ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

PATTERN CLASSIFICATION USING MULTILAYER MIXTURE FUZZY NEURAL SYSTEMS

Ms. S.KUMARI, MCA., M.PHIL.,
LECTURER,
DEPARTMENT OF COMPUTER SCIENCE,
ARIGNAR ANNA GOVT ARTS COLLEGE FOR WOMEN,
WALAJAPET, VELLORE DIST., TAMILNADU, INDIA.
kumarishanmugam@gmail.com

ABSTRACT: Bayesian neural network model focus is on how to use the model for automatic classification, i.e. on how to train the neural network to classify objects from some domain, given a database of labelled examples from the domain. The original Bayesian neural network is a one layer network implementing a naive Bayesian classifier. It is based on the assumption that different attributes of the objects appear independently of each other. This work has been aimed at extending the original Bayesian neural network model, mainly focusing on three different aspects. First the model is extended to a multi-layer network, to relax the independence requirement. This is done by introducing a hidden layer of complex columns, groups of units which take input from the same set of input attributes. Two different types of complex column structures in the hidden layer are studied and compared. An information theoretic measure is used to decide which input attributes to consider together in complex columns. Also used are ideas from Bayesian statistics, as a means to estimate the probabilities from data which are required to set up the weights and biases in the neural network. Finally a query-reply system based on the Bayesian neural network is described. It constitutes a kind of expert system shell on top of the network. Rather than requiring all attributes to be given at once, the system can ask for the attributes relevant for the classification. Information theory is used to select the attributes to ask for. The system also offers an explanatory mechanism, which can give simple explanations of the state of the network, in terms of which inputs mean the most for the outputs.

Keywords: Artificial neural network, Bayesian neural network, Machine learning, Classification task, Dependency structure, Mixture model, Query-reply system, Explanatory mechanism.

1 INTRODUCTION

Automatic classification deals, i.e. how to find out which class an object belongs to, given some of its properties. Classification is a very general problem, and there are several tasks from a wide range of domains which can be cast into classification tasks. This includes character

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

or speech recognition, fault detection, fault diagnosis, process control, action planning, and much more. More specific examples of where an automatic classification system has practical use includes assistance in making medical diagnoses based on symptoms, and product quality control in e.g. chemical industry. Machine learning, the main perspective has still been that of a neural network. There are certain appealing qualities in the neural network approach, like the simple and mainly local computational structure, the possibility of parallel implementation if such hardware is available, and some robustness to noise and errors.

1.1 THE CLASSIFICATION TASK

Classification is a set of attributes, or features, of an object, and we want to decide to which of a number of classes it belongs. The given attributes can be gathered into an input vector x. The goal is to train the system to perform the classification, given a set of previous sample patterns, each consisting of a vector of attribute values and the corresponding class label. In some cases the classification domain is deterministic in that each possible input pattern corresponds unambiguously to one class. More common though is the situation when the classes "overlap", i.e. samples from two or more classes may look exactly the same, and it is only possible to talk about the probability of a pattern belonging to a certain class. If it is not sufficient to produce probabilities as output, but one class has to be selected, it is often reasonable to select the class with the highest probability, since this will give the highest proportion of correct answers.

Important concept is that of the decision surfaces of a class assignment function. These are the decision boundaries between different classes in the input space. Different classification methods have different limitations on the form of these boundaries, which can give good hints about for what type of problems different methods is most suitable. For example, many methods give (linear) hyperplanes as decision surfaces, or some combination of a finite number of hyperplanes.

The curse of dimensionality is a number of dimensions of the input space (i.e. the number of input attributes) increases, the number of possible input vectors increases exponentially. Any method powerful enough to express any class distribution over this space will necessarily have exponentially many free parameters, whereas the number of training samples is usually very limited, and thus not sufficient for estimation of all the free parameters.

