INTELLIGENT TRACK PITCH CLASSIFICATION IN HARD DISK DRIVE WITH ENSEMBLE OF K-MEAN CLUSTERING AND DISTRIBUTED NEURAL NETWORK CLASSIFIER WITH NEURAL NETWORK FUSION

WIWAT RODSUD

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN DATA STORAGE TECHNOLOGY
INTERNATIONAL COLLEGE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2013

KMITL-2013-IC-M-005-015

Thesis:	Intelligent Track Pitch Classification in Hard Disk Drive with Ensemble of K-Mean Clustering and Distributed Neural Network Classifier with Neural Network Fusion
Student:	Wiwat Rodsud
Student ID:	52600623
Degree:	Master of Engineering (M.Eng)
Program:	Data Storage Technology
Year:	2013
Thesis Advisor:	Asst. Prof. Dr. Yuttana Kitjaidure

ABSTRACT

It is well-known that track per inch (TPI) and bit per inch (BPI) are fundamental parameters defining HDD areal density (bits per square inch). TPI and BPI settings are calibrated to achieve bit error rate (BER) performance at the desired areal density. However, both BPI and TPI calibrations are complex and time-consuming processes. This paper proposes an intelligent TPI calibration method using the distributed neural network (NN) classification model together with neural network fusion model and it is compared to the conventional NN Classification. The experiment results show that the proposed TPI calibration method provides significantly better performance than the conversional NN classification method. Moreover, the TPI calibration of the proposed method has significantly lesser time consumed at the same BER performance as the existing TPI Calibration method. We therefore propose the intelligent TPI calibration method which integrates the ensemble K-Mean Clustering and Distributed NN classification with NN fusion.

Index Terms – Hard Disk Drive, Neuron Network Fusion, Head-Disk Interaction, Classification, Distributed Algorithm

Acknowledgements

Many thanks to my advisor at College of Data Storage Technology and Application, King Mongkut's Institute of Technology Ladkrabang : Asst. Prof. Dr. Yuttana Kidjaidure for offering valuable advice, support and guidance throughout my Masters study.

Many thanks to my colleagues at Western Digital (Thailand) on the valuable comments, help and support during my Masters study. Appreciation also goes to my supervisor in the Head and Media Engineering organization for supporting my Masters study. Special appreciation goes to Tawan and the Senior Management team at Western Digital (Thailand) for granting this scholarship and opportunity, without which I would not have completed my Masters study. Special mention goes to Kittiphan, CW Mak, Waraporn and Kritsada who have provided invaluable support and guidance during my study.

I would also like to thank my parents and family for the support all these years.

Wiwat Rodsud

Contents

Contents (Cont.)

List of Figures

List of Figures (Cont.)

List of Tables

Chapter 1

Introduction

It is well-known that track per inch (TPI) and bit per inch (BPI) are fundamental parameters defining HDD areal density (bits per square inch). TPI and BPI settings are calibrated to achieve bit error rate (BER) performance at the desired areal density. However, both BPI and TPI calibrations are complex and time-consuming processes. This paper proposes an intelligent TPI calibration method using the distributed neural network (NN) classification model together with neural network fusion model and it is compared to the conventional NN Classification. The experiment results show that the proposed TPI calibration method provides significantly better performance than the conventional NN classification method. Moreover, the TPI calibration of the proposed method has significantly lesser time consumed at the same BER performance as the existing TPI Calibration method. We therefore propose the intelligent TPI calibration method which integrates the Ensemble K-Mean Clustering and distributed NN classification with NN fusion.

1.1 Statement and Significance of the Problem

In the high density track pitch implementation of many Hard Disk Drive (HDD) designs in the modern day to meet ever-increasing areal density challenges, the calibration of track pitch to accommodate the read/write transducer is the primary concern. Track pitch calibration algorithms emerged from the need to accurately determine the track density to match the areal density design at an acceptable bit-error-rate (BER) performance, such as [1] and [2], address system stability at high density track implementations. The existing TPI Calibration process evaluates one TPI setting at a time in a given set of TPIs until the TPI meets both capacity and performance. Each calibration process step requires time and the total time consumed is depended on the number of heads, zones and TPI settings.

The reduction of the TPI Calibration time significantly improves HDD manufacturing throughput. However, the HDD drive performance and areal density cannot be compromised (trade off) with the reduced process time. It is observed that the results are often not repeatable or reproducible. Empirical data from failure analysis also suggest that the accuracy of the track pitch calibration contributes largely to bit-error-rate performance. This further motivates the search of a quicker and more intelligent calibration method by the challenges to balance performance of areal density and calibration time.

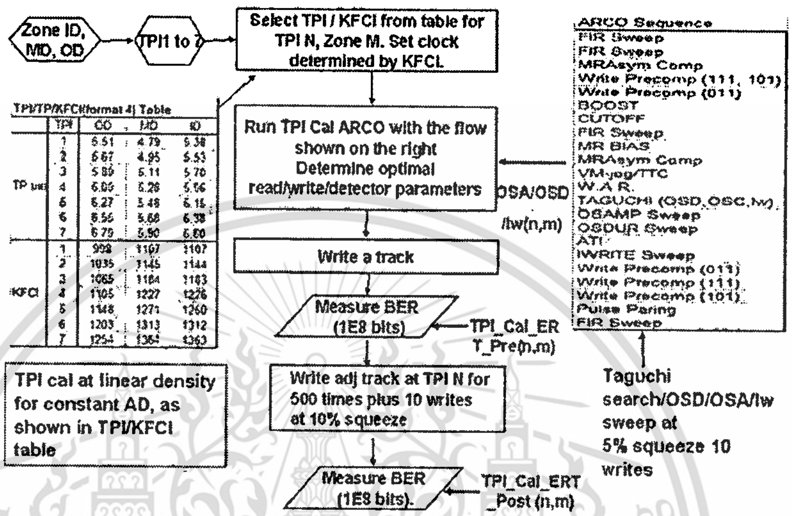


Fig. 1.1 Track Pitch Calibration and Optimization.

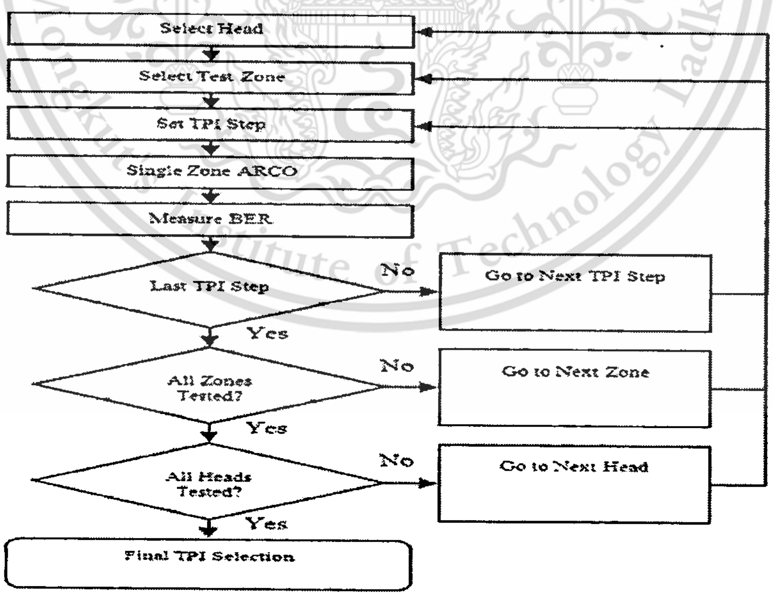


Fig. 1.2 Track Pitch Calibration Algorithm.

The total TPI calibration test time found uses the following rules,

$$t_{Max} = \sum_{i=1}^{H} t_h \times \sum_{j=1}^{TPI} t_{TPI} \times \sum_{k=1}^{Z} t_z \quad (1.)$$

Where t_{Max} is the maximum TPI test time, t_h is the test time consumption of each head, t_{TPI} is the time consumption of each TPI test time and t_Z is the time consumption of each zone, H is the number of heads, TPI is the number of designed TPI, Z is the number of zones in Hard Disk Drive.

1.2 Objective

The objectives of this study are following.

1.2.1 To study the statistic classification model and develop to predict the track pitch classification.

1.2.2 To develop the NN classification model to improve overall performance.

1.2.3 To employ the Distributed NN classification model to aggravate prediction accuracy method

1.2.4 To improve the performance by ensemble the NN classifier with K-mean method.

1.3 Expectation

This study is expected to provide the suitable classification method which is able to obtain the output accuracy close to actual track pitch calibration in HDD process. The statistic method also can be developed in order to improve the performance and feasible to apply in manufacturing. The entire goal is to achieve HDD maximum throughput and the performance still sustainable by using the proposed statistic model.

This study also aims to continue further development in order to support larger factory database by simplify the structure or add the other capable features. To enhance the classification process, the flexible designed method and model should be able to modify and suitable for any different data sets, this may require a search of experiment before implement the final model in HDD manufacturing.

1.4 Conceptual Framework

In this thesis, we study method and model of NN classification as a primary. The model also found out that we can design and adapt the structure to suit for HDD track pitch calibration parameters. In this work we approach the Distributed NN classifier to allow one single NN predict only one class of TPI before combine all result of each NN to be an input for NN fusion. Then the way this method classifies is localize on partitioning data set which is different from conventional NN classifier that classify universal input parameters.

Furthermore the other method is studied and include later to do pre-refine the data for NN classifier. We select cluster algorithm (K-Mean) added to the model so the ensemble concept is now involved in this thesis.

Then the final model is measured the BER in order to ensure HDD performance is not penalty from our proposed statistic classification method.

1.5 Hypothesis

In HDD TPI Calibration process, the read/write transducer parameters are measured and recorded. Generally, there is a fair correlation between the read/write transducer parameters and TPI shown in Fig. 1.3, however, the HDD BER performance is deteriorated significantly if the read/write transducer parameters have to predict TPI directly [3]. In order to maintain the HDD BER performance with lesser TPI calibration time, the new calibration technique is needed and statistic classification method is one of the approaches.

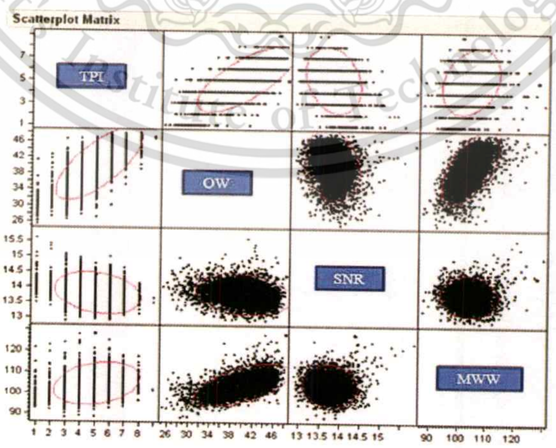


Fig. 1.3 Correlation between read/write transducer parameters and TPI.

1.6 Thesis Outline

The remainder of the thesis is organized as follows:

Chapter 2 presents the background information and reviews of theories, methods and parameters in the literature. The search includes the application of NN classification applied in a hard-disk drive device.

Chapter 3 explores the study of classification methods used in this literature. Neural Network concept is the major review as it is the most important tool used in this thesis as a part of classifier and fusion model. Additional study from clustering is included in this thesis after found out that there is a chance to improve the prediction accuracy by emerged with Conventional and Distributed NN classifier.

Chapter 4 discusses the results obtained on Conventional and Distributed NN Classifier and result after ensembles these two methods with K-Mean clustering as a pre-classifier model. We also discuss on the possibility to apply the proposed method to HDD manufacturing using BER performance and Test Time reduction against the statistic prediction accuracy. The result shown that the proposed method outperform when compare with conventional NN classification method and should be able to implement in HDD manufacturing.

Chapter 5 will conclude the final model of our work and the suggestion for future work.

Chapter2

Literature Reviews

2.1 Introduction

Hard disk drives provide data storage for data processing systems in computers and servers. Disk drives are also becoming increasingly pervasive media players, digital recorders, and other personal devices. Advances in disk drive technology have made it possible for a user to store an immense amount of digital information on increasingly smaller areas, and to selectively retrieve and alter portions of such information almost instantaneously. Particularly, recent developments have simplified disk drive manufacturing while yielding increased tack densities, thus promoting increased data storage capabilities at reduced costs.

Hard disk drives rotate high precision media, such as an aluminium or glass disk coated on both sides with thin films, to store information in the form of magnetic patterns. Electronic read/write heads suspended or floating only fractions of micro inches above the disk are used to either record information onto the thin film media, or read information from it. A read/write head may write information to the disk by creating an electromagnetic filed to orient a cluster of magnetic grains, known as a bit, in one direction or the other. To read information, magnetic patterns detected by the read/write head are converted into a series of pulses that are sent to the logic circuits to be converted to binary data and processed by the rest of the systems. To write information, a write element located on the read/write head generates a magnetic write filed that travels vertically through the magnetic recording layer and returns to the write element through a soft underlayer.

In Hard disk drive Track Pitch Calibration process requires adequate off-track capability (OTC) and sufficiently small adjacent track interference or erasure (ATI). Both OTC and ATI are sensitive to the track pitch on the magnetic media disk: each gets worse as the track pitch decreases. Despite those relationships, further reductions in track pitch are required for future generations of disk drives in order to increase the overall performance of drives. The OTC and ATI is then obtained from the signal or error rate loss, when the failure occurs during such ATI tests, the track pitch is adjusted on a trial and error basis. This process is repeated until an acceptable overall performance for the drive is attained. Although these solutions are workable, an improved solution for determining the track pitch in hard disk drives would be desirable.

For the HDD works there are many statistical analysis techniques has been studied in the several testing process, this motivate the search of other intelligent tool from data mining perspective integrate to HDD track pitch calibration process. Therefore the study and literature review concentrate on the relevant background information, theories, methods and parameters in regard to track pitch calibration. The search includes the application of intelligent feature extraction applied in a hard-disk drive device and detail of data mining concept.

2.2 Track Pitch Calibration in HDD

Several literature are studied to address the physical issue for HDD track pitch calibration process, For example; the accurately of on-track and off-track capability, adaptive track pitch profile adjustment and head switching and so on. All these improvement are approached to tackle the real problem however the new methods mostly deal with process enhancement in order to achieve a robust and sustainable track pitch calibration but result extended longer test time. This explore to us that the today track pitch calibration process is require a intelligent and faster method to determine the track pitch associate with the concern parameters.

Reference [4], [5] relates in general to hard disk drives in particular to improved system, method and apparatus for determining the track pitch to satisfy the requirement of both off-track capability and adjacent track erasure. The OTC requirement ensures sufficient soft error rate (SER) at off-track locations, while the ATI requirement ensures sufficient on-track SER after adjacent tracks are written multiple times. Both OTC and ATI are sensitive to the track pitch since they get worse as track pitch decreases. The mechanisms of their track pitch dependence, however, are different. OTC reflects direct partial erasure of the data track. ATI is caused by thermally-assisted switching of magnetization under a fringing filed from the head. As a result, the OTRC and ATI are affected differently by track pitch so they measured separately.

A high level flow diagram of track pitch calibration is show in Fig 2.1 The method begins by comprise measuring a 747 curve and determining an OTC track pitch; measuring ATI track pitch with different numbers of adjacent track writes; fitting a logarithmic function to the ATI track pitch and numbers of adjacent track writes data; calculating ATI track pitch for the desired number of adjacent track writes using a fitted function; and then selecting a final track pitch form the greater OTC and ATI track pitches.

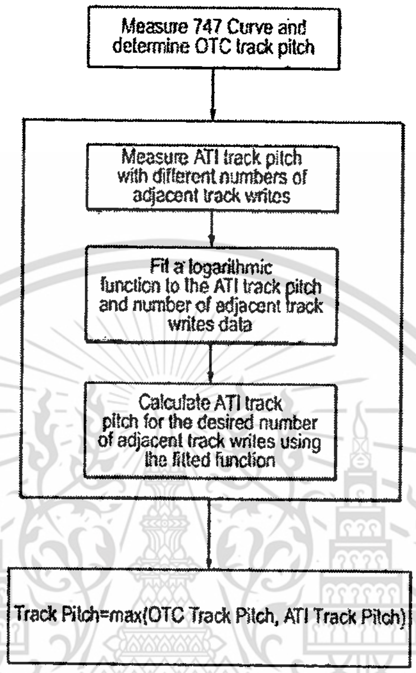


Fig. 2.1 A high level flow diagram of track pitch calibration.

The track pitch for the drive is set with the larger of the OTC and ATI track pitches. The OTC track pitch is measured with a 747 curve, and the ATI track pitch is measured by the positions of adjacent tracks at which the on-track error rate is not worse than a given value after the targeted number of adjacent track writes in the ATI requirement. Fig. 2.2 depicted bathtub curves under various squeeze track pitch (SQTP) curve. The on-track SER after the given number of adjacent track writes depends on the location of the adjacent tracks (e.g., the SQTP). The SQPT at which the OTC = 0 at the ATI SER requirement is defined as the ATI track pitch.

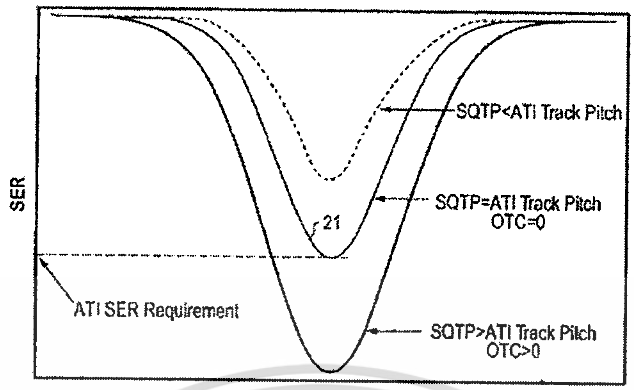
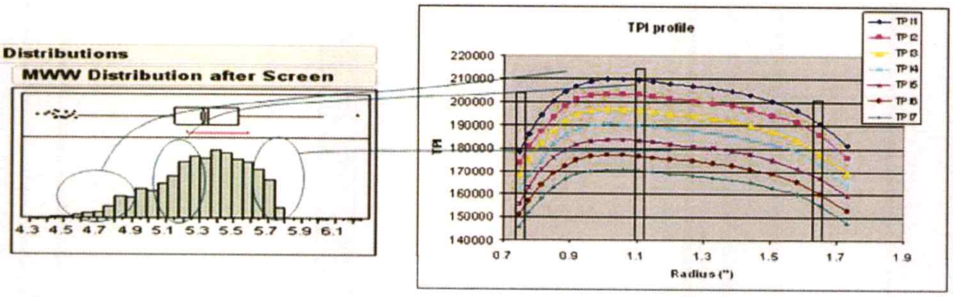


Fig. 2.2 A plot of off track position versus SER.

In order to increase the storage capacity or the reliability of HDD, Reference [6], [7] has been proposed to determine a data rack pitch for each head (each recording surface). Determination of the data track pitch so as to match head characteristics such as a read width or a write width lead to suppression of adjacent track interference (ATI) in data write and increased in data capacity per recording surface. To improve the performance, a technique has been proposed that performs a head switch for every data track in sequential data write or data read. However, if the recording surfaces have different data track pitches, their respective data tracks show different radial positions even if their data track numbers are the same. Accordingly, a transition onto the same data track on another recording surface requires additional time for a head seek. To avoid the additional seek, a data track of close radial position must be found on the recording surface of the transition destination in every head switch. To this end, additional resources and time are required for this operation. An effective approach to overcome this problem is a data track format in which a recording surface is constituted by multiple bands. Each band is constituted by multiple consecutive data tracks. Upon completion of an access to one data track, the HDD selects an adjacent data track in the same band as the next data track and switches heads at an end of the band. This reduces the number of head switches and suppresses increase in additional process time due to the head switches.

In Reference [8], [9] an ideal data tracks per inch (TPI) profile is determined for the disk, and an actual data TPI profile is generated that estimates the ideal data TPI profile by combing a first segment of a first predetermined data TPI profile and a second segment of a second predetermined data TPI profile. A predetermined data rate (linear bit density) is defined across each segment of the predetermined data TPI format such that there may be a change in the data rate at the segment boundaries. The disk comprises a plurality of TPI zones, where each TPI zone defines a band of data tracks. The data rate (linear bit derisity) of the data tracks varies between the TPI zones, for example, increases toward the outer diameter TPI zones similar to the physical zones. The disk drives comprises multiple disk surfaces, the TPI zone boundaries may be different across the disk surfaces. Each TPI segment comprises a constant data TPI over a redial segment of the disk (i.e., over a plurality of the data tracks) one or more clusters that define a serpentine pattern for accessing the data tracks across multiple disk surfaces. The boundaries of each TPI segment as well as the boundaries of each TPI zone align with ta cluster boundary. The ideal data TPI profile may be determined for each disk surface using any suitable technique. For example, a test pattern may be written over a plurality of adjacent tracks at a particular radial location and the bit error rate measured for varying data TPI setting. The data TPI setting that performs best (in terms of BER) is selected as a data point in the ideal data TPI profile, this process is repeated at many different radial locations to establish data pints across the radius of the disk to there by define the ideal data TPI profile. The work wherein the bit error rate may be measured at a few different radial locations (e.g., two inner diameter locations, a middle location, and two outer diameter locations). The geometry of the head may then be estimated in response to the data TPI settings that minimize the bit error rate at the different radial locations. For examples, a width, bevel angle, and pole height of a write element may be determined in response to the data TPI settings determined at the different radial locations. The ideal TPI profile over the entire radius of the disk is then determined in response to estimated geometry of the head (e.g., using a lookup table indexed by the estimated head geometry).



TPI Format	Data Cluster	KTPI (210*data cluster/128)
1	128	210.00
2	124	203.44
3	120	196.88
4	116	190.31
5	112	183.75
6	108	177.19
7	104	170.63

Fig. 2.3 An ideal data tracks per inch (TPI) profile is determined for the disk.

2.3 Intelligent Feature Extraction in HDD

Recently, there are many researches on intelligent feature extraction techniques to several HDD aspects. Such as servo systems [10], [11], read/write channel [12], mechanical [13] and defect classification [14], [15].

The study techniques like Genetic Algorithm, Data mining (Neural Networks, Self-Organizing Map (SOM) and Bayesian network), and Multiple Classifications Methods (Neural Net, Bayesian, Support Vector Machine and Discriminant Analysis) and Fusion technique obtain in this section of literature review show the improvement scheme of statistical analysis integration in HDD.

