM)	99
5.7.2	Maximum Margin of Twin Spheres Support Vector Machine (MMTSSVM)	100
5.8	Twin Neural Networks for EEG Signal Classification	103
5.8.1	EEG Datasets and Processing Pipeline	105
5.8.2	Results	108
5.9	Conclusion and Future Work	113
6	Minimal Complexity Neurodynamical System	115
6.1	Introduction	115

6.2	The Neurodynamical Minimal Complexity Machine	119
6.3	Simulations of the MCM Neurodynamical System	127
6.4	Developing a Network Architecture	131
6.4.1	Multiclass Classification	133
6.5	Results	134
6.5.1	Results using the MCM Neurodynamical System	134
6.5.2	Results using the MCM Neurodynamical System Neural Network	136
6.5.3	Results using the MCM Neurodynamical System Neural Network for Multi-class Datasets	138
6.6	Conclusion	140
7	Eigenbag: Data augmentation at scale	142
7.1	Introduction	142
7.2	<i>EigenSample</i>	144
7.3	EigenBag	146
7.3.1	Eigenbag for Autoencoders	146
7.4	Results	148
7.5	Conclusions	151
8	Conclusions and Future Work	152
8.1	Future Work	157
	List of Publications	179
	Brief Biodata of Author	181

List of Figures

2.1	Flowchart for experimental procedure.	25
2.2	Effect of dataset size on LCNN training time.	29
2.3	Effect of dataset size on LCNN accuracy.	30
3.1	LCNN-CNN architecture.	33
3.2	LCNN-CNN error rate	36
3.3	LCNN-DBN error rate	37
3.4	LCNN-Autoencoder error rate	39
3.5	Denoising Autoencoder-LCNN Feature Learning Experiments with 30% corruption,	39
3.6	Convergence of AlexNet on ImageNet for hyperparameters in the LCNN objective. The performance rises initially during training and persists until convergence, indicating that the LCNN learns a good model early with few training samples.	42
3.7	t-SNE 2D visualizations of a few samples from test set of CI- FAR10. We see that model complexity control consistently en- forces crisper, more distinct clustering of classes in feature space.	43

3.8	Samples of image filters on the MNIST dataset obtained by (a) SAE using KL divergence only, and (b) SAE using KL divergence + LCNN. Note that filters obtained using the LCNN are visibly sharper. The Spatial-Spectral Entropy-based Quality (SSEQ) scores for (a) and (b) are 52.57 and 32.13, respectively, indicating that the LCNN filters are about 64% superior.	45
3.9	Histograms of weights on the MNIST dataset.	46
3.10	Mean gradients in the last two layers(avgGradOP-Final Layer,avgGradIP-Penultimate Layer) of a CNN with and without the LCNN term. Note that the mean gradient in the LCNN case is almost one order of magnitude larger. At the same time, the empirical error (blue curve) is lower in the LCNN case. A	48
3.11	Mean gradients in the last two layers(avgGradOP-Final Layer,avgGradIP-Penultimate Layer) of a regularized neural network with and without the LCNN term. Note that the mean gradient in the LCNN case is almost one order of magnitude larger. At the same time, the empirical error (blue curve) is lower in the LCNN case.	49
4.1	Mode Collapse Analysis for MNIST Data	58
4.2	GAN Generated Fake MNIST Samples at different training stages	58
4.3	GAN-LCNN Generated Fake MNIST Samples at different training stages (numbers indicate mini batches processed)	58
4.4	Balance between generator and discriminator for C=0.01	59
4.5	Balance between generator and discriminator for different values of C for LCNN-GAN	59
4.6	Energy Content and Modes study for GAN and LCNN-GAN	60
4.7	Training Stability for DCGAN and LCNN-DCGAN	61

4.8	Fake Images Generated from DCGAN before and after Unstable/Mode Collapse compared with LCNN-DCGAN	62
4.9	Generative and Discriminative Loss for DCGAN on CIFAR-10	62
4.10	Fake Images Generated from DCGAN Before and After Unstable/Mode Collapse for CIFAR-10 dataset	63
4.11	Training Stability for LCNN-DCGAN for different values of C.	63
4.12	Fake Images Generated from LCNN-DCGAN for CIFAR-10 dataset	64
4.13	DCGAN: Mode Collapse/Unstable after 1900x20 mini Batches.	64
4.14	Fake Images Generated from DCGAN Before Unstable/Mode Collapse for SVHN Dataset	65
4.15	LCNN DCGAN: Stable Training Value of C = 0.01	65
4.16	Fake Images Generated from LCNN-DCGAN	66
4.17	Histogram of weights for GAN and LCNN-GAN for MNIST	67
5.1	Architecture of the Twin Neural Network	70
5.2	Comparing TWSVM and Twin NN	72
5.3	Twin NN for multiclass datasets	83
5.4	Compared to a conventional neural network, a representative activation function for the multiclass Twin NN attains a value of 0 for the specific class and ± 1 for other classes.	85
5.5	Training procedure for UCI datasets	86
5.6	Flowchart for computing accuracy of unbalanced dataset.	89
5.7	Confusion Matrix	90
5.8	Average Values of G-Means, F-Measure and MCC for Unbalanced Datasets.	94
5.9	Confusion matrices for different models for the Connect4 dataset.	96
5.10	Average Test Values of G-Means, F-Measure, MCC and Training Time for Scale up Experiment.	98