1.2 MACHINE LEARNING METHODS

There is a vast number of methods designed to perform classification in various domains. The methods differ much in their background. Some are developed

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

in the context of neural networks, others in genetic algorithms, heuristic search, case based reasoning, statistics, or logics. Sometimes this difference in background has resulted in real differences in the methods. In other cases apparent differences are merely a matter of notation, and the basic calculations are the same or very similar. Also, some methods are tailored specifically for a certain type of domain, while others are more generally applicable.

1.2.1 CASE BASED METHODS

One of the simplest ideas is just to store all encountered samples and their class labels in a dictionary. This is the basis for Case Based Learning. This dictionary can then be used to classify at least samples identical to the stored ones. New samples not in the dictionary are classified as the same class as the most similar stored sample. This gives a Nearest Neighbor classifier.

1.2.2 LOGICAL INFERENCE

There is a large group of methods, popular within artificial intelligence, which represents "knowledge" as relations between logical variables. Binary input attributes are treated directly, while numerical attributes are coded with suitable predicates. For example, a variable might be set to "true" if a real valued input attribute falls within some interval. To learn logical representations from examples, via Rule Induction. One approach is to represent class descriptions as Logical Conjunctions, i.e., expressions which are true if all of a set of conditions on the input attributes are fulfilled. To learn such a representation that distinguishes a certain class, one can start with either a very general or a very specific expression. Conditions are then added to or removed from the expression as new samples arrive, to gradually improve its discriminatory power.

1.2.3 STATISTICAL METHODS

Logical classification rules may be appropriate in deterministic domains, where each input pattern can belong to only one class. If several classes can have the same feature vector, the best one can do is to calculate the probability of the different classes, and select the most probable one. A common goal of the statistical methods is to use the training samples to estimate the probability distribution over the domain, and then use this distribution to calculate the probabilities of the classes given a specific input pattern. When estimating (continuous) probability distributions from data, there is a distinction between parametric and non-parametric models.

A middle ground between parametric and non-parametric models is semi-parametric models. They contain some model assumptions to make the problem tractable, but are general enough to include a large number of different distributions. These statistical methods

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

work best when the sample space is of relatively small dimensionality and "homogeneous", i.e. when all dimensions are measured in the same units and are of comparable relevance for the classification.

The Naive Bayesian Classifier is that all input attributes are independent (or actually, independent given the class). Then the probability distribution over the domain can be written as a product of the marginal distributions over the attributes. These marginal distributions have much fewer parameters, and are thus much easier to estimate from the training data. The independence assumption amounts to assuming that each input attribute gives some evidence for or against each class, which can be considered separately from the evidence contributed by the other attributes.

1.2.4 ARTIFICIAL NEURAL NETWORKS

The main idea behind an Artificial Neural Network is to use several simple computational units, connected by weighted links through which activation values are transmitted. The units normally have a very simple way to calculate new activation values given the values received through the connections, for example summing their inputs and feeding it through a monotonous transfer function. To use a neural network in a classification task, the pattern to classify is typically fed into the network as activation of a set of input units. This activation is then spread through the network via the connections, finally resulting in activation of the output units, which is then interpreted as the classification result. Training of the network consists in showing the patterns of the training set to the network, and letting it adjust its connection weights to obtain the correct output.

2. THE BAYESIAN NEURAL NETWORK MODEL

The one-layer Bayesian neural network is based on the idea of a naive Bayesian classifier. The network is trained according to the Bayesian learning rule, which considers the units in the network as representing stochastic events, and calculates the weights based on the correlation between these events. The activity of a unit is interpreted as the probability of that event, given the events corresponding to already activated units. The Bayesian neural network can be used as an auto associative as well as a hetero associative memory. In a classifier context, where features are considered as input and classes as output. However, it is important to remember the original symmetry with respect to features and classes; we can feed any known attributes into the network, and get out estimates of the probability of the remaining attributes, regardless of whether they are to be interpreted as classes or features.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

2.1 THE NAIVE BAYESIAN CLASSIFIER

The approach to classification taken here is to calculate the probabilities of the different classes given some observed evidence. If the objective is to make as few classification errors as possible, the class with the highest probability should be selected as the classification result is called the optimal Bayesian classification.