The search [10] of optimizing the performance of Hard Disk Drive Motion Control System by using a Multiobjective Genetic Algorithm, in developing a hard disk drive servo system, there are multiple design control objectives to be met simultaneously. While some of the objectives are constraint objectives, others are simply optimization objectives. In this paper, we investigate a new multiobjective genetic algorithm (GA) for tuning servo controller parameters such that the resultant performance can satisfy all the desired requirements. Unlike the conventional single-objective GA, this new approach is able to place higher priority on the constraint objectives than the optimization objectives. Experimental results that are based on a commercial hard disk drive are presented.

The study [11] of Neural-Networks-Based Adaptive Disturbance Rejection Method to the Control of Hard Disk Drives, this paper presents a neural-networks-based disturbance rejection adaptive scheme for dealing with repeatable and nonrepeatable runout simultaneously. The effectiveness of this method is demonstrated empirically on a commercial hard disk drive where the adaptive disturbance rejector is added to a baseline linear time-invariant (LTI) controller. The adaptive scheme can be broken into two subsystems: one subsystem is designed to suppress the repeatable runout (RRO) and the other to attenuate the residual disturbance and nonrepeatable runout (NRRO) by the use of radial basis functions. Two different methods for RRO suppression are employed in conjunction with the neural-networks-based NRRO rejector. The first one is an adaptive feedforward disturbance rejection scheme. The second is a repetitive controller. In both cases the neural modeled disturbance rejector is adapted online further increasing the track-following performance by as much as 6.4%. Experimental results of the schemes at various locations of the hard drive are included to demonstrate the general applicability of the approach on commercial drives. The total reduction of the error during track-following is measured to be as much as 25.4% respect to the baseline LTI controller

The study in [12] using Zero-Forcing preprocessing for compact Neural Network Detector in Hard-Disk Drive, the algorithm and architecture of a compact neural network based partial response maximum likelihood (PRML) detector with zero-forcing preprocessing for hard disk drive reading channels are discussed. The data preprocessing technique to improve the performance and simulation result are also presented.

The other study [13] on Data Mining for the Investigation of Unsteady Flow Field in a Hard Disk Drive, This study was performed to examine the mechanism of flow-induced vibration (FIV) in a hard disk drive (HDD). For this purpose, data mining using self-organizing map (SOM) and Bayesian network was applied to unsteady computational fluid dynamics (CFD) simulation data for a hard disk drive. The present data mining started from the extraction of temporal indices from the time series data of fluid properties given at each grid point. Then, a set of grid points was divided into several clusters based on the similarity of the temporal indices by using SOM, and the clustered data were mapped onto a real space of HDD. Through this process, characteristic phenomena latent in the unsteady flow field were classified and identified. Finally, the relations between temporal indices and FIV were constructed by using Bayesian network. The resulting network structure revealed a possible mechanism of FIV that originates from a temporal sequence of flow energy dissipation and production.

The Neural Networks for Disturbance and Friction Compensation in Hard Disk Drives, In [14], show that the tracking performance of a hard disk drive actuator can be improved by using two adaptive neural networks, each of which is tailored for a specific task. The first neural network utilizes accelerometer signal to detect external vibrations, and compensates for its effect on hard disk drive position via feed forward action. In particular, no information on the plant, sensor and disturbance dynamics is needed in the design of this neural network disturbance compensator. The second neural network, designed to compensate for the pivot friction, uses a signal activation function to introduce nonlinearities inherent to pivot friction, thus reducing the neural network's burden of expectation. The stability of the proposed scheme is analysed by the Lyapunov criterion. Simulation results show that the tracking performance of the hard disk drives can be improved significantly with the use of both neural networks compared to the case without compensation, or when only one of the networks is activated.

The study of Enhanced Failure Mode Prediction for Hard Disk Drive using Neural Network Rank-Level Fusion via Negative Correlation [15], an accurate model for predicting customer failure modes in Hard Disk Drive (HDD) has been proposed in our previous works. In this literature, the outputs of four different classifiers, Neural Networks, Discriminant Analysis, Bayesian Networks and Support Vector Machines are computed through Negative Correlation (NC) to find the most negatively correlated for all classes (while a fusion method achieves an accurate prediction, the complexity of deciding what classifiers to combine exists). It is very likely that all combinations have to be evaluated before the best predictor can be constructed. Using NC, the best classifiers suited to a specified outcome of a two class question are determined. Through utilizing NC, the resultant prediction method reduces time, complexity and improves accuracy significantly over the prior results in beyond current known reliability predictors of HDD failures.

The study of Rank-Level Fusion with Neural Network in Classification of Media Defect Patterns [16], the reduction of defects on the media surface is one of the challenges of yield enhancement in Hard Disk Drive (HDD) manufacturing. The shape classification of defects usually provides a clue to possible root causes. Nowadays, defects are visually identified by experienced inspectors. As such, a limited sample size of the defective units can be examined. Thus, trend studies and analysis become difficulty. This paper presents a statistical feature extraction technique with rank-level fusion based on neural networks to improve recognition and classification of media defects. Furthermore, the result of this study can then be used for on-line and automatic classification of media defects. Consequently, possible causes can be quickly extrapolated for purpose of diagnosis and design improvement. Three classifier techniques, discriminant function, regression trees and neural networks, are used in this proposed fusion. The experiment results demonstrated that the proposed fusion method is able to improve the classification performance significantly, which is better than both conventional fusion and individual classifier. Based on the finding, we propose a Rank-level Fusion with neural network as a based automatic recognition system to classify the Media Defect Patterns.

These results show that the NN technique provides a positive and flexible platform in solving complex, non-linear, time-varying HDD problems. The integration of the classification and fusion system has better performance than the single classification. The research also pointed out that the NN fusion improves the overall performance. Therefore, this thesis focuses on the NN technique to both classification and fusion process.

2.4 Data Mining

During the past few decades a wide range of classification data mining techniques such as [17], Clustering, Meta-Classifiers, Support Vector Machine (SVM), Artificial Neural Network (ANN), and Ensemble classifiers have been used to tackle structural class prediction problem. Among these classifiers, ensemble and SVM-based classifiers exhibited quite promising results. However, the performance of ensemble classifiers has not been adequately explored. At the same time and in parallel by exploring the impact of classification techniques, a wide range of studies tried to tackle this problem by proposing novel feature extraction methods that maintain more local and global information for this task. From the pattern recognition perspective, this problem is presented as solving a multi-class classification task. The performance of the proposed methods to solve this problem crucially relies on the selected attribute and consequently features extraction method being used as well as the classification techniques being developed.

General references regarding clustering include [18], [19], [20], [21], [22], [23], [24], 25], [26], [27], [28]. A very good introduction to contemporary data mining clustering techniques can be found in the textbook [29]. There is a close relationship between clustering techniques and many other disciplines. Clustering has always been used in statistics [29] and science [30]. The classic introduction into pattern recognition framework is given in [31]. Typical applications include speech and character recognition. Machine learning clustering algorithms were applied to image segmentation and computer vision [32]. For statistical approaches to pattern recognition see [33] and [34].

Clustering can be viewed as a density estimation problem. This is the subject of traditional multivariate statistical estimation [35]. Clustering is also widely used for data compression in image processing, which is also known as vector quantization [36]. Data 3 fitting in numerical analysis provides still another venue in data modeling [37].

Clustering in data mining was brought to life by intense developments in information retrieval and text mining [38], [39], [40] spatial database applications, for example, GIS or astronomical data, [41], [42], [43] sequence and heterogeneous data analysis [44], Web applications [45], [46], [47] DNA analysis in computational biology [48], and many others.

A number of clustering algorithms have been developed. Some deal with the specific application requirements. We briefly discuss these developments in the sub-sections Constraint-based clustering, Relation to Supervised Learning, Gradient Descent and ANN, and Evolutionary Methods.

Constraint-Based Clustering; In real-world applications customers are rarely interested in unconstrained solutions. Clusters are frequently subjected to some problem-specific limitations that make them suitable for particular business actions. Building of so conditioned cluster partitions is the subject of active research; for example, see survey [49]. The framework for the constrained-based clustering is introduced in [50].The taxonomy of clustering constraints includes constraints on individual objects (e.g., customer who recently purchased) and parameter constraints (e.g., number of clusters)

that can be addressed through preprocessing or external cluster parameters. The taxonomy also includes constraints on individual clusters that can be described in term of bounds on aggregate functions (min, avg, etc.) over each cluster. These constrains are essential, since they require a new methodology. In particular, an existential constraint is bound from below on a count of objects of a certain subset (i.e. frequent customers) in each cluster. Iterative optimization used in partitioning clustering relies on moving objects to their nearest cluster representatives. This may violate such constraint. A methodology of how to resolve this conflict is developed in [50]. The most frequent requirement is to bound number of cluster points from below. Unfortunately, K-means algorithm, which is used most frequently, sometimes provides a number of very small (in certain implementations empty) clusters. The modification of the k-means objective function and of k-means updates that incorporate lower limits on cluster volumes is suggested in [51]. This includes soft assignments of data points with coefficients subject to linear program requirements. [52] presented another modification to k-means algorithm. Their objective function corresponds to an isotropic Gaussian mixture with widths inversely proportional to numbers of points in the clusters. The result is the frequency sensitive k-means. Still another approach to building balanced clusters is to convert a task into a graph partitioning problem [53]. Important constraint-based clustering application is to cluster 2D spatial data in the presence of obstacles. Instead of regular Euclidean distance, a length of the shortest path between two points can be used as an obstacle distance.

Relation to Supervised Learning; Both Forgy.s k-means implementation and EM algorithms are iterative optimizations. Both initialize k models and then engage in a series of two-step iterations that: (1)
reassign (hard or soft) data points, (2) update a combined model. This process can be generalized to a framework relating clustering with predictive mining [54]. The model update is considered as the training of a predictive classifier based on current assignments serving as the target attributes values supervising the learning. points reassignments correspond to the forecasting using the recently trained classifier. [55] suggested another elegant connection to supervised learning. They considered binary target attribute defined as Yes on points subject to clustering, and defined as No on non-existent artificial points uniformly distributed in a whole attribute space. A decision tree classifier is applied to the full synthetic. data. Yes labeled leaves correspond to clusters of input data. The new technique CLTree (CLustering based on decision Trees) resolves the challenges of populating the input data with artificial No. points such as: (1) adding points gradually following the tree construction; (2) making this process virtual (without physical additions to input data); (3) problems with uniform distribution in higher dimensions.

Gradient Descent and Artificial Neural Networks; Soft reassignments make a lot of sense, if k-means objective function is slightly modified to incorporate, that is if it accounts for distances not only to the closest, but also to the less fit centroids: Exponential probabilities are defined based on Gaussian models. This makes the objective function differentiable with respect to means and allows application of general gradient decent method. [56] presented a detailed introduction to
this subject in the context of vector quantization. Gradient decent method in k-means is known as LKMA (Local K-Means Algorithm). Such updates are also used in a different context of artificial neural network (ANN) clustering, namely SOM (Self-Organized Map) [57]. SOM is popular in vector quantization. Bibliography related to this dynamic field can be found in the monograph [58]. We will not elaborate here about SOM except for

two important features: (1) SOM uses incremental approach. points (patterns) are processed one-by-one; (2) SOM allows to map centroids into 2D plane that provides for a straightforward visualization. In addition to SOM, other ANN developments, such as adaptive resonance theory [59], have relation to clustering. For further discussion see [60].

Evolutionary Methods; Substantial information on simulated annealing in the context of partitioning (main focus) or hierarchical clustering is accumulated. The perturbation operator used in general annealing has a simple meaning in clustering: it amounts to a relocation of a point from its current to a new randomly chosen cluster (very similar to k-means scheme). Genetic Algorithms (GA) [61] are also used in cluster analysis. An example is the GGA (Genetically Guided Algorithm) for fuzzy and hard k-means [62]. This article can be used for further references. [63] applied GA in the context of k-means objective function. A population is a set of "k-means" systems represented by grid segments instead of centroids. Every segment is described by d rules (genes), one per attribute range. The population is improved through mutation and crossover specifically devised for these rules. Unlike in normal k-means, clusters can have different size and elongation; however, shapes are restricted to segments, a far cry from density-based methods. GA were also applied to clustering of categorical data using so-called generalized entropy to define the dissimilarity [64]. Evolutionary techniques rely on certain parameters to empirically fit data and have high computational costs that limit their application in data mining. However, usage of combined strategies (e.g., generation of initial guess for k-means) has been attempted [65], [66]. Usage of GA with variable length genome to simultaneously improve k-means centroids and k itself [67] also has a merit in comparison with running multiple k-means to determine a k, since changes in k happen before full convergence is achieved.

This [68] article describes an approach to designing a distributed and modular neural classifier. This approach introduces a new hierarchical clustering which enables one to determine reliable regions in the representation space by exploiting supervised information. A Multi-Layer Perceptron is then associated to each of these detected clusters and is charged with recognizing elements of the associated cluster while rejecting all others. The obtained global classifier is constituted of a set of cooperating neural networks and is completed by a k-nearest neighbour classifier which is charged with treating elements rejected by all the neural networks. Experimental results for the handwritten digit recognition problem and comparison with neural and statistical non modular classifiers are given.

In the shape classification using Local and Global features [69], the literature study the shape classification problem by proposing a new integrating approach for shape classification that gains both local and global image representation using Histogram of Oriented Gradient (HOG). During training phase for both local and global features, we choose a finite set of random rectangular regions in the normalized and aligned object window followed by computing orientation histogram descriptors for each of them effectively. As a result, for each class, the features are constructed by a batch of the generated orientation histograms. Because the global features must be enough discriminative for each class, we then apply Adaboost to select most discriminative histogram features to learn each class classifiers. For local representation we use different scenario. For local feature extraction we use small random region on edge map of images to extract local histogram orientations just using edge pixels. Extracting local feature using edge map is inspired by the fact that the orientation histogram on non edge region is not informative enough for shape representing. Because the local histogram features are less discriminative and more common through different classes; we use a quantized set of the extracted local features to train the classifiers for each class. Given an unknown image query, using the local and global features, each local and global classifier generates a classification score. Eventually, this method uses a final classifier to integrate the tow generated local and global based scores to perform the final classification.

The study of A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus consider a network of distributed sensors [70], where each sensor takes a linear measurement of some unknown parameters, corrupted by independent Gaussian noises. We propose a simple distributed iterative scheme, based on distributed average consensus in the network, to compute the maximum-likelihood estimate of the parameters. This scheme doesn't involve explicit point-to-point message passing or routing; instead, it diffuses information across the network by updating each node's data with a weighted average of its neighbors' data (they maintain the same data structure). At each step, every node can compute a local weighted least-squares estimate, which converges to the global maximum-likelihood solution. This scheme is robust to unreliable communication links. We show that it works in a network with dynamically changing topology, provided that the infinitely occurring communication graphs are jointly connected.

The analysis and design of distributed algorithms for χ-consensus, Reference [71] presents analysis and design results for distributed consensus algorithms in multi-agent networks. We consider arbitrary consensus functions of the initial state of the network agents. Under mild smoothness assumptions, we obtain necessary and sufficient conditions characterizing any algorithm that asymptotically achieves consensus. This characterization is the building block to obtain various design results. We first identify a class of smooth functions for which one can synthesize in a systematic way distributed algorithms that achieve consensus. We apply this result to the family of weighted power mean functions, and characterize the exponential convergence properties of the resulting algorithms. We conclude with two distributed algorithms that achieve, respectively, max and min consensus in finite time.

The study of Hierarchical Decision Structure for Distributed Algorithms, the result present in [72] when the number of nodes in a distributed system becomes very large, neither a filly centralized algorithm nor a fully distributed algorithm is suitable for implementation. the number of messages exchanged in most distributed algorithms becomes too large to make them practical, while the system with a fully centralized algorithm is vulnerable to failure of the control node. The study propose another approach to constructing an algorithm for distributed systems with a large number of nodes, in which the number of messages to be exchanged would be reduced at a cost of communication delay if needed and fault tolerant capability remains. The nodes in a distributed system are partitioned into multiple groups, each of which contains multiple nodes. The process of making decision is also divided into two steps; decision in a group is made first followed by decision made based on the group decisions. We may apply different algorithm at each decision level. We examine the applicability of our approach using well-known solutions to two problems: mutual exclusion and majority consensus. As a result, we believe that this kind of flame work points another wag to the design of algorithm for distributed system.

The study on algorithms of parallel and distributed data mining calculating process, As indicated in [73] based on distributed data mining, a kind of parallel and distributed calculating architecture that store partition data information into sub-nodes is introduced by using a thought of partition database and improved Apriori algorithms. It lays emphasis on the data skew in the distributed environment. A converse clustering method is proposed to solve the data skew problem. The corresponding algorithms of parallel and distributed data mining are designed based on the large-scale transaction database. Calculating processes of these algorithms are described in detail. As the parallel and distributed data are processed after effective partition, the transmitted data size is greatly reduced through efficient communication among nodes. The proposed algorithms provide a flexible and extended calculation platform, reduce overhead traffic, and keep a favorable expansibility. The proposed algorithms aim at performing network calculation and finding advantages of network calculation by using a fairly cheap computer. The proposed algorithms can be applied to large parallel or distributed single computer environment.

With these major topics of literature review in above sections, we found a strong relationship of statistical analysis can then be applied the develop feature extraction technique to HDD testing process as well as other relate work that usually deal with data mining protocol. We greatly study and adapt those very useful features to our thesis on HDD track pitch calibration process.

Chapter3

Feature Extraction

This chapter study the classification algorithm and introduces the learning task of classification model that expect to overcome the drawback of today conventional model. Firstly we design the model by select the features that can be obtaining suitable characteristic to the data structure. K-mean clustering is use in pre-classification level and then NN is use in classification and fusion level. These methods are commonly use today in real world data mining application and then apply to industrial development.

The main propose in this study is focus on classification level that we construct the new model call distributed NN classifier. This model is designed and aims to allow the NN deal with distinguished input parameters in to sub-class of track pitch calibration while conventional NN classifier is determine overall data.

3.1 Pre-Processing

In the pre-processing step, the Empirical data are collected from the known track pitch calibration parameters that perform good correlation by using the significant as shown in Fig. 3.1. The 45 inputs parameters from the numerous affecting track pitch calibration parameters are selected and obtained for the Classification model.

Firstly we select key parameters which are related to track pitch calibration and optimization such as BER, write current, overshoot amplitude, overshoot duration, dynamic fly height, Writer width and Reader Width, signal to noise ratio and overwrite. There are 9 classes of track pitch and each track pitch recorded all key parameters for every step when performing the calibration start from class 1 to class 9. The wider track pitch has higher BER loss after calibration comparing to narrow track pitch, consequently others optimization parameters will saturate according to maximum requirement.

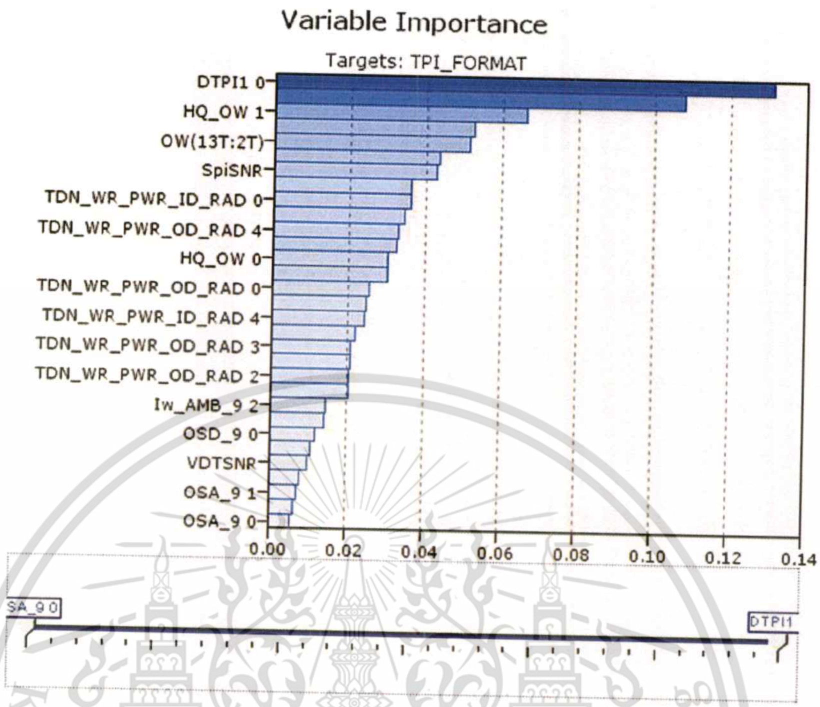


Fig. 3.1 Key input parameters obtain from Significant Test.

The total of 7483 samples covering 9 possible outcomes of the current track pitch calibration method was used. The samples went through classification model and these samples were selected from unequal samples size data set and split at random in a 80% training set and a 20% test set. The selection of data for training set and testing set is not in the scope of this study; we allowed the software to partition the data automatically using its default method.

3.2 Classification of Data

In the classification, the Clustering is added to classify the class of track pitch into subgroup consists of classes that are similar between themselves and dissimilar to class of other subgroup prior ensemble with the NN classification step.

The search for cluster is unsupervised learning of a hidden data concept. Data modeling puts clustering in a historical perspective rooted in mathematics, statistics, and numerical analysis. From a machine learning perspective clusters correspond to hidden patterns and the resulting system represents a data concept. Clustering is a division of data into groups of similar objects. It represents many data objects by few clusters, and hence, it models data by its clusters, necessarily loses certain fine details, but achieves simplification.

There is a close relationship between clustering techniques and many other disciplines. Clustering has always been used in statistics and science. There are many introductions into pattern recognition framework. Typical applications include speech and character recognition. Clustering machine learning algorithms were applied to image segmentation and computer vision which is also widely used for data compression in image processing; Clustering can be viewed as a density estimation problem. This is the subject of traditional multivariate statistical estimation.

3.1.1 Classification of Clustering Algorithms

Corresponding terms of clustering algorithms are explained further and in reality they are overlap; *Hierarchical Methods* (Agglomerative Algorithms, Divisive Algorithms), *Partitioning Methods* (Relocation Algorithms, Probabilistic Clustering, K-medoids Methods, K-means Methods and Density-Based Algorithms), *Grid-Based Methods Methods Based on Co-Occurrence of Categorical Data, Constraint-Based Clustering, Clustering Algorithms Used in Machine Learning* (Gradient Descent and Artificial Neural Networks and Evolutionary Methods), *Scalable Clustering Algorithms* and *Algorithms For High Dimensional Data* (Subspace Clustering, Projection Techniques and Co-Clustering Techniques)

3.1.2 K-Means Method

The K-Means algorithm is proposed in this study that suit for unequal input parameters. This tool is popular used in scientific and industrial applications. The name comes from representing each of k clusters C_j by the mean (or weighted average) c_j of its points, the so-called centroid which has good geometric and statistical sense for numerical attributes while this obviously does not work well with categorical attributes. The sum of discrepancies between a point and its centroid expressed through appropriate distance is used as the objective function.