5.11	Processing pipeline for EEG signals using the Twin Neural Network	106
5.12	Confusion matrices for a few representative subjects for multi-class classification.	109
5.13	Confusion matrices for a few representative subjects for binary classification.	112
6.1	(a) Example of a linearly separable dataset, and (b) a dataset with points drawn from a normal distribution.	128
6.2	Plots of convergence of the decision variables w , b and h , and their first derivatives \dot{w} , \dot{b} and \dot{h} with time, for the linearly separable dataset shown in Fig. 6.1a.	129
6.3	Plots of convergence of the decision variables w , b and h , and their first derivatives \dot{w} , \dot{b} and \dot{h} with time, for the dataset with points drawn from a normal distribution, as shown in Fig. 6.1b .	130
6.4	Developing the MCM Neurodynamical System as a Neural Network. The solid lines represent the connections between neurons of the neural network layers, while dashed lines denote updates in solving the neurodynamical system.	132
6.5	Box Plots of test set accuracies using the MCM Neurodynamical System on representative datasets.	137
6.6	Box Plots of test set accuracies for the MCM Neurodynamical System Neural Network architecture on representative datasets. .	139
6.7	Histogram of weights for Neural Network and MCM NeuroDynamical System Neural Network for UCI dataset Ionosphere . . .	140
7.1	Eigenbag for autoencoders	147
7.2	Autoencoder Architecture	148
7.3	Generated Samples of digit 5	149

7.4 CNN classifier Architecture	150
---	-----

List of Tables

2.1	Summary of methods with which the LCNN is compared	26
2.2	Classification accuracies on the UCI Datasets	27
2.3	Training Time for the UCI Datasets	28
2.4	p-values for LCNN vs other approaches	31
3.1	LCNN-CNN Error Rates	35
3.2	LCNN-DBN Error Rates	36
3.3	LCNN-DBN Error Rates	36
3.4	LCNN-DeAE Error Rates	38
3.5	LCNN-DeAE Error Rates	38
3.6	Summary of Deep Learning Architectures used for Experiments .	40
3.7	Notations used.	41
3.8	Performance on ImageNet for the best hyperparameter settings (for GoogLeNet, LCNN was applied to all layers).	42
3.9	Results on CIFAR-10 with Caffe Quick CIFAR-10 architecture. .	43
4.1	Results on inception score for CIFAR-10	66
5.1	Results on UCI datasets for the Twin NN.	87
5.2	Wilcoxon signed ranks test for the Twin NN	87

5.3	Description of large unbalanced datasets, attributes are numeric (N) or categorical (C)	89
5.4	G-means of large datasets with high unbalance	91
5.5	F-measure for large datasets with high unbalance	91
5.6	MCC for large datasets with high unbalance	92
5.7	p-values for Wilcoxon signed ranks test for different measures on unbalanced Datasets	93
5.8	Average Values of G-Means, F-Measure and MCC for Unbalanced Datasets	93
5.9	Training time for unbalanced datasets	93
5.10	Multi-class datasets used in experiments	95
5.11	Results on multi-class datasets (test set accuracy in %)	95
5.12	Dataset for scaling experiment	97
5.14	Training Time for scaling experiment	97
5.13	Performance parameters for scaling experiment	99
5.15	Results on unbalanced datasets using the WLTWSVM	101
5.16	Results on unbalanced datasets using the MMTSSVM	102
5.17	Kappa values for multi-class classification dataset (BCI Compe- tition IV, Dataset 2a), compared with SVM with RBF kernel, Random Forests and W-SVM.	110
5.18	Comparison of average kappa values.	111
5.19	Kappa values for binary classification dataset (BCI Competition IV, Dataset 2b), compared with SVM with RBF kernel, Random Forests and W-SVM.	111
5.20	Comparison of average kappa values.	113
6.1	Test Set Accuracies for the Linear MCM Dynamical System (mean ± standard deviation). Best results are shown in boldface.	134

6.2	Test Set Accuracies and number of Support Vectors (#SVs) for the Kernel MCM Dynamical System (KMCM-DS) compared with standard RBF Kernel SVM (KSVM) (mean \pm standard deviation). Best results are shown in boldface.	135
6.3	Comparative results for test set accuracy on datasets using the neural network architecture of the MCM Neurodynamical System (last column). Best results are shown in boldface.	136
6.4	Results for multiclass datasets (mean \pm standard deviation), showing that the performance of the proposed MCM Neurodynamical System is either the best (shown in boldface) or close to the best, over a collection of benchmark datasets.	138
7.1	Comparison of Data Augmentation Methods	151