In most situations it is more natural to deal with P(x |y), the probability of the attribute value x of a certain classy. There may be some information on what each class "looks like", i.e. about the distribution of x for each class. To calculate P(y |x) Bayes theorem for conditional probabilities can then be used:

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)} \quad (2.1)$$

Suppose now that we are given the values of N input attributes, x ={x1,x2,...xN},which can be considered independent both unconditionally and conditionally given y. This means that the probability of the joint out come x can be written as a product,

$$P(x)=P(x1)\cdot P(x2)\cdots P(xN) \quad (2.2)$$

and so can the probability of x within each classy,

$$P(x|y)=P(x1|y)\cdot P(x2|y)\cdots P(xN|y) \quad (2.3)$$

2.2 THE ONE-LAYER BAYESIAN NEURAL NETWORK

The usual equation for signals propagating in a neural network, with units that sum their inputs, is

$$sj = \beta j + \cdot \sum wji\ oi \quad (2.7)$$

where sj is the support value of unit j, βj is its bias, oi is the output from unit i and wji the weight from i to j. The support value of each unit is fed through a non-linear transfer function, to produce the output activity of the unit:

$$oj = f(sj) \quad (2.8)$$

2.3 THE INDEPENDENCE ASSUMPTION

To derive the network model above, we had to assume independence between the different attributes, both unconditionally and conditionally given each classy, Eqs. (2.2) and (2.3). Complete independence in this way is of course usually not the case in real situations Of the two relations the first one, Eq. (2.2), is actually less important than it might seem. To see this, note that Bayes theorem (2.1) can be written in the alternative form (which is the same as in Eq.(2.16))

$$P(y|x)=\frac{P(y)P(x|y)}{P(x)}=\frac{P(y)P(x|y)}{\sum yP(y)P(x|y)} \propto P(y)P(x|y)$$

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

Since the denominator is the same for all classes y, an error in it gives the same over-or under-estimation for all classes. This means that if we are only interested in the probability ordering of the classes, this is not affected at all. Furthermore, if we want the actual class probabilities we can always get them by normalization (since the probabilities P(y |x) must sum to 1 overall classes), thus compensating for this error.

The second relation, Eq. (2.3), is more important since it directly affects the ordering of the classes, which of course is serious in a classification system. There is an important special case in which this relation holds though. It is when we can express each class with one single pattern, a prototype, and the instances are considered as noisy versions of this pattern. The prototype may have graded values of its attributes, representing the frequencies of different attribute values in the population.

Some situations where this condition is likely to occur is when classifying an object (like an insect or plant) from some high level features of the object, where typically objects with similar feature vectors belong to the same class. Thus the independence assumption is related to the assumption that similar samples are likely to belong to the same class, where similar means that they have many attributes in common. If the same class can look in two entirely different ways (e.g. it is a mouse either if it has four small legs and gray fur, or if it has a button and a cord, but it will never have a combination of these appearances), it cannot be described by only one prototype. Such cases may be less common in this kind of situation, but there are examples of domains where similar things can occur.

Even in less extreme situations, this independence assumption is what limits the operation of the one-layer Bayesian neural network the most. Therefore it is necessary to study some multi-layer extensions of the model that can help overcoming this limitation.

2.4 THE MULTI-LAYER BAYESIAN NEURAL NETWORK

When all input attributes are not independent, the naive Bayesian classifier is not appropriate. If it were possible we would like to estimate the whole distribution P(X |Y) directly, but this is impossible already for moderate numbers of input attributes. The solution we concentrate on here is to make something in between. Those attributes that are dependent must be considered together, but hopefully every attribute is not dependent on all others. In other words, we will try to utilize the independencies that can be found in the problem domain at hand. We will consider two different ways to handle dependencies. In the first, which is the simplest, the input attributes are partitioned into groups which are independent. Each group can be considered as one complex attribute, and

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

the joint distribution over it is estimated. Since the different groups are independent of each other, their probability distributions can then be combined as before. The second method is more involved. It consists of trying to estimate a dependency graph between the input attributes, and uses this graph to calculate the joint probability distribution over the whole input space.