The sum of the squares of errors can be rationalized as (a negative of) log-likelihood for normally distributed mixture model and is widely used in statistics. Therefore, K- means algorithm can be derived from general probabilistic framework. Note that only means are estimated. A simple modification would normalize individual errors by cluster radii (cluster standard deviation), which makes a lot of sense when clusters have different dispersions. An objective function based on $-$ L_2 norm has many unique algebraic properties. For example, it coincides with pair-wise errors and with the difference between the total data variance and the inter-cluster variance. Therefore, the cluster separation is achieved simultaneously with the cluster tightness. Two versions of k-means iterative optimization are known. The first version is similar to EM algorithm and consists of two-step major iterations that (1) reassign all the points to their nearest centroids, and (2) recomputed centroids of newly assembled groups. Iterations continue until a stopping criterion is achieved (for example, no reassignments happen).

The second (classic in iterative optimization) version of k-means iterative optimization in Fig. 3.2 shown reassigns points based on more detailed analysis of effects on the objective function caused by moving a point from its current cluster to a potentially new one. If a move has a positive effect, the point is relocated and the two centroids are recomputed. It is not clear that this version is computationally feasible, because the outlined analysis requires an inner loop over all member points of involved clusters affected by centroids shifts. However, in case it is known that all computations can be algebraically reduced to simply computing a single distance. Therefore, in this case both versions have the same computational complexity.

The wide popularity of k-means algorithm is well deserved. It is simple, straightforward, and is based on the firm foundation of analysis of variances. The k-means algorithm also suffers from all the usual suspects; The result strongly depends on the initial guess of centroids (or assignments), Computed local optimum is known to be a far from the global one, It is not obvious what is a good k to use, The process is sensitive with respect to outliers, The algorithm lacks scalability, Only numerical attributes are covered and resulting clusters can be unbalanced.

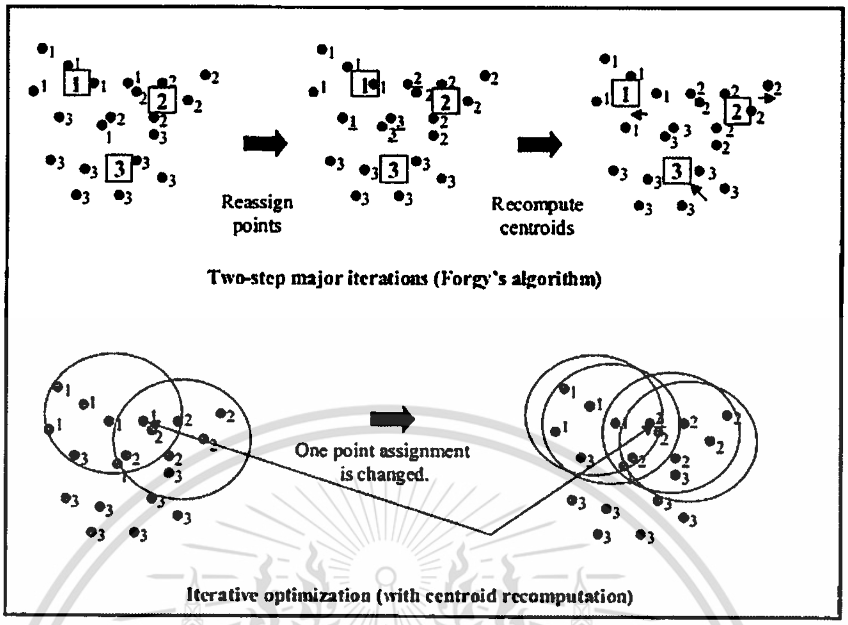


Fig. 3.2 K-means iterative optimization.

In addition to accelerating the iterative process it tries to incorporate a search for the best k in the process itself. While more comprehensive criteria discussed in How Many Clusters? require running independent k-means and then comparing the results (costly experimentation), X-means tries to split a part of already constructed cluster based on outcome of best in class criterion. This gives a much better initial guess for the next iteration and covers a user specified range of admissible k.

The tremendous popularity of k-means algorithm has brought to life many other extensions and modifications. Mahalanobis distance can be used to cover hyperellipsoidal clusters. Maximum of intra-cluster variances, instead of the sum, can serve as an objective function.

In this study we used K-Mean to divide the samples into subgroup base on assigned number of clusters, we assigned from 2 to 9 clusters according to the number of track pitch that we need to classify. Fig.3.3 Shown the groups of cluster after samples were tested (3 and 9). The K-Mean return the cluster distance and number which are very useful information that we can use as another input parameters for NN Classification.

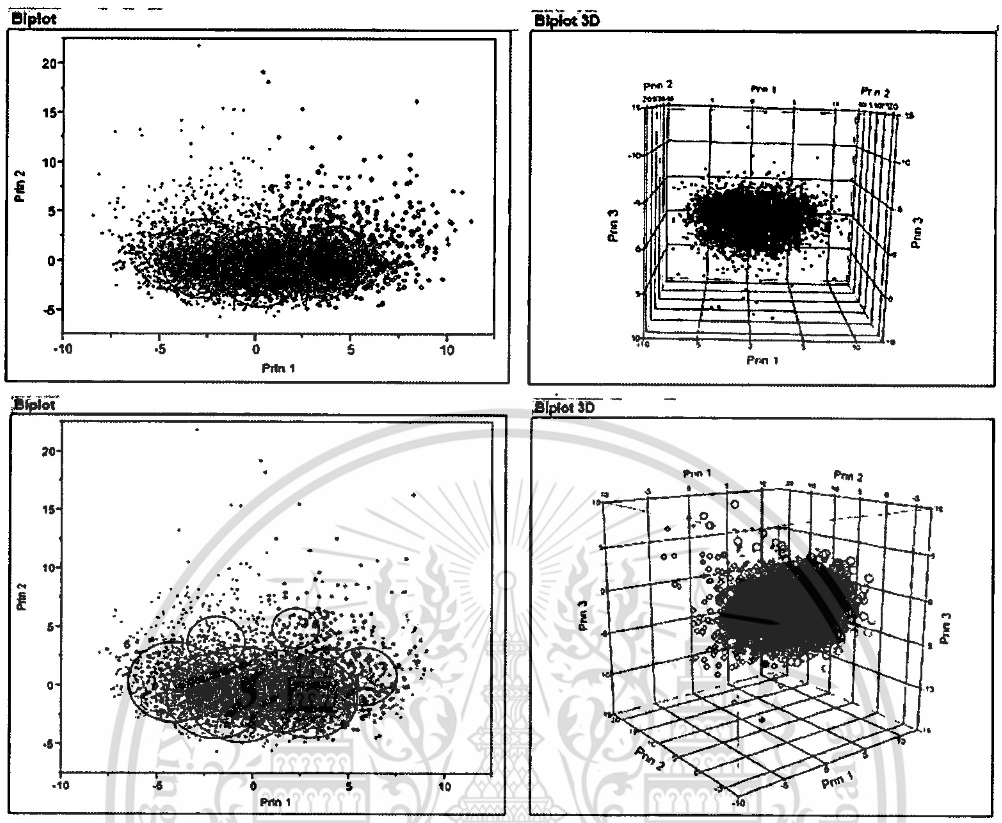


Fig. 3.3 Biplot 2D and 3D for K-Mean Cluster.

3.3 Classification of Neural Networks Algorithms

Neural network is simulating the behavior of biological neural networks. By computer, the idea is that the computer has the intelligence to learn that the man has to learn, cultivable and can apply knowledge and skills to solve various problems, many researchers have developed a neural network to be applied widely. Applications of artificial neural networks including the control of aerospace, automotive management. Banking, the military etc.

Neural networks have a history dating back to the year 1943 McCuloch and Pitts of the University of Chicago, USA, presented the paper "Boolean brain", which became the origin of the mathematical models of neural systems. The researchers have developed a neural network model of various types, and include how to send data over networks and learning networks. Each method has different complexity.

Neural networks consist of processors, called neurons (Neuron) every neuron can have input, one or multiple output and all output will be extracted to the input of other neurons within the network. Communication between neurons is a common feature of all other inputs to determine the weight of the contact force and assist in the decision. The neurons in the network are fixed. However, some networks may be able to customize the configuration of the network or neuron can be adjusted by itself at this point to the ability to learn and self-adjustment of the neural networks.

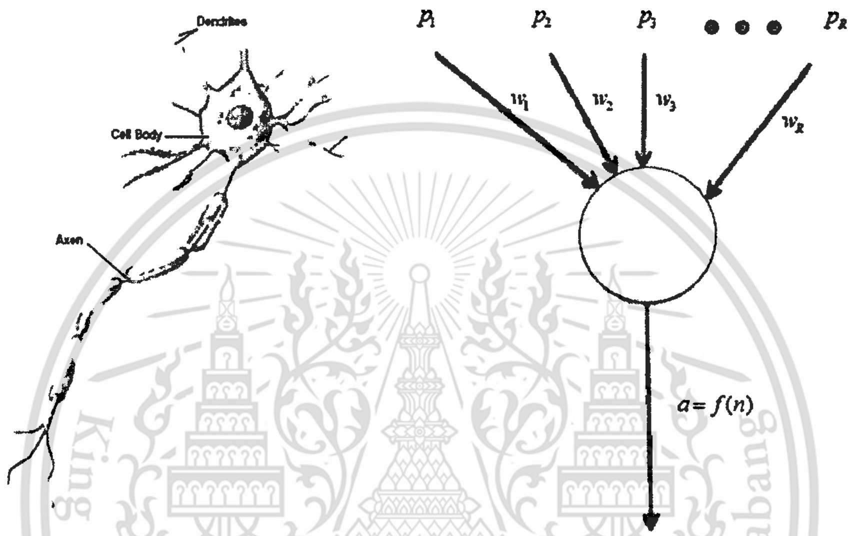


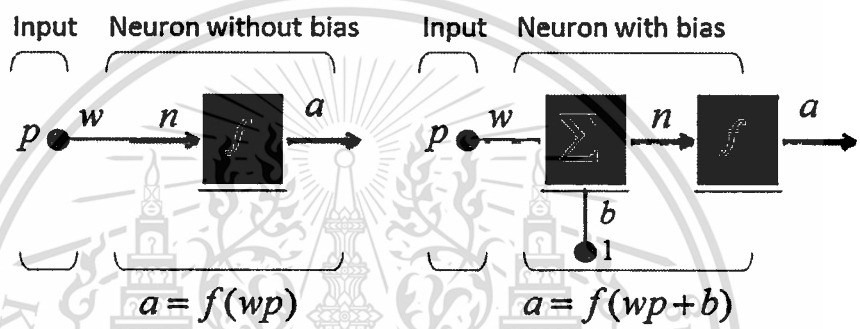
Fig. 3.4 The biological neural networks and artificial neural networks

The structure of neural networks it is similar to the biological neural networks as shown in Fig. 3.4. In general, they consist of subsets, called neurons which are a biological neuron. Neuron branches are used to transmit multiple data fields, depending on the design. That the Synapse is biological. That acts as a gateway to information or signals can be transmitted from one neuron to another neuron. That the Synapse is the weight of the body that Synapse Weight (W) on the incoming data. The data must be multiplied by the weight of the Synapse to get information, for example, the weight may be derived from learning the Delta Learning Rule that is a comparison between the actual and the target. Then the error into the equation to determine the weight to this.

3.3.1 The Basic Neural Networks

The nature and function of nerve cells or neurons as mentioned above, were introduced to the mathematical theory and modeling work in the background using a neural network.

$$n = wp \tag{3.1}$$
$$a = f(n) = f(wp) \tag{3.2}$$



(a.) Without bias. (b.) With bias

Fig. 3.5 The basic neural networks.

Neural networks are simplified input, a scalar input. Without bias of the input scalar $\boldsymbol{p}$ is entered through point and multiplied by the scalar weighted ($\boldsymbol{w}$) and the result, a scalar $\boldsymbol{wp}$ is the input to be weighted ($\boldsymbol{wp}$) be forwarded to the transfer function ($\boldsymbol{f}$) which is the scalars output ($\boldsymbol{a}$) as shown in Fig. 3.5. (a.) The basic neural network without bias and (b.) The basic neural network within bias The input is a scalar value of the output can be calculated from the equation (3.4)

$$n = wp + b \tag{3.3}$$
$$a = f(n) = f(wp + b) \tag{3.4}$$

Where $\boldsymbol{f}$ is the transfer function. There are various types of transfer functions. The example of step function and sigmoid function so that the transfer function to receive input $\boldsymbol{n}$ ($\boldsymbol{n = wp}$ or $\boldsymbol{n = wp + b}$) for the output of $\boldsymbol{a}$ ($\boldsymbol{a = f(n)}$)

The Concept of neural network is weight (w) and bias (b) can adjusted for behave as we wish. Example remembering what had been learned before. Therefore we can teach to work as we want by adjusting the parameters of weight and bias of neural networks. Sometimes, artificial neural networks adjust the parameters to get what they expect to automatically

Neural networks with multiple inputs in the form of vector $\boldsymbol{p} = [p_1\, p_2\, p_3 \dots .\, p_R]^T$ have $\boldsymbol{R}$ inputs; each input is multiplied by a weight vector $\boldsymbol{W} = [w_{11}\, w_{12}\, w_{13} \dots .\, w_{1R}]$ Transfer function is then supplied to the output shown in Fig. 3.6 and equation is as follows.

$$n = \boldsymbol{Wp} + b = w_{11}p_1 + w_{12}p_2 + w_{13}p_3 \dots . w_{1R}p_R + b \tag{3.5}$$

$$a = f(n) = f(\boldsymbol{Wp} + b) \tag{3.6}$$

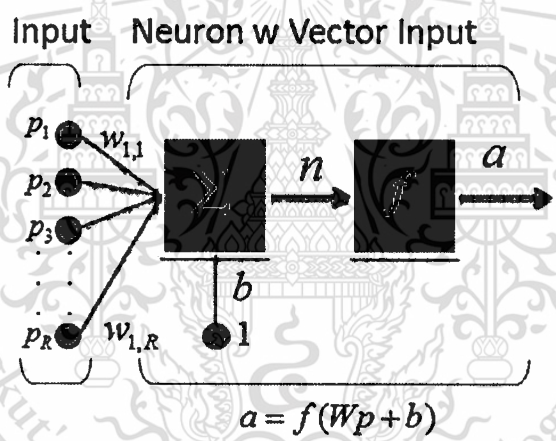


Fig. 3.6 The neural network unit with multiple inputs.

3.3.2 Structure of Neural Networks

Neural networks with only a single neuron may not be enough to solve complex problems. Simple method to fix the problem by using neurons in parallel or layers. Each type has different features as follows.

3.3.2.1 Single Layers ANNs

The General system will have more than one variable. Neural networks are a multi input and multi output for applied to multiple variables in Fig. 3.7 shown the Single Layer ANNs.

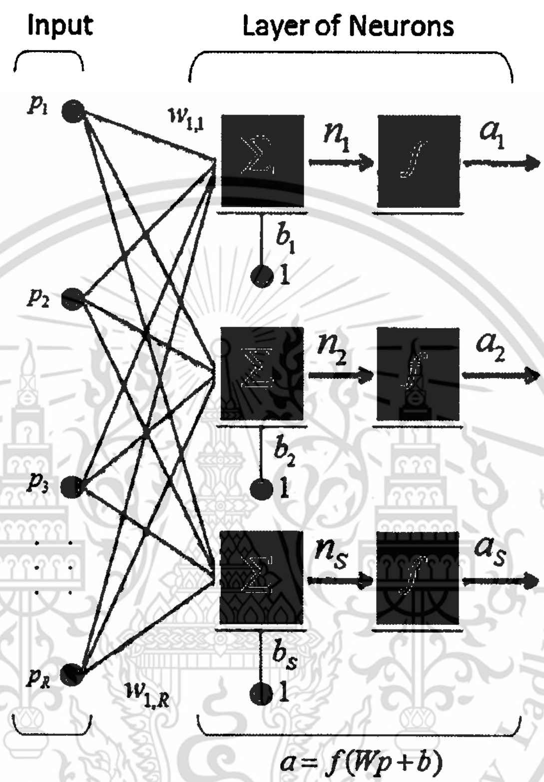


Fig. 3.7 Single Layer ANNs.

From equation (3.6) values can be calculated from the equation. (3.7) to (3.10)

$$n_1 = w_{11}p_1 + w_{12}p_2 + w_{13}p_3 \ .... w_{1R}p_R + b_1 \tag{3.7}$$

$$n_2 = w_{21}p_1 + w_{22}p_2 + w_{23}p_3 \ .... w_{2R}p_R + b_2 \tag{3.8}$$

$$n_3 = w_{31}p_1 + w_{32}p_2 + w_{33}p_3 \ .... w_{3R}p_R + b_3 \tag{3.9}$$

$$n_S = w_{S1}p_1 + w_{S2}p_2 + w_{S3}p_3 \ .... w_{SR}p_R + b_S \tag{3.10}$$

And the output can be obtained from the equation (3.11) to (3.14)

$$a_1 = f(n_1) = f(w_{11}p_1 + w_{12}p_2 + w_{13}p_3 \ .... w_{1R}p_R + b_1) \tag{3.11}$$

$$a_2 = f(n_2) = f(w_{21}p_1 + w_{22}p_2 + w_{23}p_3 \,....\, w_{2R}p_R + b_2) \quad (3.12)$$
$$a_3 = f(n_3) = f(w_{31}p_1 + w_{32}p_2 + w_{33}p_3 \,....\, w_{3R}p_R + b_3) \quad (3.13)$$
$$a_S = f(n_S) = f(w_{S1}p_1 + w_{S2}p_2 + w_{S3}p_3 \,....\, w_{SR}p_R + b_S) \quad (3.14)$$

Fig. 3.8 Neural Network represent a class of multiple inputs and multiple outputs a vector matrix. Where $\boldsymbol{p}$ is a_n input vector size $\boldsymbol{R} \times 1$ which $\boldsymbol{W}$ the weight matrix size $S \times R$ therefore $\boldsymbol{b}$ are the bias matrix have size $S \times 1$ and $\boldsymbol{n}$ are the summation matrix between $\boldsymbol{Wp}$ and $\boldsymbol{b}$. The transfer function matrix is $\boldsymbol{f}$ and $\boldsymbol{a}$ is the output matrix. The $\boldsymbol{R}$ is the scalar value for quantity of input number and $\boldsymbol{S}$ is the scalar value of quantity of neural output.

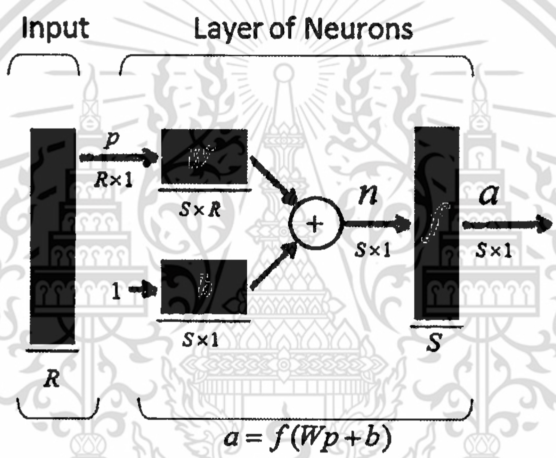


Fig. 3.8 Artificial neural network single-layer to simple model.

Where

$$\boldsymbol{p} = [p_1\, p_2\, p_3 \,....\, p_R]^T \quad (3.15)$$

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & \cdots & w_{1R} \\ w_{21} & w_{22} & w_{23} & \cdots & w_{2R} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ w_{S1} & w_{S2} & w_{S3} & \cdots & w_{SR} \end{bmatrix} \quad (3.16)$$

$$\boldsymbol{b} = [b_1\, b_2\, b_3 \,....\, b_S] \quad (3.17)$$

Therefore Fig. 3.8 the value can be calculated using the following equation.

$$\boldsymbol{n} = \boldsymbol{Wp} + \boldsymbol{b} \quad (3.18)$$

$$a = f(n) = f(Wp + b) \quad (3.19)$$

Fig. 3.9 Shown the neural network in MATLAB Toolbox by $\boldsymbol{IW}$ are input weight. The connection from one neuron to other neurons. For number 1 are the number of layer in layer 1.

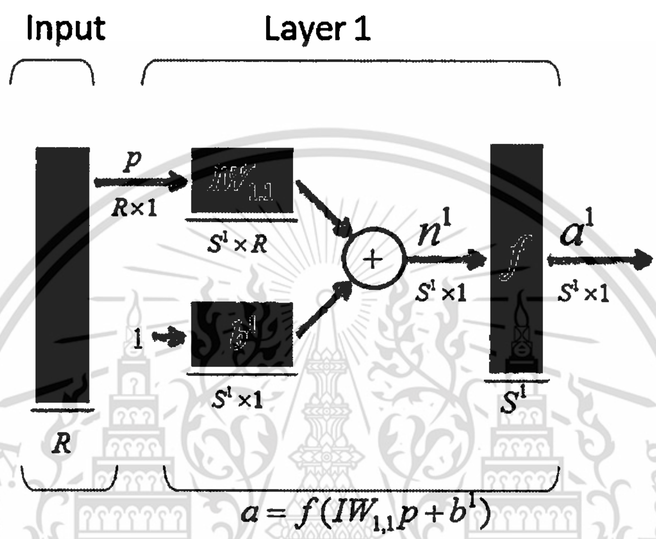


Fig. 3.9 Artificial neural network single-layer MATLAB Toolbox.

3.3.2.2 Multi-Layer Multi-Neuron ANNs

Multi-layer neural networks are most commonly used. This can be a very complex subject that can be applied to almost all types. Provided that the number of layers and the number of neurons, Fig. 3.10 shown the neural network 3 layers and Fig. 3.11 shown the neural network 3 layers in matrix form. The multi-layer neural network with a different name to the first layer is connected directly to the input we call input layer, this layer not co-location requested but input layers will pass the value to next layer. The last layer called the output layer is a output of networks. And the layers between input layer and output layer called hidden layers.

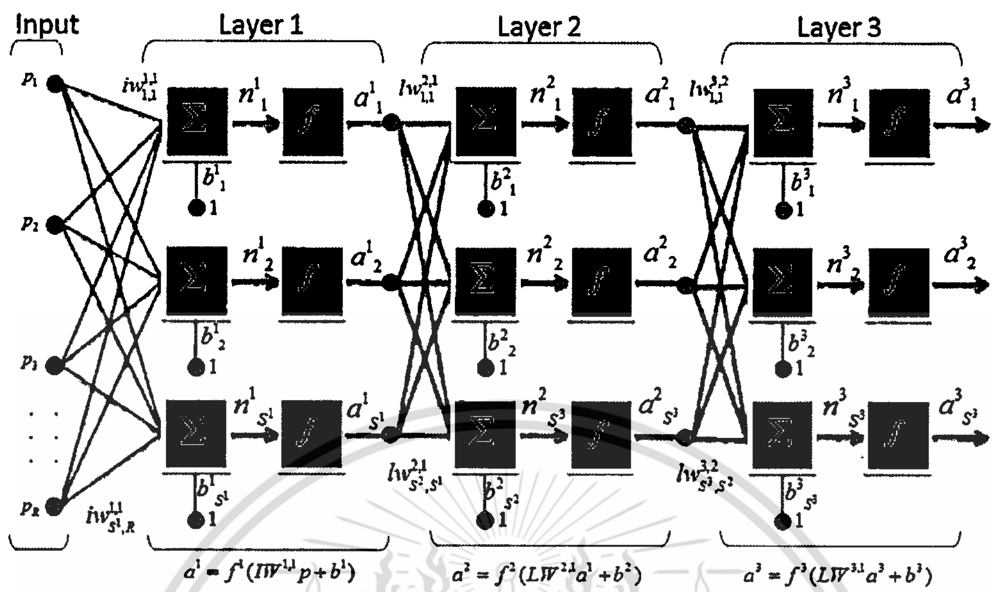


Fig. 3.10 The neural network 3layers.

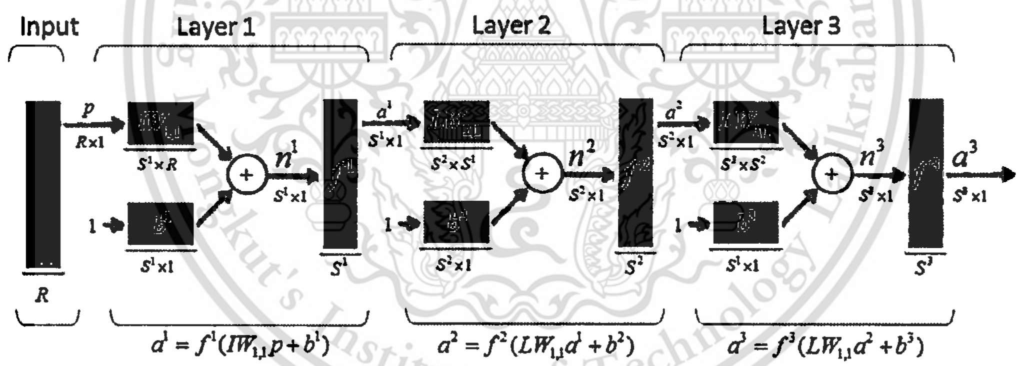


Fig. 3.11 The neural network 3 layers in matrix form.

The number of hidden layers in a network has no absolute rule. Therefore, the number of hidden layers and nodes in each layer can be obtained from the experiment.

3.3.3 Neural Networks Transfer Function

The transfer functions have many types such as Hard-limit, Linear, Logarithmic sigmoid, Hyperbolic tangent sigmoid and etc. The transfer functions of each type are as follows.

3.3.3.1 Hard-Limit Transfer Function

$$f(n) = \begin{cases} 0 & ;n < 0 \\ 1 & ;n \geq 0 \end{cases} \tag{3.20}$$

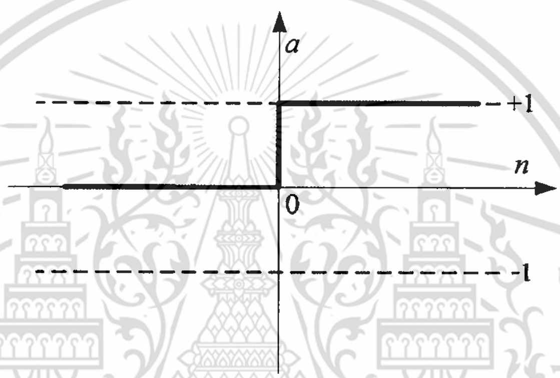


Fig. 3.12 Hard-limit transfer function.

3.3.3.2 Hard-Limit Symmetry Transfer Function

$$f(n) = \begin{cases} -1 & ; n < 0 \\ 1 & ; n \geq 0 \end{cases} \quad (3.21)$$

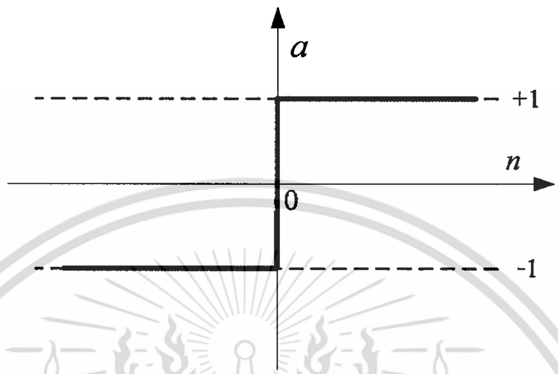


Fig. 3.13 Hard-limit symmetry transfer function.

3.3.3.3 Logarithmic Sigmoid Transfer Function

$$f(n) = \frac{1}{1 + e^{-n}} \quad (3.22)$$

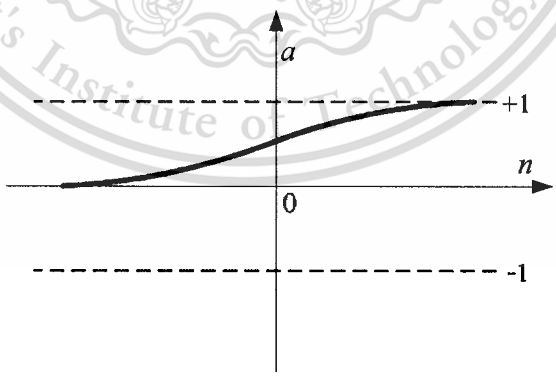


Fig. 3.14 Logarithmic sigmoid transfer function.

3.3.3.4 Positive Linear Transfer Function

$$f(n) = \begin{cases} 0 & ; n < 0 \\ n & ; n \geq 0 \end{cases} \tag{3.23}$$

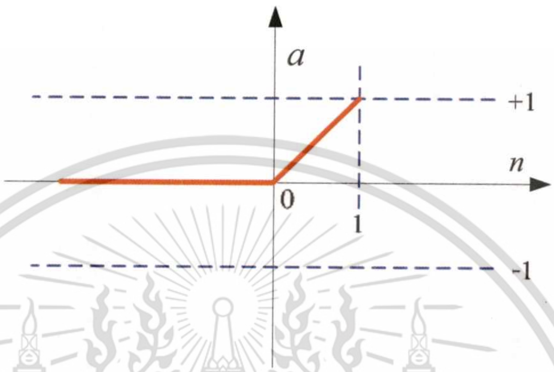


Fig. 3.15 Positive linear transfer function.

3.3.3.5 Linear Transfer Function

$$f(n) = n \tag{3.24}$$

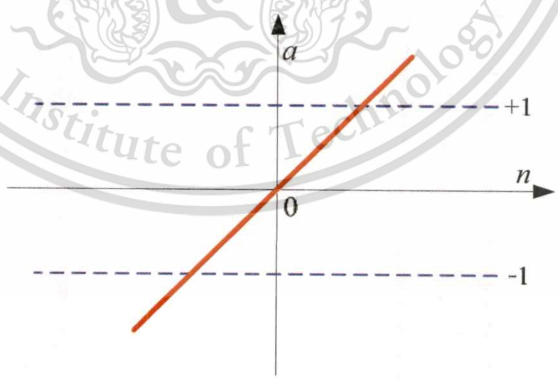


Fig. 3.16 Linear transfer function.

3.3.3.6 Sigmoid Transfer Function (Hyperbolic Tangent)

$$f(n) = \frac{e^n - e^{-n}}{e^n + e^{-n}} \tag{3.25}$$

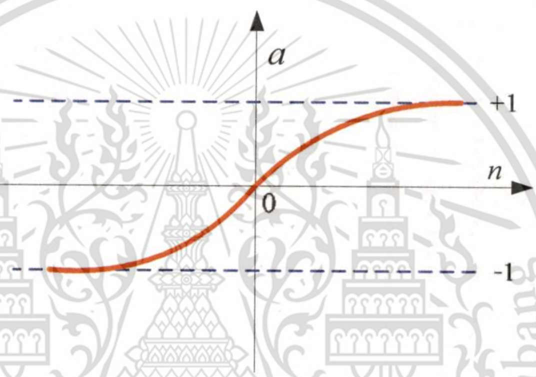


Fig. 3.17 Sigmoid transfer function.

The log sigmoid transfer function is used to solve nonlinear. The sigmoid function is sensitive to the small input signal. When the signal is only slightly positive the output is close to "1" and when the signal is only slightly negative the output is close to "0". But the signal to very positive output still value "1" and the signal is very negative output still value "0" which is the property of nonlinear gain.

3.3.4 The Teaching and Learning of Neural Networks

The initial artificial neural networks like a child. When born with brains but not improved, the children cannot do any activity. Except activity is natural called "instinct" which naturally add some features to the brain to the child improved in the womb of the mother. For the example control of breathing, feel, cry when hungry. The children are learning the process. The brains of children will be teaching to improve brain cells together to get the new features and improve the brain network.

The artificial neural networks are created like a child brain when completed, each neuron is created it does not have any features because of no appropriate weight. The training is desired behavior the weight convenient for the application. The training of neural network is the modify of the weight.

The solution of teaching neural network can be separated into two types.

Incremental Mode is update the weight after each input is fed to the trained neural networks.

Batch Mode is update the weights after all input are fed to the trained neural networks.

Artificial neural networks have 3 types of training.

3.3.4.1 Supervised Training

The training instructor will assign a set of learning with neural networks. Within a data set consisting of data input and output to training. When the input to the network will be processed as an answer. The answer from the network is being used to calculate the error is the difference between the responses of neural networks with the target of the input set. That is how much. If you are an advanced trainer, it will be adjust the weights and the training continues until the error between the responses of neural networks to answer the same target is low enough that it is acceptable to stop the training. The example of learning methods popular in training with the trainer is back propagation Learning. Learning is the set of input and output targets. By neural networks trained with the first round and compared between the output of neural network and the target value after that calculate the error of the output. The value of the output layer backward to the hidden layer to improve the weight consistent with the error.

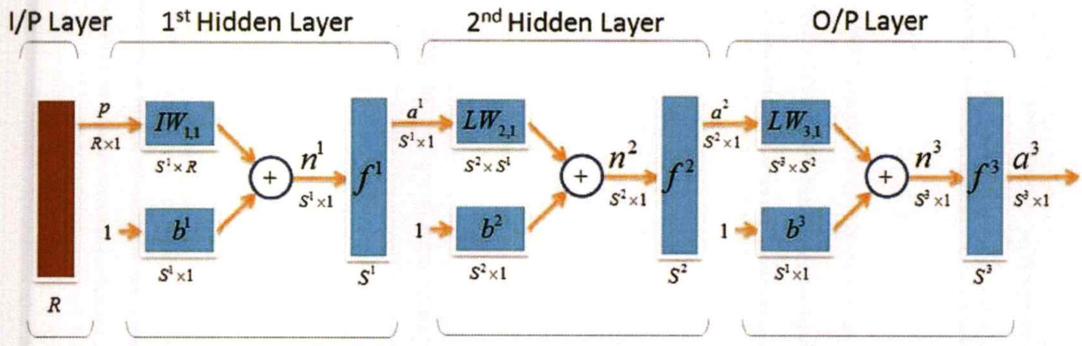


Fig. 3.18 Artificial neural network use back propagation learning rule.

Fig. 3.18 shown the artificial neural network used back propagation learning rule. The back propagation learning rule can be explained as follows.

1. When the neural network calculated the outputs. The errors determined by comparing the outputs and target values can show in equation (3.26)

$$\boldsymbol{e} = \boldsymbol{t} - \boldsymbol{a} \tag{3.26}$$

Where $\boldsymbol{e}$ is the data set of error values.

$\boldsymbol{t}$ is the data set of target values.

$\boldsymbol{a}$ is the data set outputs of neural network.

2. Found the error sensitivity each neurons by propagate the error of network to all neurons. Which sensitivity equation each layers shown in (3.27) and (3.28)

$$\boldsymbol{S^M} = -2\dot{\boldsymbol{F}}^{\boldsymbol{M}}(\boldsymbol{n^M})\boldsymbol{e} \tag{3.27}$$

$$\boldsymbol{S^m} = \dot{\boldsymbol{F}}^{\boldsymbol{m}}(\boldsymbol{n^m})\boldsymbol{W^{m+1}}\boldsymbol{S^{m+1}} \tag{3.28}$$

Where $\boldsymbol{S^M}$ is the sensitivity in output layer.

$\boldsymbol{S^m}$ is the sensitivity in hidden layers.

$\dot{\boldsymbol{F}}^{\boldsymbol{M}}(\boldsymbol{n^M})$ is invert transfer function of output layer.

$\dot{\boldsymbol{F}}^{\boldsymbol{m}}(\boldsymbol{n^m})$ is invert transfer function of hidden layers.

$\boldsymbol{W^{m+1}}$ is the weight of neuron in next layers.

3. Adjusted to the weight and bias by equation (3.29) and (3.30) respectively

$$\boldsymbol{W}_{k+1} = \boldsymbol{W}_k - \alpha s^m (\boldsymbol{a}^{m-1})^T \quad (3.29)$$

$$\boldsymbol{b}_{k+1} = \boldsymbol{b}_k - \alpha s^m \quad (3.30)$$

Where $\boldsymbol{W}_{k+1}$ is the updated weight.
$\boldsymbol{W}_k$ is the previous weight.
$\boldsymbol{b}_{k+1}$ is the updated bias.
$\boldsymbol{b}_k$ is the previous bias.
α is the learning rate.

3.3.4.2 Unsupervised Training

Unsupervised training is used only the input data set, no request of the target value to training the networks. Each neurons represent a cluster of data with similar properties. When the networks get the input values. The networks to calculate the relationships within the set of input. The adjustment use the difference between input values and weights. The advantages of this training are similar biological neural networks and not request the target values. The example of unsupervised training are Hebb learning and Kohonen learning had shown in equation (3.31) and (3.32) respectively.

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \alpha \boldsymbol{a}_{k+1} \boldsymbol{p}_{k+1}^T \quad (3.31)$$

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \alpha(\boldsymbol{p}_{k+1} - \boldsymbol{w}_k) \quad (3.32)$$

From Equation (3.31) and (3.32) Hebb and Kohonen learning rule have no target data set in the adjusted the weight. The Hebb learning rule [15] use the input and output of neuron in adjusting weight. And the Kohonen learning rule uses the difference between input and the previous weight to update the weight. The learning stops when the difference between the input values and the weights is acceptable.

3.3.4.3 Reinforce Learning

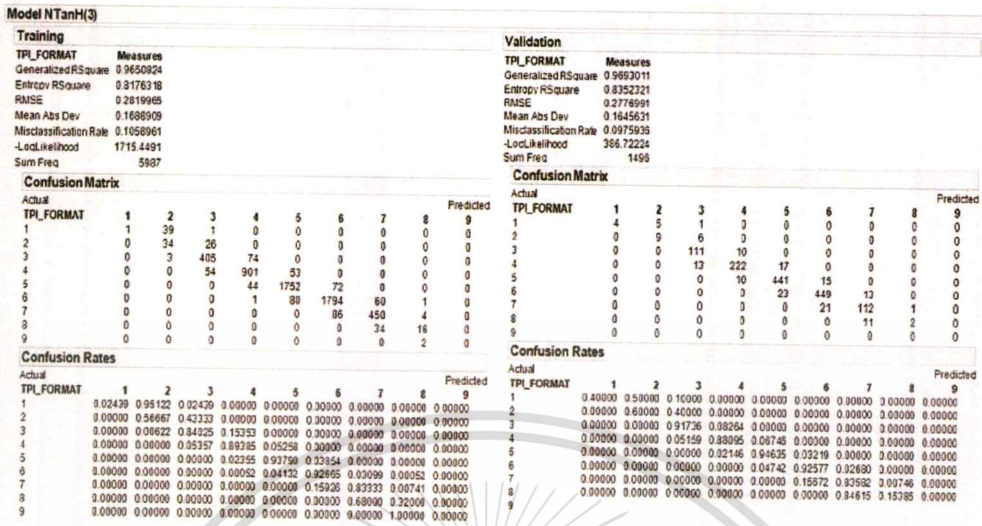
Learning is the ability to learn in the past is the principle of reward and punishment. Changing the weight is the most reward while minimize the punishment.

Reinforcement learning is similar to supervised learning, except that, instead of being provided with the correct output for each network input, the algorithm is only given a grade. The grade (or score) is a measure of the network performance over some sequence of inputs. But, this type of learning is currently much less common than supervise learning.

3.3.5 Neural Networks Classification

The neural networks are simulated biological; it has a connection to a network to solve problems with different characteristics. The neural networks can use in many applications such as Pattern recognition, Classification, Signal Processing and Control System and so on. The Neural Networks are used in this study for the classifier because of its ability to learn by examining an individual record and generating a prediction. The NN with pruning method is chosen, starting with a large network and remove the weakest units in the hidden and input layers as training proceeds. To prevent over training, we ensure that the number of samples is larger than the number of variables.

Empirical data are collected from the samples of each probable outcomes of the Track Pitch Calibration (TPI). Neural Networks classifier is proposed to use for classify or predict the track pitch calibration outcome. NN classifier contains 9 inputs, 1 hidden layer with 18 neurons and has 9 outputs at the output layer. As result show in Fig. 3.19 which is the confusion matrix between actual track pitch and prediction track pitch from NN classifier (Identify as TPI format number). Training and Validation data sets are divided automatically from the analysis software. The result is obtain percentage of matching prediction track pitch number, RSquare, RMSE, Mean Abs Dev, Misclassification Rate and LogLikelihood that we can measure the effective of NN classifier validation result after training through the model which are identical.



Model NTanH(3)

Training

TPI_FORMAT	Measures
Generalized RSquare	0.9650824
Entropy RSquare	0.8176318
RMSE	0.2819965
Mean Abs Dev	0.1686909
Misclassification Rate	0.1058961
-LogLikelihood	1715.4491
Sum Freq	5987

Confusion Matrix

Actual TPI_FORMAT	Predicted 1	2	3	4	5	6	7	8	9
1	1	39	1	0	0	0	0	0	0
2	0	34	26	0	0	0	0	0	0
3	0	3	405	74	0	0	0	0	0
4	0	0	54	901	53	0	0	0	0
5	0	0	0	44	1752	72	0	0	0
6	0	0	0	1	80	1794	60	1	0
7	0	0	0	0	0	86	450	4	0
8	0	0	0	0	0	0	34	16	0
9	0	0	0	0	0	0	0	2	0

Confusion Rates

Actual TPI_FORMAT	Predicted 1	2	3	4	5	6	7	8	9
1	0.02439	0.95122	0.02439	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.56667	0.43333	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00622	0.84025	0.15353	0.00000	0.00000	0.00000	0.00000	0.00000
4	0.00000	0.00000	0.05357	0.89385	0.05258	0.00000	0.00000	0.00000	0.00000
5	0.00000	0.00000	0.00000	0.02355	0.93790	0.03854	0.00000	0.00000	0.00000
6	0.00000	0.00000	0.00000	0.00052	0.04132	0.92665	0.03099	0.00052	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.00000	0.15926	0.83333	0.00741	0.00000
8	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.68000	0.32000	0.00000
9	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000

Validation

TPI_FORMAT	Measures
Generalized RSquare	0.9693011
Entropy RSquare	0.8352321
RMSE	0.2776991
Mean Abs Dev	0.1645631
Misclassification Rate	0.0975935
-LogLikelihood	386.72224
Sum Freq	1496

Confusion Matrix

Actual TPI_FORMAT	Predicted 1	2	3	4	5	6	7	8	9
1	4	5	1	0	0	0	0	0	0
2	0	9	6	0	0	0	0	0	0
3	0	0	111	10	0	0	0	0	0
4	0	0	13	222	17	0	0	0	0
5	0	0	0	10	441	15	0	0	0
6	0	0	0	0	23	449	13	0	0
7	0	0	0	0	0	21	112	1	0
8	0	0	0	0	0	0	11	2	0
9	0	0	0	0	0	0	0	0	0

Confusion Rates

Actual TPI_FORMAT	Predicted 1	2	3	4	5	6	7	8	9
1	0.40000	0.50000	0.10000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	0.00000	0.60000	0.40000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3	0.00000	0.00000	0.91736	0.08264	0.00000	0.00000	0.00000	0.00000	0.00000
4	0.00000	0.00000	0.05159	0.88095	0.06746	0.00000	0.00000	0.00000	0.00000
5	0.00000	0.00000	0.00000	0.02146	0.94635	0.03219	0.00000	0.00000	0.00000
6	0.00000	0.00000	0.00000	0.00000	0.04742	0.92577	0.02680	0.00000	0.00000
7	0.00000	0.00000	0.00000	0.00000	0.00000	0.15672	0.83582	0.00746	0.00000
8	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.84615	0.15385	0.00000
9									

Fig. 3.19 Neural Networks Confusion matrix.

The analysis software also obtains the surface profile graphic to show the relation or sensitivity between individual input parameters as show in Fig. 3.20.

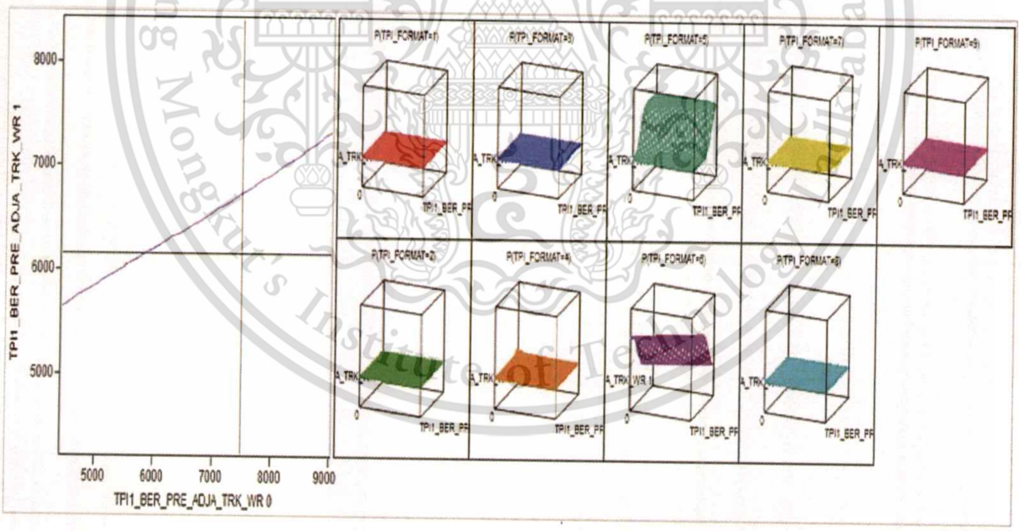


Fig. 3.20 Neural Networks Classification.