3. GRADED INPUTS AND CONTINUOUS VALUED ATTRIBUTES

Although they both concern graded values in some way, these two situations are very different in nature. The latter case concerns probabilities of discrete inputs (for example the probability that an animal has a tail), whereas in the former the inputs are themselves from some continuous domain (for example the length of the tail). In the Bayesian neural network model the activity of a unit is interpreted as a probability, so it is not appropriate to code a continuous attribute directly to an interval between zero and one and feed this into one unit. However, as will be seen below, the same mechanism that can handle arbitrary probabilities as inputs can be used indirectly to handle continuous valued attributes as well.

3.1 PROBABILITY DISTRIBUTIONS AS INPUT

Thus suppose we make a (non-conclusive) observation of an attribute, which leaves us with a distribution P(Xi) over the corresponding variable Xi. Then we can express the probability of the class y given this distribution, using the probabilities of y given the individual outcomes xii · of Xi, as

$$P(y|Xi)=\sum P(y|xii\cdot)\cdot P(xii\cdot|Xi)$$

I

The expression P(xii · | Xi) denotes the probability of the outcome xii · according to the current distribution of Xi (and is thus not the same as the prior probability of the outcome).

3.2 UNCERTAIN EVIDENCE

One important purpose of the above treatment of graded input is to be able to handle continuous valued attributes. How this is done is shown in the next section. However, if the network is to be able to handle uncertain evidence from the user, then there are some further details which need to be addressed.

If there are no complex columns in the network, the introduction of uncertain evidence is quite straightforward. One objection is than in a typical situation the value of uncertainty from the observation is not the same uncertainty one is supposed to feed into the network. The network expects the probability of the feature in question given the possibly non-conclusive observation of it. But if a measurement is done with some partially unreliable instrument, what is known is instead typically the probability of error for the

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

instrument, which means the probabilities of the instrument showing certain results given the real feature. What has to be done is, once again, to "invert" the conditioning via Bayes theorem, Eq. (2.42):

$$P(\text{feature}|\text{observation}) \propto P(\text{observation}|\text{feature}) \cdot P(\text{feature}) \quad (3.15)$$

A more severe problem is perhaps how to use graded inputs in connection with complex columns. For completely determined input values of some primary attributes, the activity of a complex unit can be calculated by multiplying the activities of the units it represents a combination of. This would also be a good idea in the case of graded input, if the primary attributes were independent. But the very reason to create the column for a set of attributes is just that they are not independent.

With at least one specific model of the uncertainty in the observations, this problem has a simple solution. Assume that two features,f1 andf2, (which are dependent) are measured with two measuring devices, giving the resultse1 ande2. Suppose now that the probability of error in each of the measuring devices are independent of each other, i.e. an uncertainty in one measurement does only depend on the feature it was supposed to measure, and not on the other feature. Then what is known is the probabilities P(e1 |f1) and P(e2|f2). But since these observations are not directly dependent on other features than the ones they try to measure, we can write

$$P(f1,f2|e1,e2) \propto P(e1,e2|f1,f2)P(f1,f2) = P(e1|f1)P(e2|f2)P(f1,f2) \quad (3.16)$$

So if the user tells the system the probabilities of making the current observations given the possible values of the attributes, then the system can combine this information and feed it into the hidden layer of the network, just using multiplication and normalization.

In the general case, when there is no specific model of the noise, or the uncertainty, the problem is harder. The trouble lies actually already in the problem formulation. We would like to calculate the distribution of AB from the marginal distributions A and B. But if two attributes are not independent, it is not sufficient to know their marginal distributions, to be able to calculate their joint distribution. One the one hand, this would require the user to specify the joint distributions for every complex column in the network. On the other hand, the exact complex column structure in the network is normally not accessible to the user (normally the user do not want to bother about the internal structure of the network). Further, if the probability distribution over an attribute given an observation is hard to specify for the user, then the joint probability of two or more attributes is worse.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

3.3 CONTINUOUS VALUED ATTRIBUTES

When graded input occurs, it is as probabilities over the attributes. Here we will handle the situation when the evidence we want the network to use comes from a measurement with a result in some continuous interval. It is not possible to code the continuous attribute directly as graded input activity of one unit in the network, since this would then be interpreted as a probability. For example, the result of giving a fifty percent probability for a binary feature will always be an equal mix of the situations when the feature is present and not, whereas a medium sized object does not necessarily have properties exactly between those of a large and a small object.