3.4 Distributed Neural Network Classification

Supervised classification is one of the most common machine learning and data mining tasks. It deals with the problem of identifying interesting regularities between a number of independent variables and a target or dependent categorical variable in a given data set. For example, given a set of training instances (xi1, xi2, ..., xik, yi), i = 1..N, the task is to compute a classifier, or model, or concept that approximates an unknown function y=f(x) that correctly labels any instance drawn from the same source as the training set.

There exist many ways to represent a classification model and many more algorithms to generate it. Typical classifier learning approaches include concept learning, neural networks, decision trees, rule learning, Bayesian learning and instance-based learning. All these approaches construct models that share the common ability to classify previously unknown instances of a domain based on instances of the same domain that were used for their training.

The output of a classifier can be i) the label of a class, ii) rankings for all the classes and iii) measures of uncertainty such as belief, confidence, probability, possibility, plausibility or other for each class.

Especially distributed classification approaches [17] tend to combine the distributed predictive models in an attempt to derive a single global model. However, there might not really exist such a single model in the distributed data, but two or more groups of models. One should not just blindly integrate local models, but should do some sort of reasoning about the relations between local models before the integration.

The way that multiple classifiers are combined is an important research issue that has been investigated in the past from the communities of statistics, pattern recognition, machine learning, knowledge discovery and data mining. When only the label of the predicted class is available, then the simplest combination method that can be used is Majority Voting, which does not require a training stage. In this case, the class that receives the most classifier predictions is the final result. Weighted Majority Voting, weights the decision of each classifier by its performance on the training data. When a measure of belief, confidence, certainty or other about the classification is available along with the class label, then a number of different rules for combining these measures have been suggested, like Sum, Min, Max, Prod and Median is an interesting study of these rules. Stacked Generalization, also known as Stacking in the literature, is a method that combines multiple classifiers by learning the way that their output.

Supervised classification task solving in the field of pattern recognition is currently well performed both by neural and statistical algorithms. Neural networks, and more particularly Multi-Layer Perceptrons (MLP). The reasons for this success essentially come from their universal approximation property, their good generalisation capabilities, which have been proved for many simple applications in recent years. Comparisons of various neural and statistical algorithms have shown that the superiority of one algorithm over another cannot be claimed. Performances strongly depend on the characteristics of the problem (number of classes, size of learning set, dimension of the representation space, etc) and on the efforts devoted to the "design phase" of the algorithms (i.e., classifier architecture determination, tuning of learning parameters, etc). Also notice that a sufficient level of classification accuracy may be reached through a reasonable design effort, and further improvements often requires increasingly expensive design phase. So, obtaining good generalisation behaviour with an MLP is not a trivial task when dealing with complex problems, since there is no reliable and generic rule currently available to determine suitable neural network architecture and this can require long trial and error research.

Moreover, neural networks also have many other defects that are well known and documented. In particular, it has been shown in that the MLP tends to draw open separation surfaces in the input data space, and thus cannot reliably reject patterns. Another drawback of the MLP is the so-called moving target problem since there is no communication between neurones on a layer, each neurone decides independently which part of the classification problem it will tackle.

To overcome these problems, several proposed to develop multi-experts decision systems. This idea is mainly justified by the need to take into account several sources of information which can be complementary in order to reach high classification accuracy and to make the decisions more reliable, and/or to facilitate the classifier design. In this way, several strategies covering most aspects including nature of experts, methods or topologies of decision combination, etc, have been reported.

Distributing a classification problem presents two main points of interest. The first one is to simplify both the design and the training of a neural network (or any other classifier) by dividing up a given task into several simpler sub-tasks. Such a simplification is expected to lead to an improvement of the generalisation capabilities and accuracy rejection trade-off over those allowed by a single classifier. The second advantage of the classification task distribution is to engineer an easy-to-update modular classifier: when new data are added in the training database, it can be expected that some modules remain unchanged, while some others will just have to be retrained or modified. Moreover, when a new sub-class appears, a new module can easily be added avoiding a complete rebuilt of the classifier.

On the other hand, sub-solutions provided by a modular classifier are usually integrated via a multi-expert decision making strategy. The multi-expert approaches based on experts cooperating differ widely (especially) both with respect to the combination decision strategy and also in the way the problem is approached. In other words, the choice of experts which correspond to the way of splitting the initial task, and/or the context definition of each expert must be considered. The expert context definition include the data representation (set of features) used by each expert, the type of classifier output ...etc. The designer should take into account all these parameters in the combination scheme in order to obtain an optimal behaviour in regards of the classification performances.

In order to obtain high performances without significantly increasing the system complexity, two ways of Multiple Classifier Systems design may be followed. First, one can consider that the chosen set of classifiers providing reasonable accuracy generates a sufficient number of uncorrelated errors. In this case, high accuracy could be reached if an efficient combination strategy is used to exploit this "complementary" behaviour. It must be noticed that such behaviour is not easy to obtain in the case of a real problem. The second way which can be followed in the design of Multiple Classifier Systems is the above mentioned approach and it consists in decomposing the classification task into simpler sub-tasks, each one being handled by a separate expert. This approach differs from the first one mainly in the sense that no potential and explicit "complementary behaviour" between experts is expected, the main goal of such a method being to specialise each expert in a particular sub-task. The task decomposition should produce sub-tasks as simple as possible allowing a simple and robust classification module. The multi-experts decision-making module, even simpler, usually has enough information to make an accurate global decision.

In the classification level we construct two different models as shown in Fig. 3.21 which are conventional NN classifier and proposed distributed NN classifier to measure the performance and compare the result between the two methods with and without NN fusion. Both methods are requiring multiple NNs in order to obtain various best output of each one to next learning in NN fusion. Conventional NN Classifier is received the input as normal distribution (determine overall data) but Distributed NN classifier is received the input by divided into sub track pitch class (determine one class of data only).

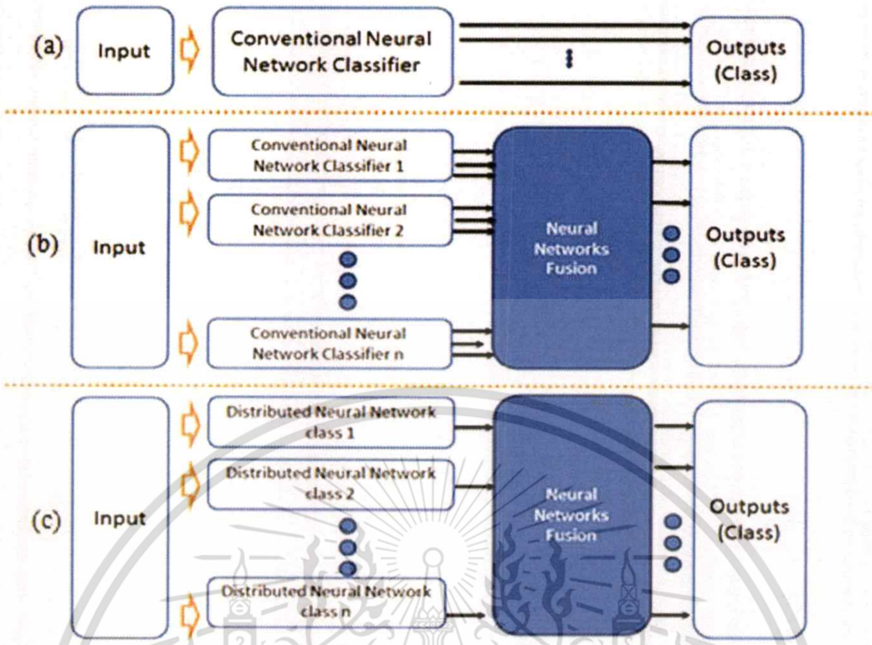


Fig. 3.21 The structure of (a) Conventional NN Classifier, (b) Conventional NN classifier with NN Fusion, (c) Distributed Classifier with NN Fusion.

In this study, a cooperative modular Neural Network, in combination with a statistical classifier, is introduced with the objective to improve the performances in terms of robustness, adaptability and accuracy-rejection trade-off over those allowed by a single classifier. In light of the above, the proposed approach tends to reach this objective by splitting the initial classification task into a simpler sub-tasks obtained by extracting the topology of the learning data set in the feature space.

The data are first split randomly into training and test group with 80% and 20% respectively by analysis software. The training data set is then proceeding through the conventional and the distributed NN classifiers.

There are 45 inputs parameters applying to the input of NN classifiers. The number of outputs of the classifiers is equivalent to the designed Product TPI. In this paper, there are 9 TPIs. Correspondingly, there are 9 outputs (classes) of the proposed NN Classifier. Since having 9 classes, the number of the distributed NN classification models obtains 9 models as well. The distributed NN classification conceptual is to perform the distributed tasks for a collective goal while the conventional NN classification performs oppositely.

Therefore, the distributed NN models are depended on the number of the classes, the more classes require the larger number of NN models, whereas the number of the conventional NN classification model does not depend on the number of the classes, only one NN model handles all classes.

With this basic concept, it improves a classification performance, capability and speed of the process, since one model is concentrated to one class or one task only.

The distributed NN Classifier works very well on both non-homogeneous and homogeneous inputs while conventional NN Classifier potentially has poorer performance when it operates on the non-homogeneous or the data of each class are not equal as shown in Fig. 3.22 which is track pitch distribution.

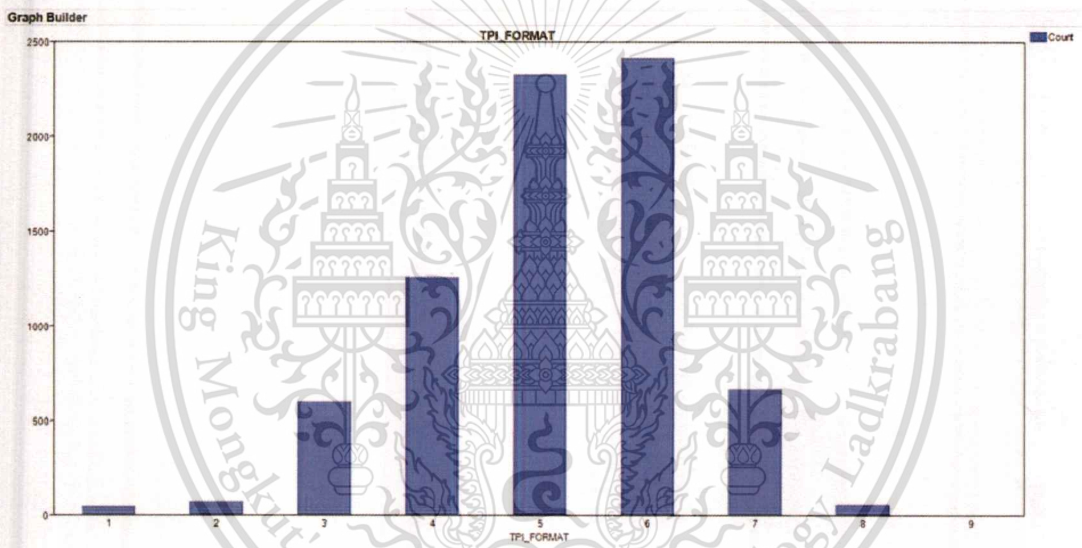


Fig. 3.22 Track pitch distribution, the sample size of each track pitch is unequal (Identify as TPI format).

In practical, there is a difficulty to have the same amount of sample size for each class as shown in Fig. 3 and becoming a challenge to the classifier system, less sample size leads to improper weight adjustment and poorer results. As mentioned earlier, the distributed NN model concentrates to one class and this helps to improve the accuracy individually since the NN model is not being shared to other classes.

Nevertheless, the distributed NN method needs fusion process additionally to fuse the output from all classes for ranking the results. Though, it is still beneficial to the overall system performance when comparing to the others models and the detail will be discuss in following.

3.5 Ensemble of K-Mean Clustering and Distributed NN Classification

The further approach proposed in this study redefines the learning task of a neural network so that a simple network building rule can lead to good generalization capabilities with an easy design phase. This redefinition follows a "divide and conquer" strategy with the objective to split the classification problem into several simpler sub-tasks. This classification task distribution is achieved while ensuring a coherency with the data topology in the representation space. The main idea behind this process is to use a "supervised hierarchical clustering" which enables one to determine reliable regions in the representation space. A specialized MLP is then associated to each detected region, and a k-Nearest Neighbour classifier is charged to treat the remaining part of the learning set (non reliable regions). Thus, the whole classifier is a set of cooperative one class neural networks (experts), and it is expected to reach a high accuracy value with simplest learning and designing phases [74].

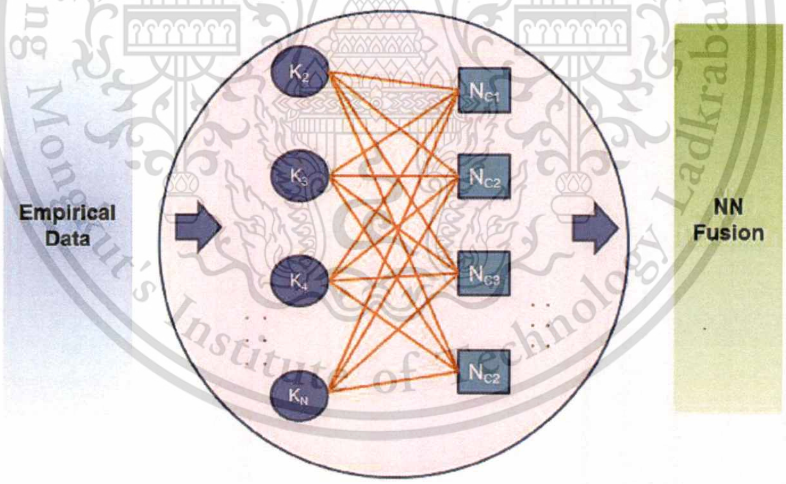


Fig. 3.23 The Ensemble model of K-Mean Clustering and Distributed NN Classifier.

Thus, we propose an unsupervised procedure which provides an automated task decomposition without any a priori knowledge. The simplest problem to be given to a neural network is probably a linearly separable one. Unfortunately, few real problems are as simple as this. A second type of simple classification problem - although more complex than the previous one - may be encountered for a two class problem such that at least one of them is constituted by an homogeneous region (i.e. a unique pure cluster containing elements of the same class only). The proposed “divide and conquer” strategy aims to identify this kind of cluster - called an "islet" - in order to provide as many tasks as islets [75], [76]. If N islets are detected, the classifier will thus be constituted from N cooperative neural networks, each of them being quite simple to configure and being expected to present good decision boundaries. In other words, the partial goal of this approach remains to obtain a classifier capable of defining a closed separation surfaces in the feature space allowing a reliable rejection behaviour. Such a behaviour is required in applications like pattern recognition

As mentioned above, the first stage towards the classification task distribution consists in capturing the data structure in the feature space. To achieve this, the problem decomposition starts with a clustering phase in order to extract reliable clusters or regions. Thus, it implies to determine the number and the constitution of the clusters in the learning data set. The most commonly used techniques are certainly Self-Organizing Maps and partitional clustering methods (k-means) which is used in contrast to hierarchical clustering, where hierarchical is in fact based on a nested sequence of partitions. The main problem with these methods is that in practice, the number of clusters is required in advance to obtain a good representation of the data.

The second step of the modular and cooperative classifier design consists in exploiting the clustering result in order to extract the data topology. This extraction is made by labelling the vectors according to their class in the supervised meaning. In this way, it is possible to compare the supervised and unsupervised information provided respectively by a vector label and the dendrogram. Afterwards, an analysis of the composition of the sub-trees reveals the presence of islets (i.e. clusters comprising at least P elements from the same class, P being user-defined and correspond to the minimum allowed size of an islet). Fig. 5 illustrates the resulting distribution after applying such a technique.

Each islet is then learnt by an MLP which has to solve a two class problem: recognize its associated islet while rejecting all other elements. Since the elements of an islet are close to one another, it can be expected that the problem is simple enough to allow a basic MLP building rule to be efficient.

3.6 Neural Network Fusion

The data fusion approach combines data from multiple sensors (and associated databases if appropriate) to achieve improved accuracies and more specific inferences that could not be achieved by the use of only a single sensor. This concept is hardly new:- living organisms have the capability to use multiple senses to learn about the environment. The brain then fuses all this available information to perform a decision task.

One of the first definitions of data fusion came form the North American Joint Directors of Laboratories (JDL), who define data fusion as a:- multilevel, multifaceted process dealing with the automatic detection, association, correlation, estimation and combination of data from single and multiple sources. Data fusion principles apply to many domains, and have been (often implicitly) at the core of modern applications in the diverse areas spanning engineering, computing, and biomedicine. The recent interest in the theory and taxonomy of multi-sensor data fusion has been reflected by a number of special issues of leading international journals and conferences.

3.6.1 Data Fusion Principle

When approaching a problem from the data fusion viewpoint, we differentiate between the following levels of abstraction:

Observation/measurement space contains vectors of measurement functions which can be univariate, multivariate, and/or multidimensional, depending on temporal, spatial or other independent variables. It may be possible to build a state-space model, or to assess the data modality.

Transform domain representations, which seek features from time and/or frequency models (fast Fourier transform (FFT), (nonlinear) autoregressive (N) ARMA models, wavelet), blind processing (independent component analysis (ICA), blind source separation (BSS)), particle/Kalman filter, kernels and support vector machines (SVM), kernel ICA);

Decision space, where the classes within the data fusion model (and the corresponding basins of attraction from the measurement space) are mapped into the relevant probabilities of the occurrence of an event.

3.6.2 Models of Data Fusion

Data fusion is based on the manipulation of multiple measurements, where classifiers operate on features extracted from the real world measurements; an overview of the ways for combining classifiers can be found in [77]. Authors distinguish between the two fusion classes in Fig. 3.23.

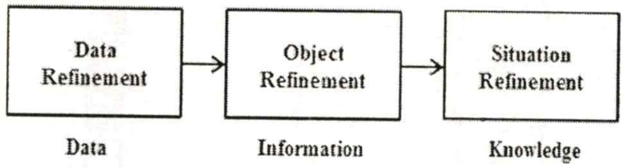


Fig. 3.24 General data fusion concept.

Data fusion, where the classifier operates on either the raw data or features extracted directly from the measurements;

Decision fusion, where the decisions from the individual classifiers for different data channels are combined.

The choice depends on the statistical relationship between the data channels, mutual entropy, or joint Gaussianity, and to this end coupling of mathematical modelling and information processing is under investigation. The main issues are signal nonlinearity (with associated non-Gaussianity), nonstationary, intermittent data natures and noises. This makes it very difficult to perform estimation by standard methods since no assumption on the data model and distribution can be ascertained. In some applications, such as functional Magnetic Resonance Imaging (fMRI), there is even no "ground truth", to rely upon. Multi-sensor practical systems therefore aim at providing higher accuracy and improved robustness against uncertainty and sensor malfunction, and also for the information extracted from different sources to be integrated into a single signal or quantity. Signal processing algorithms for "sensor" or "data fusion" can be based on

Probabilistic models: Bayesian reasoning, evidence theory, robust statistics;

Least squares: Kalman filtering, regularization, set membership;

Intelligent fusion: Fuzzy logic, neural networks, genetic algorithms.

In this thesis we use intelligent neural networks fusion.

3.6.3 Architectures and Performance Aspects

Combining multi–sensor data in the data fusion framework has the potential of faster and cheaper processing and new interfaces, together with reducing overall uncertainty (increase in reliability). Such data can be combined in various ways, for instance by:- i) linear combiner, ii) combination of posteriors (weights, model significance), iii) product of posteriors (independent information). Based on the different ways of combining information and different semantic levels, we differentiate between the following data fusion architectures, shown in Fig. 3.24.

Centralised: simple algorithms, but inflexible to sensor changes;

Hierarchical: collaborative processing, two way communication;

Decentralised: robust to sensor changes and failures, complex algorithms.

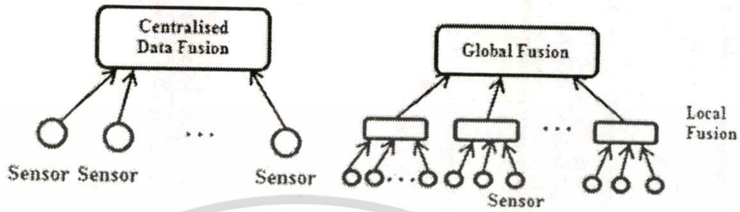


Fig. 3.25 Centralized and hierarchical data fusion.

This synergy of information fragments offers some advantages over standard algorithms, such as:-Improved confidence due to complementary and redundant information; Robustness and reliability in adverse conditions (smoke, noise, occlusion); Increased coverage in space and time; dimensionality of the data space; Better discrimination between hypotheses due to more complete information; System being operational even if one or several sensors are malfunctioning; Possible solution to the vast amount available information.

The paradigm of optimal fusion in this sense is to minimise the probability of unacceptable error, depending on the stage at which fusion takes place, data fusion is often categorized as the low– (LLF), intermediate– (ILF) or high–level (HLF) fusion, where:-

LLF (data fusion) combines raw data sources to provide better information;

ILF (feature fusion) combines features that come from heterogeneous or homogeneous raw data. The aim is to find relevant features amongst various features coming from different methods (FFT), discrete cosine transform (DCT), wavelet, delay vector variance (DVV).

HLF (decision fusion) combines decisions or confidence levels coming from several experts (hard and soft fusion).

In practice, any combination of these three levels can be employed, for instance; Data in – Data out, Data in – Feature out, Feature in – Feature out, Feature in – Decision out, Decision in - Decision out.

Classifier Fusion has been a very active field of research. It was used for improving the classification accuracy of pattern recognition systems, as single classification learning algorithms were approaching their limits. It was also used as a method for scaling up data mining to very large databases, through combining classifiers trained in parallel from different parts of the database. Finally, it was used for learning from geographically distributed databases, where bandwidth or privacy constraints forbid the gathering of data in a single place, through the fusion of locally learned classification models.

There are two general groups of Classifier Fusion methods. The first group encompasses methods that combine the outputs of the classifiers, while the second group deals with the structure of the multiple classifier system

Methods that fuse classifier outputs can be further categorized based on two properties. The first is the classifier output type on which they can operate, and the second is the need of training data for the fusion process. According to these, 0 presents the main methods.

Table 3.1 Classifier combination schemes and their characteristics.

	Re-Training	
Output	Yes	No
Label	Knowledge-Behavior Space (Huang & Suen, 1995)	Majority Voting (Lam & Suen, 1995)
Ranking	The Highest Rank Logistic Regression Intersection of Neighborhoods Union of Neighborhoods	Borda Count
Distribution	Stacked Generalization (Wolpert, 1992) Dempster-Shaffer Combination (Rogova, 1994) Fuzzy Templates (Kuncheva et al., 1995) Fuzzy Integrals (Tahani & Keller, 1990)	Sum, Product, Min, Max, Median rules (Kittler et al., 1998)

In general, the main objective of the fusion process in this study is to collect all the results from the NN classifiers and also improve the performance of the systems as shown in many researches.

The NN Fusion method is chosen since it is able to improve the system performance when comparing to conventional fusion methods such as border count fusion as shown below

$$BC = \frac{\sum_{i=1}(r_i)}{m} \tag{3.33}$$

Where m is the matchers and r_i is the racks assigned

The border count fusion usually uses the summation of the ranks assigned by individual matchers to calculate the final rank which is similar to Average calculation.

Conversely, NN is able to "learn" by adjusting the strengths (weight) and bias of these connections.

Technically, the weight of NN fusion is adaptively adjusted by the learning process via the error back propagate process as shown in equation and until they can approximate a function that computes the proper output for a given input pattern. These weight and leaning features become the key advantages to improve the NN fusion performance better than the conventional fusion.

Nevertheless, the NN fusion method requires a complex computation system and also complexity in implementation and maintenance aspect unlike the conventional fusion. However, these key drawbacks are not the key concerns for today anymore. With the current processing and resources performance of the computer, the concerns should be diminished.

Chapter4

Results and Discussion

7483 samples from 9 classes, Models constructed and validate by JMP software. Experiments performed 3 times with 80% of Training and 20% of Validating set. 45 critical parameters were selected from significant test distinguished by zone. 9 Outputs is determined base on number of classes for prediction.

Evaluates 9 Distributed NN classifiers with and without K-Mean ensemble. Ensemble with K-Mean is performed subsequently from 2 to 9 clusters. Observe the classification accuracy of each class compare to convention.

Fuse the results from 9 classifiers by use Neural Network method.

4.1 Empirical Data / Pre-Processing

In the pre-processing step, the input data are collected from the known track pitch calibration parameters that perform good correlation by using the significant test as shown in Fig 3.1. The 45 inputs parameters from the numerous affecting TPI calibration parameters are selected and obtained for the NN classifier model.

4.2 Results of Distributed NN Classification without Clustering Ensemble

In the beginning of our study we constructed our classification model as show in Fig. 4.1 to compare the performance between conventional and distributed NN classifier. NN fusion is added to the last step to compile the output from the two models and refine the classification as the last step.

Conventional NN classifier contain 10 NNs, each NNs perform the classification individually to predict all classes (Globalize) as the result show in Table IV. Distributed NN classifier contain 9 NNs, each NNs perform the classification to specific classes (Localize) independently and result show in Table V.

Multiples NNs is required for these models to signify the learning of each NNs. The two models are designed to enhance the classification accuracy from different input distribution. The Conventional NN Classifier, all NNs classify all classes. The Distributed NN Classifier, one NN classify concentrated on one class. From these models each NN learnt differently, though we return the output from the classifiers to NN fusion to learn again. NN Fusion uses that result as an input and determine prior output distribution to aggravate the final classification accuracy. The NN fusion model contains 9 inputs, 1 hidden layer with 18 neurons and has 9 outputs at the output layer.

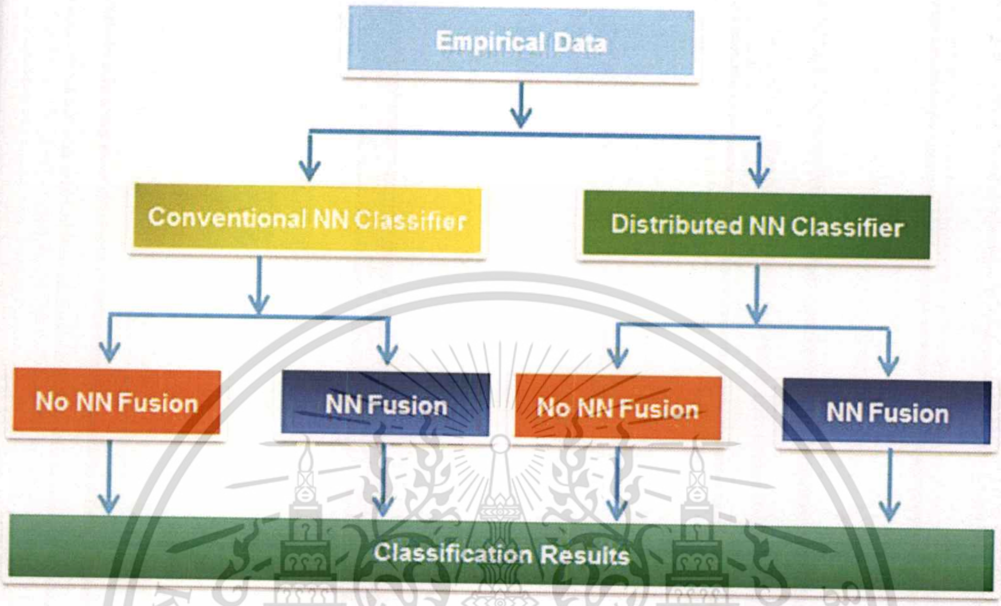


Fig. 4.1 Diagram showing model flow Conventional and Distributed NN Classification without Clustering Ensemble.

The unequal samples were tested through the assigned models for 3 trial and we summarized the classification result on each class to measure the accuracy and repeatability as show in Table II and Table III.

The result from Conventional NN classifier showing that it is performs well for the class which has high samples size but poor result for the class which has small samples. This is one of the negative sides for the models that determine the input data from overall distribution and exhibit higher error on the tail end.

Table 4.1 Classification results from Conventional NN Classifier with and without NN Fusion and non Cluster ensemble of each class.

Method	Conventional NN Classifier without NN Fusion				Conventional NN Classifier with NN Fusion			
TPI	Trial 1	Trial 2	Trial 3	Avg	Trial 1	Trial 2	Trial 3	Avg
TPI 1 = 51	0.00%	0.00%	84.31%	28.10%	94.12%	64.71%	92.16%	83.66%
TPI 2 = 75	45.33%	62.67%	53.33%	53.78%	56.00%	62.67%	42.67%	53.78%
TPI 3 = 603	85.74%	83.91%	82.59%	84.08%	85.74%	87.89%	86.57%	86.73%
TPI 4 = 1260	85.40%	88.25%	87.62%	87.09%	91.03%	90.00%	90.87%	90.63%
TPI 5 = 2334	90.83%	91.13%	91.09%	91.02%	92.93%	93.32%	93.53%	93.26%
TPI 6 = 2421	92.32%	91.82%	92.15%	92.10%	92.77%	93.56%	93.06%	93.13%
TPI 7 = 674	80.71%	78.93%	87.39%	82.34%	85.46%	83.09%	85.61%	84.72%
TPI 8 = 63	0.00%	7.94%	17.46%	8.47%	34.92%	31.75%	0.00%	22.22%
TPI 9 = 2	0.00%	0.00%	50.00%	16.67%	0.00%	50.00%	0.00%	16.67%

The Distributed NN classifier shows every class with better accuracy comparing to Conventional NN Classifier and work well with unequal samples which has a small samples size. This result proven that the problem from Conventional NN Classifier now can be addressed when the NNs do focus on the individual distribution independently.

Table 4.2 Classification results from Distributed NN Classifier with and without NN Fusion and non Cluster ensemble of each class.

Method	Distributed NN Classifier without NN Fusion				Distributed NN Classifier with NN Fusion			
TPI	Trial 1	Trial 2	Trial 3	Avg	Trial 1	Trial 2	Trial 3	Avg
TPI 1 = 51	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
TPI 2 = 75	57.30%	62.67%	56.00%	58.66%	84.00%	70.67%	72.00%	75.56%
TPI 3 = 603	89.72%	85.57%	83.25%	86.18%	85.07%	89.05%	88.39%	87.51%
TPI 4 = 1260	90.24%	90.95%	91.59%	90.93%	90.08%	91.67%	92.22%	91.32%
TPI 5 = 2334	94.82%	91.86%	92.12%	92.93%	94.30%	95.16%	93.32%	94.26%
TPI 6 = 2421	91.70%	95.29%	88.97%	91.99%	93.23%	94.30%	92.40%	93.31%
TPI 7 = 674	87.39%	83.38%	86.35%	85.71%	85.31%	84.57%	87.54%	85.81%
TPI 8 = 63	90.48%	80.95%	87.30%	86.24%	92.06%	87.30%	90.48%	89.95%
TPI 9 = 2	100.00%	100.00%	100.00%	100.00%	100.00%	0.00%	0.00%	33.33%

4.3 Results of Distributed NN Classification with Clustering Ensemble

After understanding the result from Conventional and Distributed NN Classifier, we study additional structure by adding the Cluster to the model to do pre classifies or grouping the samples prior input the data to NN. We use K-Means clustering method in this study; the k-means method first selects a set of n points called cluster seeds as a first guess of the means of the clusters. Each observation is assigned to the nearest seed to form a set of temporary clusters. The seeds are then replaced by the cluster means, the points are reassigned, and the process continues until no further changes occur in the clusters. We validated by defined the number of clusters correspondence to the number of track pitch classes, so we tested from 2 clusters until 9 clusters to observe which clusters number return the best output.

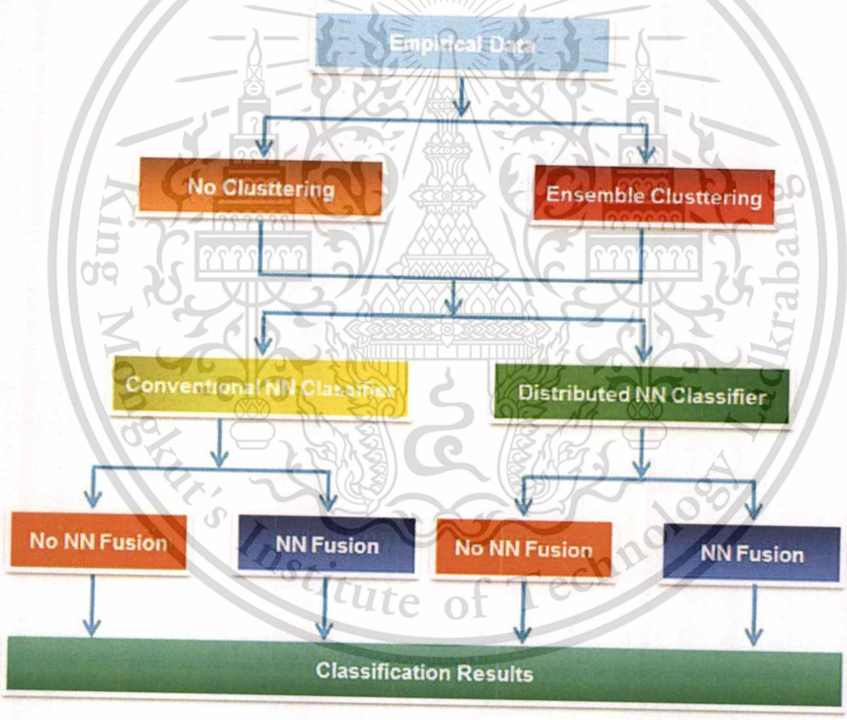


Fig. 4.2 Diagram showing model flow Conventional and Distributed NN Classification with Clustering Ensemble.

With adding clustering, the model can be improved significantly as the result show in Table IV and Table V. Every classes perform a better result and with higher numbers of cluster potentially able to sustain higher accuracy.

Table 4.3. Classification results from Conventional NN Classifier with and without NN Fusion and with Cluster ensemble of each class.

Method	Ensemble Cluster + Conventional NN Classifier without NN Fusion								Ensemble Cluster + Conventional NN Classifier + NN Fussion with NN Fusion							
Class/Cluster	2	3	4	5	6	7	8	9	2	3	4	5	6	7	8	9
TPI 1=51	62.94%	96.08%	27.84%	53.33%	32.35%	50.59%	51.37%	61.76%	88.24%	92.16%	86.27%	84.31%	96.08%	84.31%	96.08%	92.16%
TPI 2=75	46.00%	65.33%	38.53%	43.20%	43.60%	54.27%	32.00%	46.00%	52.00%	48.00%	62.67%	45.33%	57.33%	60.00%	72.00%	66.67%
TPI 3=603	82.27%	83.25%	84.34%	83.96%	84.03%	84.18%	83.98%	84.59%	82.26%	88.06%	86.90%	89.39%	84.91%	87.89%	86.73%	88.23%
TPI 4=1260	90.16%	90.87%	89.74%	89.52%	89.67%	89.27%	88.21%	89.22%	90.40%	90.87%	90.87%	90.32%	90.32%	90.00%	90.71%	91.90%
TPI 5=2334	92.28%	92.20%	92.75%	92.60%	93.00%	93.05%	92.24%	92.14%	94.04%	93.27%	93.87%	93.14%	94.56%	94.73%	93.62%	93.40%
TPI 6=2421	92.74%	91.70%	93.02%	93.01%	92.99%	93.36%	92.93%	92.46%	92.81%	92.07%	92.94%	94.34%	94.84%	93.89%	94.75%	93.23%
TPI 7=674	82.60%	85.01%	81.57%	81.91%	83.22%	81.74%	83.56%	82.88%	81.90%	89.17%	86.50%	83.83%	79.97%	84.72%	84.42%	82.94%
TPI 8=63	10.79%	33.33%	23.17%	22.70%	13.17%	20.48%	9.84%	23.81%	44.44%	0.00%	42.86%	33.33%	46.03%	41.27%	0.00%	41.27%
TPI 9=2	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Overall	89.02%	89.68%	89.05%	89.18%	89.23%	89.47%	88.73%	89.04%	90.08%	90.42%	91.02%	90.85%	91.05%	91.31%	91.10%	90.99%

Table 4.4 Classification results from Distributed NN Classifier with and without NN Fusion and with Cluster ensemble of each class.

Method	Ensemble Cluster + Distributed NN Classifier without NN Fusion								Ensemble Cluster + Distributed NN Classifier + NN Fussion with NN Fusion							
Class/Cluster	2	3	4	5	6	7	8	9	2	3	4	5	6	7	8	9
TPI 1=51	100.00%	100.00%	96.08%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	96.08%	100.00%	100.00%	100.00%	100.00%	100.00%
TPI 2=75	41.33%	69.33%	76.00%	72.00%	50.67%	45.33%	81.33%	53.33%	88.00%	73.33%	80.00%	81.33%	85.33%	82.67%	97.33%	74.67%
TPI 3=603	86.73%	91.38%	88.23%	92.54%	87.40%	89.22%	91.38%	94.03%	91.04%	87.89%	89.55%	91.71%	87.56%	89.55%	90.55%	89.05%
TPI 4=1260	92.54%	91.03%	92.54%	92.46%	92.70%	91.90%	87.78%	90.48%	92.46%	92.46%	92.78%	93.10%	90.87%	92.86%	92.70%	90.79%
TPI 5=2334	94.34%	94.17%	94.34%	94.16%	95.07%	85.56%	94.99%	95.67%	94.64%	94.17%	94.77%	95.20%	94.90%	94.30%	94.73%	95.59%
TPI 6=2421	90.83%	94.47%	93.35%	94.96%	93.97%	94.34%	93.02%	91.33%	94.55%	93.76%	94.09%	94.34%	94.42%	94.34%	94.63%	93.97%
TPI 7=674	82.34%	83.53%	89.61%	85.91%	86.80%	87.39%	81.90%	84.87%	86.20%	86.80%	87.24%	85.91%	86.65%	87.24%	83.23%	86.80%
TPI 8=63	95.24%	98.41%	92.06%	87.30%	98.41%	96.83%	98.41%	98.41%	100.00%	96.83%	93.65%	96.83%	100.00%	100.00%	98.41%	98.41%
TPI 9=2	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Overall	90.73%	92.38%	92.61%	93.02%	92.57%	89.72%	91.59%	91.92%	93.18%	92.41%	92.94%	93.33%	92.69%	93.00%	93.05%	92.76%

The result on this study is obtained to overall summary in Table VI and Table VII. It is observed that the distributed NN classifier performs better accuracy and constantly compare to Conventional NN Classifier. Base on the data from 3 trial validation resulted 91.4% in average without NN fusion and 91.95% in average with NN fusion.

Table 4.5 Overall results from Intelligent TPI classification with non Cluster Ensemble.

Intelligent TPI Classification Methods	% Accuracy			
	Trial 1	Trial 2	Trial 3	Average
Conventional NN Classifier without NN Fusion	87.33%	87.66%	88.80%	87.93%
Conventional NN Classifier with NN Fusion	90.43%	90.45%	90.32%	90.40%
Distributed NN Classifier without NN Fusion	91.58%	91.22%	90.62%	91.14%
Distributed NN Classifier with NN Fusion	91.60%	92.54%	91.70%	91.95%

Further more when assembled the Cluster to Conventional and Distributed NN Classifier model, the percentage of classification accuracy significant improve compare to non Clustering. By selected top 3 and average out we can maximum the performance with Distributed NN Classifier and NN Fusion at 93.19%.

Table 4.6 Overall results from Intelligent TPI classification with Cluster Ensemble.

Ensemble Cluster + Intelligent TPI Classification Methods	% Accuracy			
	Top1	Top2	Top3	Average
Cluster + Conven NN Classifier	89.47%	89.23%	89.18%	89.29%
Cluster + Conven NN Classifier + NN Fussion	91.31%	91.10%	91.05%	91.15%
Cluster + Distributed NN Classifier	93.33%	92.61%	92.57%	92.84%
Cluster + Distributed NN Classifier + NN Fussion	93.33%	93.18%	93.05%	93.19%

The Fig. 4.3 shows the level of improvement is add up after we concur the Distributed NN Classifier with K-Mean Cluster to the classification model. The data also proved that the designed method is suitable and feasible to unequal samples size for track pitch classification. The model is simplified and should able to implement for larger data set.

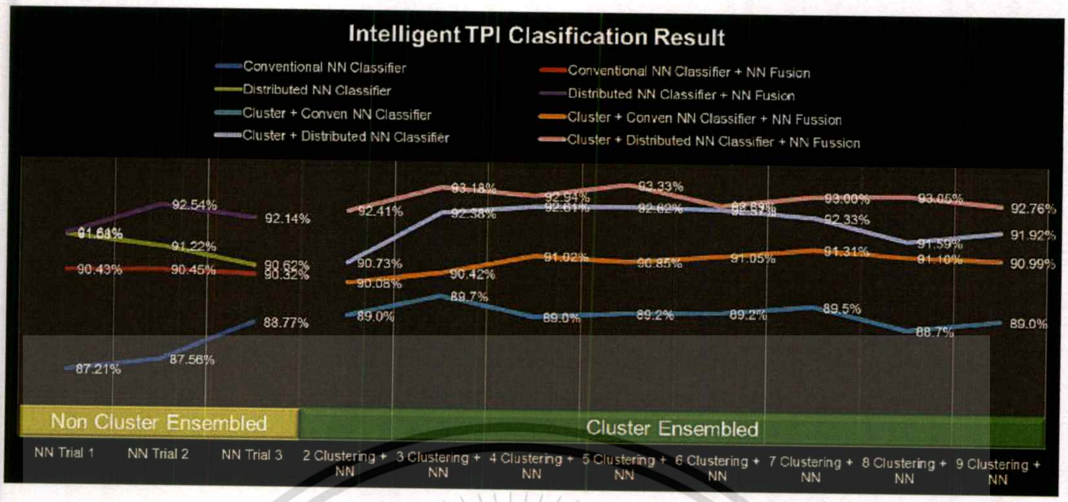


Fig. 4.3 Overall Intelligent TPI Classification accuracy with and without Cluster ensemble.

4.4 Discussion: Distributed NN Classification

The distributed NN classification is approached to combine the distributed predictive models in an attempt to derive a single global model. However, such a single model in the distributed data might not really exist but two or more groups of models. There should not one just blindly integrate local models, but do some sort of reasoning for the relations between local models before the integration.

The combination of multiple classifiers is an important research issue that has been investigated in the past from the communities of statistics, machine learning, pattern recognition, data mining and knowledge discovery. When only the label of the predicted class is available, then the simplest combination method that can be used is Majority Voting, which does not require a training stage. In this case, the class that receives the most classifier predictions is the final result. Weighted Majority Voting, weights the decision of each classifier by its performance on the training data. When a measure of belief, confidence, certainty or other about the classification is available along with the class label, then a number of different rules for combining these measures have been suggested, like Sum, Min, Max, Prod and Median is an interesting study of these rules. Stacked Generalization is a method that combines multiple classifiers by learning the way that their output.

In this study, The Distributed NN Classifier allows the individual NN classifier to train on only one track pitch number (outcome) and the outputs are then combined via the NN Fusion model. The final accuracy aggregation shown that the accuracy of each TPI can then be improved significantly and reflected to overall prediction accuracy is at 91.60%, 92.54% and 91.70% for trial 1, 2 and 3 respectively and 91.95% in average.

The comparison results of all three different models, the Distributed NN Classifier with the NN Fusion and the Conventional NN Classifier with and without the NN Fusion suggest that the Distributed Neural Network Fusion provides superior in accuracy of track pitch classification, and also perform very good correlation to the current practice in HDD as confirmed by p-Value of the results of mean comparison shown in Fig 4.4.

Level	-Level	Difference	Lower CL	Upper CL	p-Value
CONVENTIONAL W/O FUSION	CONVENTIONAL W FUSION	0.0112254	-0.008203	0.0306540	0.2575
CONVENTIONAL W/O FUSION	DISTRIBUTED W FUSION	0.0071718	-0.012257	0.0266004	0.4694
CONVENTIONAL W/O FUSION	HDD	0.0061473	-0.022996	0.0352901	0.6793
HDD	CONVENTIONAL W FUSION	0.0050782	-0.025641	0.0357975	0.7459
DISTRIBUTED W FUSION	CONVENTIONAL W FUSION	0.0040536	-0.017688	0.0257754	0.7145
HDD	DISTRIBUTED W FUSION	0.0010245	-0.029695	0.0317438	0.9479

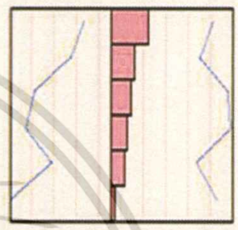


Fig. 4.4 p-Value of the results of mean comparison between NN Classification and factory standard in Hard Disk Drive.

4.5 Discussion: Ensemble of Clustering and Distributed NN Classification

The Clustering is an addition approach in this study by combine with NN Classification. The K-mean is a selected algorithm which is the most popular tool in scientific and industrial applications. It has the good geometric and statistical sense for numerical attributes. The sum of discrepancies between a point and its centroid expressed through appropriate distance is used as the objective function. For example, the -norm based objective function, the sum of the squares of errors between the points and the corresponding centroids, is equal to the total intra-cluster variance. The sum of the squares of errors can be rationalized as (a negative of) log-likelihood for normally distributed mixture model and is widely used in statistics.

The final Ensemble model of K-Mean Clustering and Distributed NN Classification is a suitable method for the unequal attributors as the result return outperform at 93.33% accuracy over without clustering model.

The final Ensemble model of K-Mean Clustering and Distributed NN Classification is a suitable method for the unequal attributors as the result return outperform at 93.33% accuracy over without clustering model.

4.6 Classification Accuracy, BER and Test Time Comparison

To measure the advantage of the proposed technique, the classification accuracy, test time and BER performance are compared to current practice in HDD and the conventional NN classifier from multiple trials.

Table 4.7 Accuracy, BER and Test Time Comparison.

Intelligent TPI Classification Methods (Compare to Current Practice in HDD)	Classification Accuracy (%)	BER Comparison (Order)	Test Time Reduction per One Head (%)
Current Practice in HDD	Ref.	Ref.	50 minutes
Conventional NN Classifier with NN Fusion	90.40%	-0.029	70%
Distributed NN Classifier with NN Fusion	91.95%	-0.024	60%
Ensemble Clustter and Distributed NN Classifier with NN Fusion	93.19%	-0.021	55%

The intelligent track pitch classification using NN requires input data from actual track pitch calibration process in HDD at least 1 step, prior performing the analysis using NN Classifier with NN Fusion, whereas the current practiced in HDD has to calibrate for every step until reaching the last step of track pitch selection.

There are3 steps proposed intelligent classification; Clustering, NN classifier and NN Fusion is required about 5-10 minutes of each steps to finish the classification process. Table VIII shown the total test time of the current practice can reduce by 55%-70%. The BER performance also validated in order to confirm no any trade off from the proposed method and there is insignificant BER different between existing and all classification methods however as a result showed that the proposed system accuracy is the best in class or superior method. This can imply that the proposed system potentially have less variation or better sigma values as opposed to other classifiers when validate with larger data.

Chapter 5

Contributions and Further Work

5.1 Final Model

In this thesis, we have studied and proposed the new intelligent track pitch classification using Ensemble K-Mean Clustering with Distributed NN classifier and NN Fusion method as shown in Fig. 5.1 to classify the Track Pitch Calibration format accurately with certain test time reduction and no impact to the performance.

Empirical data is screened after performed pre-processing by significant test and then pre-classified by K-Mean prior input to Distributed NN classifier, these pre-processing and pre-classification steps are used to reduce the dimension of data and cascade the group of distribution.

In the classification step we approach the Distributed NN model to classify each track pitch class individually that aim to allow NN learn on one single distribution rather than entire multiple distributions like conventional method, then the two NN classifiers experienced differently and the result proving that with the focusing learning Distributed NN classifier, the result of each class one to one comparison obtain significant better accuracy.

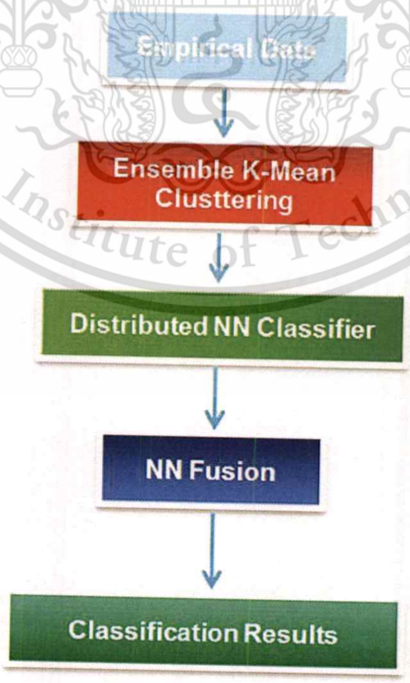


Fig. 5.1 Diagram showing final model flow.

The Distributed NN classifier is trained smaller input data then each NN classifier can provide a good result due to lesser complexity of learning and when we combined the experience of each NN Classifier to NN Fusion, the final classification accuracy also more improve significantly.

As a result, we showed that track pitch classification can be obtained through integration

of K-Mean clustering and Distributed NN classifiers with NN fusion. The final result is able to accurately predict Track Pitch at 93.19% confidence level with 55% test time reduction per one head and has small impact in BER performance.

In summary, we successfully constructed a new classification model. This model can then be applied to the Track Pitch Calibration process in the manufacturing of HDD to achieve a quick and accurate convergence in track pitch selection.

5.2 Further Work

The designed classification model of this work shown that the track pitch calibration in HDD can be classified by enhancing statistic model, the model also feasible to allow one class train with equally input parameters, not necessary to determine only normal distribution that always encountered on high variation when it's train on small quantity of the tail end. This also imply us that the model is simplify to HDD process, that's mean In future work when we apply the model to HDD process we can force by collect more samples of each class to train in our model.

Furthermore we can modify the model in order to improve the classification accuracy, as we also observed the pre-classifier of K-Mean able to cascade down again to individual class before entered the output to Distributed NN Classifier (One class will have one K-Mean clustering and one NN Classifier). This may create a larger model however we can satisfy by the end result of classification accuracy.

References

[1] Guoxiao Guo., Qi Hao. and Teck-Seng Low., "**Access system requirement for high track per inch hard disk drives.**", Asia-Pacific Recording Conference, 2000.

[2] Guoxiao Guo., Qi Hao., Teck-Seng Low., "A dual-stage control design for high track per inch hard disk drives.", **IEEE Transactions on Magnetics**, vol. 37, Issue 2, Part 1, 2001.

[3] Pornchai Supnithi., Kosit Bunsri. and Majid Novid. "A Flexible KFCI Compensation for Head/Media Optimization in Perpendicular Disk Drives.", **IEEE Transactions on Magnetics**, vol. 47, no. 7, July 2011

[4] Takahiro Saito., Kazunari Tsuchimoto. and Atsushi Kanamaru., "Disk Drive and Method for Determining Data Track Format of The Disk Drive Device.", **US Patent Application Publication 2011/8009376 B2**, 30 Aug 2011.

[5] Shanlin Duan., Zhen Jin., Jie Zhang. and Kezhao Zhang. "System, Method and Apparatus for Determining Track Pitch in A Hard Disk Drive to Satisfy the Requirements of Both Off-Track Capacity and Adjacent Track Erasure,", **US Patent Application Publication 2011/0075286 A1**, 31 Mar 2011.

[6] Michael S. Rothberg., "Disk Drive Refreshing Zones Based on Serpentine Access of Disk Surface.", **US Patent Application Publication 2011/7872822 B1**, 18 Jan 2011.

[7] Chun Sei Tsai., Stephany W. Kwan. and Kameron Kam-Wai Jung., "Disk Drive Employing Different Zone Boundaries Across Disk Surfaces.", **US Patent Application Publication 2008/7436610 B1**, 14 Oct 2008.

[8] Lin Mei., Xiaokun L. Chew. and Teik EE Yeo., "Disk Drive Selecting TPI Profile by Estimating Head Geometry.", **US Patent Application Publication 2010/7688540 B1**, 30 Mar 2011.

[9] Chun Sei Tsai., Stephany W. Kwan. and Kameron Kam-Wai Jung., "Disk Drive Generating Actual Data TPI Profile by Combining Segments of Predetermined Data TPI Profile.", **US Patent Application Publication 2011/8031423 B1**, 4 Oct 2011.

[10] Kay-Soon Low. and Tze-Shyan Wong. "A Multiobjective Genetic Algorithm for Optimizing the Performance of Hard Disk Drive Motion Control System.", **IEEE Transactions on Industrial Electronics**, vol. 54, no. 3, June 2007

[11] Jason Levin., Néstor O. Pérez-Arancibia., Petros A. Ioannou. and Tsu-Chin Tsao. "A Neural-Networks-Based Adaptive Disturbance Rejection Method and Its Application to the Control of Hard Disk Drives.", **IEEE Transactions on Magnetics**, vol. 45, No. 5, May 2009

[12] Michelle Y. Wang. and Bing J. Sheu. **"Compact Neural Network Detector for Hard-Disk Drive Using Zero-Forcing Preprocessing."**, Department of Electrical Engineering, University of Southern California, Los Angeles, CA 90089-0271

[13] Seiichiro Morizawa., Shinkyu Jeong., Koji Shimoyama. and Shigeru Obayashi. **"Data Mining for the Investigation of Unsteady Flow Field in a Hard Disk Drive."**, 2010 Second World Congress on Nature and Biologically Inspired Computing, Dec 2010, Kitakyushu, Fukuoka, Japan

[14] Chow Yin Lai., Frank L. Lewis., V. Venkataramanan., Xuemei Ren., Shuzhi Sam Ge. and Thomas Liew. **"Neural Networks for Disturbance and Friction Compensation in Hard Disk Drive."** Proceedings of the 47th IEEE Conference on Decision and Control Cancun, Mexico, Dec. 9-11, 2008

[15] W. Tepin. and Y. Kidjaidure. **"Enhanced Failure Mode Prediction for Hard Disk Drive using Neural Network Rank-Level Fusion via Negative Correlation."**, The 4th International Data Storage Technology Conference. Thailand Conference, July 30 – Aug 1, 2010.

[16] K. Pomoung. and Y. Kidjaidure, **"Rank-Level Fusion with Neural Network in Classification of Media Defect Patterns."**, The 3rd International Data Storage Technology Conference. Thailand – Conference, Jan 9-12, 2012.

[17] Abdellatif Ennaji., Arnaud Ribert. and Yves Lecourtier., **"From Data Topology to a Modular Classifier."**, Perception System Information Laboratorty, University of Rouen, F-76821, Mont Saint Aignan Cedex, France

[18] Hartigan J. 1975. **"Clustering Algorithms."**, John Wiley & Sons, New York, NY.

[19] Spath H. 1980. **"Cluster Analysis Algorithms."**, Ellis Horwood, Chichester, England.

[20] Jain A., Dubes and R. 1988. **"Algorithms for Clustering Data."**, Prentice-Hall, Englewood Cliffs, NJ.

[21] Kaufman L. and Rousseeuw P. 1990. **"Finding Groups in Data: An Introduction to Cluster Analysis."**, John Wiley and Sons, New York, NY.

[22] Dubes R.C. 1993. **"Cluster Analysis and Related Issues."** In Chen, C.H., Pau, L.F., and Wang, P.S. (Eds.) Handbook of Pattern Recognition and Computer Vision, 3 32, World Scientific Publishing Co., River Edge, NJ.

[23] Everitt B. 1993. **"Cluster Analysis (3rd ed.)."** Edward Arnold, London, UK.

[24] Mirkin B. 1996. **"Mathematic Classification and Clustering."**, Kluwer Academic Publishers.

[25] Jain A.K, Murty M.N. and Flynn P.J. 1999. **"Data clustering: a review."**, ACM Computing Surveys, 31, 3, 264-323.

[26] Fasulo D. 1999. **"An analysis of recent work on clustering algorithms."**, Technical Report UW-CSE01 -03-02, University of Washington.

[27] Kolatch E. 2001. **"Clustering Algorithms for Spatial Databases: A Survey."**, PDF is available on the Web.

[28] Ghosh J., 2002. **"Scalable Clustering Methods for Data Mining."** In Nong Ye (Ed.) Handbook of Data Mining, Lawrence Erlbaum, to appear.

[29] Arabie P. and Hubert L.J. 1996. **"An overview of combinatorial data analysis."**, in Clustering and Classification, 5-63, World Scientific Publishing Co., NJ.

[30] Massart D. and Kaufmann L. 1983. **"The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis."**, John Wiley & Sons, New York, NY.

[31] Duda R. and Hart P. 1973. **"Pattern Classification and Scene Analysis."**, John Wiley & Sons, New York, NY.

[32] Jain A.K. and Flynn P.J. 1966. **"Image segmentation using clustering."**, In Advances in Image Understanding: A Festschrift for Azriel Rosenfeld, IEEE Press, 65-83.

[33] Demster A., Laird N., and Rubin D. 1977. **"Maximum likelihood from incomplete data via the EM algorithm."**, Journal of the Royal Statistical Society, Series B, 39, 1, 1-38.

[34] Fukunaga K. 1990. **"Introduction to Statistical Pattern Recognition."**, Academic Press, San Diego, CA.

[35] Scott, D.W. 1992. **"Multivariate Density Estimation."**, Wiley, New York, NY.

[36] Gersho A. and Gray R.M. 1992. **"Vector Quantization and Signal Compression."**, Communications and Information Theory. Kluwer Academic Publishers, Norwell, MA.

[37] Daniel C. and Wood F.C. 1980. **"Fitting Equations To Data: Computer Analysis of Multifactor Data."**, John Wiley & Sons, New York, NY.

[38] Cutting D., Karger D., Pedersen J. and Tukey J. 1992. **"Scatter/gather: a clusterbased approach to browsing large document collection."**, In Proceedings of the 15th ACM SIGIR Conference, 318-329, Copenhagen, Denmark.

[39] Steinbanch M., Karypis G., and Kumar V. 2000. **"A comparison of document clustering techniques."**, 6th ACM SIGKDD, World Text Mining Conference, Boston, MA.

[40] Dhillon I., Fan J., and Guan Y. 2001. **"Efficient clustering of very large document collections."**, In Grossman R.L., Kamath C., Kegelmeyer P. Kumar V., and Namburu R.R. (Eds.) Data Mining for Scientific and Engineering Applications, Kluwer Academic Publishers.

[41] Xu X., Ester M., Kriegel H.-P. and Sander J. 1998. **"A distribution-based clustering algorithm for mining large spatial datasets."**, In Proceedings of the 14th ICDE, 324-331, Orlando, FL.

[42] Sander J., Ester M., Kriegel H.-P. and Xu X. 1998. **"Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications."**, In Data Mining and Knowledge Discovery, 2, 2, 169-194.

[43] Ester M., Frommelt A., Kriegel H.-P., and Sander J. 2000. **"Spatial data mining: database primitives, algorithms and efficient DBMS support."**, Data Mining and Knowledge Discovery, Kluwer Academic Publishers, 4, 2-3, 193-216.

[44] Cadez I., Smyth P., and Mannila H. 2001. **"Probabilistic modeling of transactional data with applications to profiling, Visualization, and Prediction."**, In Proceedings of the 7th ACM SIGKDD, 37-46, San Francisco, CA.

[45] Cooley R., Mobasher B., and Srivastava J. 1999. **"Data preparation for mining world wide web browsing."**, Journal of Knowledge Information Systems, 1, 1, 5 32.

[46] Heer J. and Chi E. 2001. **"Identification of Web user traffic composition using multimodal clustering and information scent."**, 1st SIAM ICDM, Workshop on Web Mining, 51-58, Chicago, IL.

[47] Foss A., Wan W. and Zaane O. 2001. **"A non-parametric approach to Web log analysis."**, 1st SIAM ICDM, Workshop on Web Mining, 41-50, Chicago, IL.

[48] Ben-Dor A. and Yakhini Z. 1999. **"Clustering gene expression patterns."**, In Proceedings of the 3rd Annual International Conference on Computational Molecular Biology (RECOMB 99), 11-14, Lyon, France.

[49] Han J. and Kamber M. 2001. **"Data Mining."**, Morgan Kaufmann Publishers.

[50] Tung A.K.H., Hou J. and Han J. 2001. **"Spatial clustering in the presence of obstacles."**, In Proceedings of the 17th ICDE, 359-367, Heidelberg, Germany.

[51] Bradley P. S., Bennett K. P. and Demiriz A. 2000. **"Constrained k-means clustering."**, Technical Report MSR-TR-2000-65. Microsoft Research, Redmond, WA.

[52] Banerjee A. and Ghosh J. 2002. **"On scaling up balanced clustering algorithms."**, In Proceedings of the 2nd SIAM ICDM, 333-349, Arlington, VA.

[53] Strehl A. and Ghosh J. 2000. **"A scalable approach to balanced, high dimensional clustering of market baskets."**, In Proceedings of 17th International Conference on High Performance Computing, Springer LNCS, 525-536, Bangalore, India.

[54] Kalton A., Langley P., Wagstaff K., and Yoo J. 2001. **"Generalized clustering, supervised learning, and data assignment."**, In Proceedings of the 7^{th} ACM SIGKDD, 299-304, San Francisco, CA.

[55] Liu B., Xia Y. and Yu P.S. 2000. **"Clustering through decision tree construction."**, In SIGMOD 2000.

[56] Marroquin J.L. and Girosi F. 1993. **"Some extensions of the k-means algorithm for image segmentation and pattern classification."**, A.I. Memo 1390. MIT, Cambridge, MA.

[57] Kohonen T. 1990. **"The self-organizing map."** Proceedings of the IEEE, 9, 1464 1479.

[58] Kohonen T. 2001. **"Self-Organizing Maps."** Springer Series in Information Sciences, 30, Springer.

[59] Carpenter G.A., Grossberg S., and Rosen D.B. 1991. **"Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system."** Neural Networks, 4, 759-771.

[60] Jain A.K. and Mao J. 1996. **"Artificial neural networks: A tutorial."**, IEEE Computer, 29, 3, 31-44.

[61] Goldberg D. 1989. **"Genetic Algorithms in Search, Optimization, and Machine Learning."** Addison-Wesley.

[62] Hall L.O., Ozyurt B. and Bezdek J.C. 1999. **"Clustering with a genetically optimized approach."**, IEEE Trans. on Evolutionary Computation, 3, 2, 103 112.

[63] Sarafis I., Zalzala A.M.S. and Trinder P.W. 2002. **"A genetic rule-based data clustering toolkit."**, To be published in Congress on Evolutionary Computation (CEC), Honolulu, USA.

[64] Cristofor D. and Simovici D.A. 2002. **"An information-theoretical approach to clustering categorical databases using genetic algorithms."**, 2nd SIAM ICDM, Workshop on clustering high dimensional data, Arlington, VA.

[65] Babu G.P. and Marty M.N. 1993. **"A near-optimal initial seed value selection in Kmeans algorithm using a genetic algorithm."**, Pattern Recogn. Lett. 14, 10, 763-169.

[66] Babu G.P. and Marty M.N. 1994. **"Clustering with evolution strategies."**, Pattern Recognition, 27, 2, 321-329.

[67] Lee C-Y. and Antonsson E.K. 2000. **"Dynamic partitional clustering using evolution strategies."** In Proceedings of the 3rd Asia-Pacific Conference on Simulated Evolution and Learning, Nagoya, Japan.

[68] Xu X., Ester M., Kriegel H.-P. and Sander J. 1998. **"A distribution-based clustering algorithm for mining large spatial datasets."**, In Proceedings of the 14th ICDE, 324-331, Orlando, FL.

[69] Kart-Leong Lim. and Hamed Kiani Galoogahi. **"Shape Classification Using Local and Global Features."**, 2010 Fourth Pacific-Rim Symposium on Image and Video Technology

[70] Lin Xiao., Stephen Boyd. and Sanjay Lall. **"A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus."**, International Conference on Information Processing in Sensor Networks, April 2005, p63-70, Los Angeles, 2005.

[71] Jorge Cortes., **"Analysis and design of distributed algorithms for χ consensus."**, Proceedings of the 45th IEEE Conference on Decision & Control Manchester Grand Hyatt Hotel San Diego, CA, USA, December 13-15, 2006

[72] Cheeha Kim., Jai-Yong Lee. and Chan-Mo Park **"Hierarchical Decision Structure for Distributed Algorithms."**, Department of Computer Science/PIRL Pohang Inatitute of Science and Technology P.O. Box 125 Pohmg, 790-600, KOREA

[73] Ying-Wu Fang., Xiu-Bing Zhao., Guang-Peng Zhang., Yi Wang., Yi Sun. and Yong Fang Zhang. **"Study on Algorithms of Parallel and Distributed Data Mining Calculating process."**, Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 August 2005

[74] Pavel Berkhin. **"Survey of Clustering Data Mining Techniques."**, Accrue Software, 1045 Forest Knoll Dr., San Jose, CA, 95129

[75] X. Yao and Md. Monirul Islam "Evolving Artificial Neural Network Ensembles.," **IEEE Computational Intelligence Magazine,** February 2008. pp. 31-42.

[76] Reza Olfati Saber., J. Alex Fax and Richard M. Murray. **"Consensus and Cooperation in Networked Multi-Agent Systems.,"** Proceedings of the IEEE, vol. 95, no. 1, January, 2007

[77] Danilo P. Mandic., Dragan Obradovic., Anthony Kuh., Tulay Adali., Udo Trutschell., Martin Golz. **"Data Fusion for Modern Engineering Applications: An Overview."**, W. Duch et al. (Eds.): ICANN 2005, LNCS 3697, pp. 715–721, 2005. C Springer-Verlag Berlin Heidelberg 2005

Author Biography

Name: Mr. Wiwat Rodsud

Date of Birth: 21 September 1980

Place of Birth: Phatthalung, Thailand

Address: 497 Apaiboriruk Road, Khuhasawan, Muang, Phatthalung, 93000

Education:

1999-2003 Bachelor degree, Faculty of Engineering, Majoring in Electrical Engineering, King Mongkut's Institute of Technology Ladkrabang.

Work Experience:

2003-Now Head and Media Engineering at Western Digital (Thailand) Co., Ltd.

Scholarships:

2009-2011 Scholarship for study in the Master degree from Data Storage Technology Innovation, King Mongkut's Institute of Technology Ladkrabang.

Publications:

1. W. Rodsud, Y. Kidjaidure, K. Pomoung and C.W. Mak, "Intelligent Track Pitch Classification in Hard Disk Drive with integration of Distributed Neural Network Classifier and Neural Network Fusion," The 4th International Data Storage Technology Conference. Thailand –Conference, January9-10, 2012.

DST-CON 2011

The 4th International Data Storage Technology Conference

Thailand – The paragon of the world's HDD manufacturing

IMPACT Convention Center (Hall 9)
Muangthong Thani, Nonthaburi, Thailand
January 9-10, 2012

:: Organized by
The Board of Investment of Thailand (BOI)
National Science and Technology Development Agency (NSTDA)
Hard Disk Drive Institute (HDDI)
I/U CRC in HDD Component

Sponsored by

Conference Program

Jan. 9, 2012

Time/Venue	Sapphire 4-5
08.00-09.00	Registration
09.00-09.30	Opening Ceremony
09.45-10.15	Keynote Session Keynote Speaker: Dr.Chatri Sripaipan (NSTDA)
	Coffee Break
10.35-11.15	Keynote Session Keynote Speaker: Dr.Mark Re (Seagate Technology)
11.15-12.00	Keynote Session Keynote Speaker: Mr.Dave Rauch (Western Digital)
	Lunch (Jupiter 4-6)

Jan. 9, 2012

Time/Venue	Sapphire 4	Sapphire 5	Sapphire 6
13:00 - 13:20	IP5	IP1	IP2
13:20 - 13:40	DSTA1	ACD1	IP3
13:40 - 14:00	DSTA2	ACD2	PPI1
14:00 - 14:20	DSTA3	ACD3	PPI2
14:20 - 14:40	DSTA4	ACD4	PPI3
14:40 - 15:00	Coffee Break		
15:00 - 15:20	IP7	IP4	IP6
15:20 - 15:40	DSTA5	ACD5	PPI4
15:40 - 16:00	DSTA6	ACD6	PPI5
16:00 - 16:20	DSTA7	ACD7	PPI6
16:20 - 16:40	DSTA8	ACD8	PPI7
16:40 - 17:00	DSTA9	ACD9	PPI8
17:00 - 17:20	DSTA10	ACD10	PPI9
17:20 - 17:40	DSTA11	ACD11	PPI10

DSTA 3

Title: Intelligent Track Pitch Classification in Hard Disk Drive with integration of Distributed Neural Network Classifier and Neural Network Fusion

Author: [1]Wiwat Rodsud
[2]Yuttana Kidjaidure
[3]Kittiphan Pomoung
[4]Chee Wai Mak

Affiliation: [1]Faculty of Engineering, KMITL, Bangkok, Thailand
[2]Western Digital (Thailand) Company Limited, Ayutthaya, Thailand

E-mail: [1]wiwat.rodsud@wdc.com
[2]kkyuttan@kmitl.ac.th
[3]kittiphan.pomoung@wdc.com
[4]mak.cheewai@wdc.com

Abstract: It is well-known that Track Per Inch (TPI) and Bit Per inch (BPI) are fundamental parameters defining HDD Areal Density (bits per square inch). TPI and BPI settings are calibrated to achieve Bit Error Rate (BER) performance at the desired areal density. However, both BPI and TPI calibrations are complex and time-consuming processes. This paper proposes an intelligent TPI calibration method using the distributed neural network (NN) classification model together with neural network fusion model and it is compared to the conventional NN Classification. The experiment results show that the proposed TPI calibration method provides significantly better performance than the conversional

Intelligent Track Pitch Classification in Hard Disk Drive with integration of Distributed Neural Network Classifier and Neural Network Fusion

Wiwat Rodsud[#1], Yuttana Kidjaidure[#2], Kittiphan Pomoung[*3], Chee Wai Mak[*4]

[#] *Faculty of Engineering, KMITL*
Bangkok, Thailand

[*] *Western Digital (Thailand) Company Limited*
Ayutthaya, Thailand

[1]wiwat.rodsud@wdc.com

[2]kkyuttan@kmitl.ac.th

[3]kittiphan.pomoung@wdc.com

[4]mak.cheewai@wdc.com

Abstract— **It is well-known that Track Per Inch (TPI) and Bit Per inch (BPI) are fundamental parameters defining HDD Areal Density (bits per square inch). TPI and BPI settings are calibrated to achieve Bit Error Rate (BER) performance at the desired areal density. However, both BPI and TPI calibrations are complex and time-consuming processes. This paper proposes an intelligent TPI calibration method using the distributed neural network (NN) classification model together with neural network fusion model and it is compared to the conventional NN Classification. The experiment results show that the proposed TPI calibration method provides significantly better performance than the conversional NN classification method. Moreover, the TPI calibration of the proposed method has significantly lesser time consumed at the same BER performance as the existing TPI Calibration method. We therefore propose the intelligent TPI calibration method which integrates the distributed NN classification with NN fusion.**

Keywords— **Hard Disk Drive, Neuron Network Fusion, Head-Disk Interaction, Classification, Distributed Algorithm**

I. Introduction

In the high density track pitch implementation of many Hard Disk Drive (HDD) designs in the modern day to meet ever-increasing areal density challenges, the calibration of track pitch to accommodate the read/write transducer is the primary concern. Track pitch calibration algorithms emerged from the need to accurately determine the track density to match the areal density design at an acceptable bit-error-rate (BER) performance, such as [1] and [2], address system stability at high density track implementations. The existing TPI Calibration process evaluates one TPI setting at a time in a given set of TPIs until the TPI meets both capacity and performance. Each calibration process step requires time and the total time consumed is depended on the number of heads, zones and TPI settings.

The total TPI calibration test time found uses the following rules,

$$t_{Max} = \sum_{i=1}^{H} t_h \times \sum_{j=1}^{TPI} t_{TPI} \times \sum_{k=1}^{Z} t_z \qquad (1.)$$

Where t_{Max} is the maximum TPI test time, t_h is the test time consumption of each head, t_{TPI} is the time consumption of each TPI test time and t_Z is the time consumption of each zone, H is the number of heads, TPI is the number of designed TPI, Z is the number of zones in Hard Disk Drive.

The reduction of the TPI Calibration time significantly improves HDD manufacturing throughput. However, the HDD drive performance and areal density cannot be compromised (trade off) with the reduced process time. It is observed that the results are often not repeatable or reproducible. Empirical data from failure analysis also suggest that the accuracy of the track pitch calibration contributes largely to bit-error-rate performance. This further motivates the search of a quicker and more intelligent calibration method by the challenges to balance performance of areal density and calibration time.

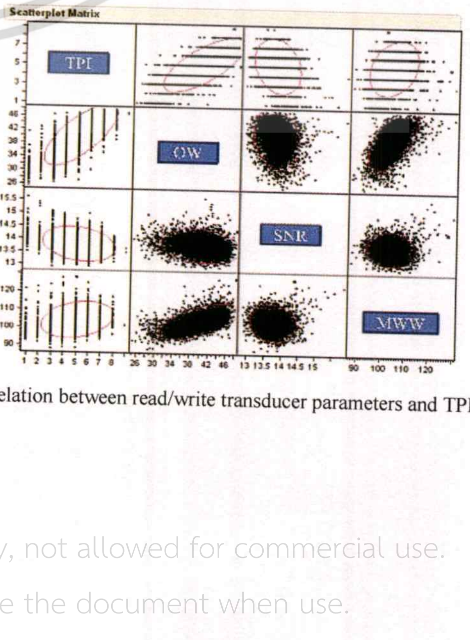


Fig. 1 Correlation between read/write transducer parameters and TPI

Before performing the HDD TPI Calibration process, the read/write transducer parameters are measured and recorded. Generally, there is a fair correlation between the read/write transducer parameters and TPI shown in Fig. 1, however, the HDD BER performance is deteriorated significantly if the read/write transducer parameters have to predict TPI directly. In order to maintain the HDD BER performance with lesser TPI calibration time, the new calibration method is proposed.

Recently, there are many researchers apply the NN technique to several HDD aspects [3]-[5]. In [3], the proposed method addresses time-varying uncertainty and non-linearity of the problem. In [4], multiple classifications methods, namely Neural Net, Bayesian, Support Vector Machine and Discriminant Analysis technique are combined via fusion technique to predict reliability. The research also pointed out that the NN fusion improves the overall performance. The integration of the classification and fusion system has better performance than the single classification.

In [5], a proposed compact neural network based partial response maximum likelihood (PRML) detector with zero-forcing pre-processing for hard disk drive reading channels can improve performance.

These results show that the NN technique provides a positive and flexible platform in solving complex, non-linear, time-varying HDD problems. Therefore, this paper focuses on the NN technique to both classification and fusion process. The remainder of this paper is organized as follows: Methodology is described in Section II. Experiment and Results are detailed in Section III. Finally, conclusion is given in Section IV.

II. Methodology

The Neural Networks are used in this paper because of its ability to learn by examining an individual record and generating a prediction. The NN with pruning method is chosen, starting with a large network and remove the weakest units in the hidden and input layers as training proceeds. To prevent over training, we ensure that the number of samples is larger than the number of variables.

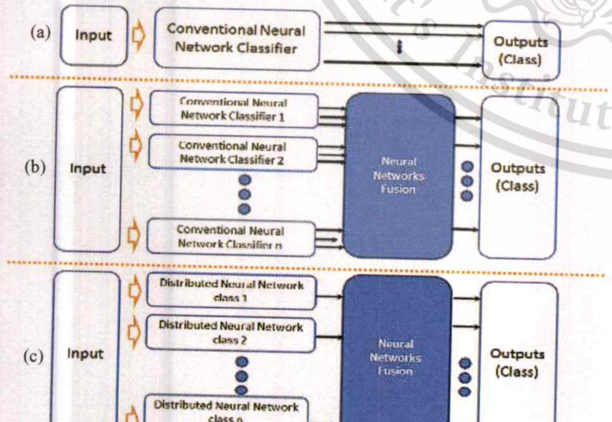


Fig. 2 The structure of (a) Conventional NN Classifier, (b) Conventional NN classifier with NN Fusion, (c) Distributed Classifier with NN Fusion.

Empirical data are collected from the samples of each probable outcomes of the Track Pitch Calibration. In this paper, Distributed Neural Networks classifier and NN fusion model are proposed to predict the track pitch calibration outcome of the empirical data.

The results and performance of the proposed method are being compared to the conventional NN classifier, with and without NN fusion as shown in Fig. 2

A. *Pre-Processing*

In the pre-processing step, the input data are collected from the known track pitch calibration parameters that perform good correlation by using the significant test as shown in Fig. 1. The 45 inputs parameters from the numerous affecting TPI calibration parameters are selected and obtained for the NN classifier model.

B. *Distributed NN Classifier*

The distributed NN classification conceptual is to perform the distributed tasks for a collective goal while the conventional NN classification performs oppositely.

Therefore, the distributed NN models are depended on the number of the classes, the more classes require the larger number of NN models, whereas the number of the conventional NN classification model does not depend on the number of the classes, only one NN model handles all classes.

With this basic concept, it improves a classification performance, capability and speed of the process, since one model is concentrated to one class or one task only.

The distributed NN Classifier works very well on both non-homogeneous and homogeneous inputs while conventional NN Classifier potentially has poorer performance when it operates on the non-homogeneous or the data of each class are not equal.

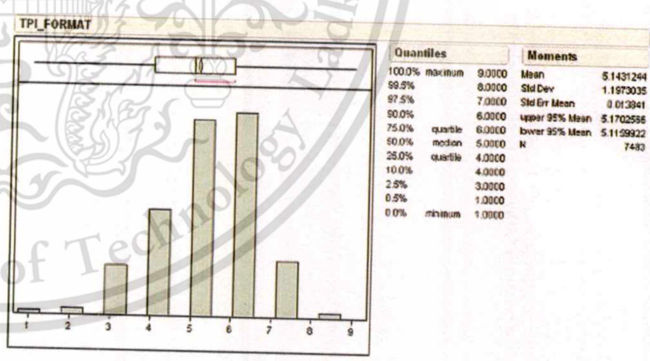


Fig. 3 Actual TPI distribution, the sample size of each TPI is unequal.

In practical, there is a difficulty to have the same amount of sample size for each class as shown in Fig. 3 and becoming a challenge to the classifier system, less sample size leads to improper weight adjustment and poorer results. As mentioned earlier, the distributed NN model concentrates to one class and this helps to improve the accuracy individually since the NN model is not being shared to other classes.

Nevertheless, the distributed NN method needs fusion process additionally to fuse the output from all classes for ranking the results. Though, it is still beneficial to the overall system performance when comparing to the others models and the results will be shown in following.

The data are first split randomly into training and test group with 60% and 40% respectively. The training data set is then proceeding through the conventional and the proposed NN classifiers.

There are 45 inputs parameters applying to the input of NN classifiers. The number of outputs of the classifiers is equivalent to the designed Product TPI. In this paper, there are 9 TPIs. Correspondingly, there are 9 outputs (classes) of the proposed NN Classifier. Since having 9 classes, the number of the distributed NN classification models obtains 9 models as well.

C. *NN Fusion*

In general, the main objective of the fusion process in this paper is to collect all the results from the distributed NN classifiers and also improve the performance of the systems as shown in many researches [4],[6].

The NN Fusion method is chosen since it is able to improve the system performance when comparing to conventional fusion methods such as border count fusion [6] as shown below

$$BC = \frac{\sum_{i=1}(r_i)}{m} \tag{2.}$$

Where m is the matchers and r_i is the racks assigned

The border count fusion usually uses the summation of the ranks assigned by individual matchers to calculate the final rank which is similar to Average calculation.

Conversely, NN is able to "learn" by adjusting the strengths (weight) (3.) and bias (4.) of these connections.

$$W^m(t+1) = W^m(t) - \alpha S^m (a^{m-1})^T \tag{3.}$$
$$b^m(t+1) = b^m(t) - \alpha S^m \tag{4.}$$

Where α is the learning rate, m indicates the layer number,

Technically, the weight of NN fusion is adaptively adjusted by the learning process via the error back propagate process as shown in equation (5.) and (6.) until they can approximate a function that computes the proper output for a given input pattern. These weight and leaning features become the key advantages to improve the NN fusion performance better than the conventional fusion.

$$\boldsymbol{S^M} = -2F^M(\boldsymbol{n^M})(\boldsymbol{t} - \boldsymbol{a}) \tag{5.}$$
$$\boldsymbol{S^m} = F^m(\boldsymbol{n^m})(\boldsymbol{W^{m+1}})^T \boldsymbol{S^{m+1}} \tag{6.}$$

Where $\boldsymbol{t}$ is the targets, $\boldsymbol{a}$ is the NN outputs,

Nevertheless, the NN fusion method requires a complex computation system and also complexity in implementation and maintenance aspect unlike the conventional fusion. However, these key drawbacks are not the key concerns for today anymore. With the current processing and resources performance of the computer, the concerns should be diminished.

III. Results and Model Established

A total of 7483 samples covering 9 possible outcomes of the current track pitch calibration method were used. These samples were selected from unequal samples size data set and were split at random in a 60% training set and a 40% test set.

The NN fusion model for distributed NN classifier contains 9 inputs, 1 hidden layer with 18 neurons and has 9 outputs at the output layer.

To measure the advantage of the proposed technique, the classification accuracy, test time and BER performance are compared to the conventional NN classifier with NN Fusion from multiple trials.

TABLE I

Results from Intelligent Classification Models of Each Class

Medthods	Conventional NN Classifier without NN Fusion				Conventional NN Classifier with NN Fusion				Distributed NN Classifier with NN Fusion			
TPI / Trial	Trial 1	Trial 2	Trial 3	Average	Trial 1	Trial 2	Trial 3	Average	Trial 1	Trial 2	Trial 3	Average
TPI 1	0.00%	0.00%	84.31%	28.10%	94.12%	64.71%	92.16%	83.66%	100.00%	100.00%	100.00%	100.00%
TPI 2	45.33%	62.67%	53.33%	53.78%	56.00%	62.67%	42.67%	53.78%	84.00%	70.67%	72.00%	75.56%
TPI 3	85.74%	83.91%	82.59%	84.08%	85.74%	87.89%	86.57%	86.73%	85.07%	89.05%	88.39%	87.51%
TPI 4	85.40%	88.25%	87.62%	87.09%	91.03%	90.00%	90.87%	90.63%	90.08%	91.67%	92.22%	91.32%
TPI 5	90.83%	91.13%	91.09%	91.02%	92.93%	93.32%	93.53%	93.26%	94.30%	95.16%	93.32%	94.26%
TPI 6	92.32%	91.82%	92.15%	92.10%	92.77%	93.56%	93.06%	93.13%	93.23%	94.30%	92.40%	93.31%
TPI 7	80.71%	78.93%	87.39%	82.34%	85.46%	83.09%	85.61%	84.72%	85.31%	84.57%	87.54%	85.81%
TPI 8	0.00%	7.94%	17.46%	8.47%	34.92%	31.75%	0.00%	22.22%	92.06%	87.30%	90.48%	89.95%
TPI 9	0.00%	0.00%	50.00%	16.67%	0.00%	50.00%	0.00%	16.67%	100.00%	0.00%	0.00%	33.33%

TABLE III

Comparison Results from Intelligent Classification Models

Intelligent TPI Classification Methods	% Accuracy			
	Trial 1	Trial 2	Trial 3	Average
Conventional NN Classifier without NN Fusion	87.33%	87.66%	88.80%	87.93%
Conventional NN Classifier with NN Fusion	90.43%	90.45%	90.32%	90.40%
Distributed NN Classifier with NN Fusion	91.60%	92.54%	91.70%	91.95%

Level	- Level	Difference	Lower CL	Upper CL	p-Value
CONVENTIONAL W/O FUSION	CONVENTIONAL W FUSION	0.0112254	-0.008203	0.0306540	0.2575
CONVENTIONAL W/O FUSION	DISTRIBUTED W FUSION	0.0071718	-0.012257	0.0266004	0.4694
CONVENTIONAL W/O FUSION	HDD	0.0061473	-0.022996	0.0352901	0.6793
HDD	CONVENTIONAL W FUSION	0.0050782	-0.025641	0.0357975	0.7459
DISTRIBUTED W FUSION	CONVENTIONAL W FUSION	0.0040536	-0.017668	0.0257754	0.7145
HDD	DISTRIBUTED W FUSION	0.0010245	-0.029695	0.0317438	0.9479

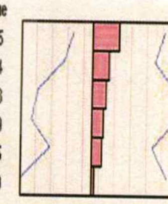


Fig. 3 Comparisons for each pair of each method using Student's t

A. *Results from Distributed Neural Network Classifier with NN Fusion*

This method allows the individual NN classifier to train on only one TPI (outcome). The Distributed NN Classifier outputs are then combined via the NN Fusion model.

The final aggregated accuracy shown that the classification accuracy of each TPI can then be improved significantly and reflected to overall prediction accuracy is at 91.60%, 92.54% and 91.70% for trial 1, 2 and 3 respectively and 91.95% in average.

The comparison results of all three different models, the Distributed NN Classifier with the NN Fusion and the Conventional NN Classifier with and without the NN Fusion are shown in Table. 1 and Table 2. We conclude that the Distributed Neural Network Fusion provides superior in accuracy of TPI classification, and also perform very good correlation to the current practice in HDD as confirmed by p-Value of the results of mean comparison shown in Fig. 3.

B. *Test time and BER comparison Results*

TABLE III
RESULTS OF TEST TIME REDUCTION AND BER COMPARISON

Intelligent TPI Classification Methods (Compare to Current Practice in HDD)	Test Time Reduction per One Head (%)	BER Comparison (Order)
Conventional NN Classifier without NN Fusion	-70%	-0.038
Conventional NN Classifier with NN Fusion	-60%	-0.029
Distributed NN Classifier with NN Fusion	-60%	-0.024

The intelligent TPI classification using NN requires input data from actual TPI calibration process in HDD at least 1 step, prior performing the analysis using NN Classifier with NN Fusion, whereas the current practiced in HDD has to calibrate for every step until reaching the last step of TPI selection.

The two steps proposed intelligent classification requires about 5 minutes of each steps to finish the classification process. Table. 2. shown the total test time of the current practice can reduce by 60%-70% per one head. The BER performance also validated in order to confirm no any trade off from the proposed method and there is insignificant BER different between existing and all classification methods however as a result showed that the proposed system accuracy is the best in class. This can imply that the proposed system potentially have less variation or better sigma values as opposed to other classifiers when facing the larger data.

IV. CONCLUSIONS

In this paper, we have studied and proposed the new intelligent track pitch classification using Distributed NN classifier with NN Fusion method to classify the TPI format accurately with certain test time reduction and no impact to the performance.

As a result, we showed that track pitch classification can be obtained through integration of NN classifiers and NN fusion. The final Distributed NN classifier with NN Fusion is able to accurately predict Track Pitch at 91.95% confidence level with 60%-70% test time reduction per one head and has small impact in BER performance.

In summary, we successfully constructed a new Neural Networks integration model. This model can then be applied to the Track Pitch Calibration process in the manufacturing of HDD to achieve a quick and accurate convergence in track pitch selection.

ACKNOWLEDGMENT

We are grateful to Western Digital for supporting on the Master scholarship and for providing data that is necessarily for the paper, and would like to thank the referees for their valuable comments, suggestions and help in this work.

REFERENCES

[1] Guoxiao Guo, Qi Hao, Teck-Seng Low, "Access system requirement for high track per inch hard disk drives", Asia-Pacific Recording Conference, 2000.

[2] Guoxiao Guo, Qi Hao, Teck-Seng Low, "A dual-stage control design for high track per inch hard disk drives", IEEE Transactions on Magnetics, Volume 37, Issue 2, Part 1, 2001.

[3] F. Hong, B. Hredzak, S. S. Ge, E. H. Ong, Z. He, and J. Zhang, "Adaptive Neural Network Compensation of Pivot Nonlinearity in Hard Disk Drives," The 46th IEEE Conference on Decision and Control New Orleans, LA, USA, Dec, 12-14, 2007.

[4] W. Tepin and Y. Kidjaidure, "Customer Failure Modes Prediction for Hard Disk Drive using Neural Networks Rank-Level Fusion," The 8th Electrical Engineering/ Electronics, Computer, Telecommunications and Information Technology (ECTI) Association, Thailand – Conference, 476-479, 2011.

[5] Wang, M.Y. Sheu, B.J, "Compact Neural Network Detector for Hard-Disk Drive Using Zero-Forcing Preprocessing," Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on.

[6] K. Pomoung and Y. Kidjaidure, "Rank-Level Fusion with Neural Network in Classification of Media Defect Patterns," The 3rd International Data Storage Technology Conference. Thailand – Conference, July 30 – Aug 1, 2010.