The same mechanism that allows uncertain inputs in the Bayesian neural network model, can however be used in directly to handle continuous valued attributes. The idea is to transform the continuous variable into a number of discrete variables, which can be used directly by the Bayesian network. The probability density function over some continuous variable Z can be approximated by a finite sum (although it is a density function, it is here denoted by P(z)):

$$p(z) = \sum_{i=1}^{n} P(vi)P(z|vi)$$

This can be seen as a partition of P(z) into n sub-distributions P(z | vi), each with probability P(vi) of being the "source" of z. Expressed in another way, random numbers with the distribution P(z) can be generated by first selecting among the sub-distributions with probabilities P(vi), and thereafter generating the number from the selected distribution P(z |vi).The above equation also defines P(vi |z), the probability of each component vi being the source of a given z, as

$$P(vi|z) = \frac{P(z|vi)P(vi)}{P(z)} \propto P(z|vi)P(vi) \quad (3.18)$$

where instead of keeping track of P(z), we can again use normalization (over i) of the right hand side.

At least in the case of a one-dimensional variable Z such a localized mixture model can be thought of as giving rise to a kind of soft interval coding. Each value in the interval will "belong" to different degrees to the different components, which thus makes a soft division of the interval between themselves. Of course an analogy of this holds also in the multi-dimensional case, where each component dominates some local piece of the input space. This is reminiscent of the coding used in the context of fuzzy sets, although the interpretation of activities is here in terms of probabilities rather than of fuzzy membership.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

3.4 EMPIRICAL EVALUATION

To evaluate the different suggested strategies for handling continuous attributes in the Bayesian neural network, they have been tried out on a few test databases. Several different parameter settings and variations of the methods were compared. The variance can be handled in some different ways. As mentioned above, since estimation of the full covariance matrix gives a large number of parameters, this may give overtraining and impaired generalization. Therefore it is often useful to constrain the form of the covariance matrix in different ways. Three different strategies are evaluated here: use of a single variance parameter (i.e. the same variance in all directions), a diagonal covariance matrix (i.e. the variance can have separate values for each input dimension), and the full covariance matrix.

The direct use of a single variance parameterσ2 is not appropriate unless all input dimensions are measured in the same units. If the different dimensions represent completely different quantities, as is often the case in the kind of classification problems with mixed inputs considered here, the variance will probably be radically different along different axes. One way to compensate for this is to apply whitening to the data, which means that all continuous attributes are normalized to have unit variance in all directions (and possibly mean value at zero, but this does not change anything here).

One version of whitening transforms the whole space with the help of a principal component analysis, and thus in addition removes all non-diagonal elements in the covariance matrix. However, when a single variance parameter was used here, each attribute was transformed separately to get unit variance. When a diagonal covariance matrix is used, no additional scaling of the attributes are necessary. The difference from the single variance parameter case is that now the attributes can be scaled locally (i.e. for each mixture component) rather than globally. The diagonal covariance matrix is estimated in the same way as the full covariance matrix, except that all non-diagonal elements are kept fixed at zero.

Both the standard and the Bayesian version of the EM algorithm are tested. Both versions are also tried when constraining the probabilities of mixture components to be equally probable. The goal is to compare the two methods for preventing the component densities from collapsing to zero probability (or to zero variance).

One further variation to consider is whether training of the mixture models via the EM algorithm is to be done unsupervised or in a class dependent manner. When the class dependent method is used, each component function is assigned to one of the classes, and during training it only responds to samples from that class. This effectively causes the component functions to be split up in

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

groups which model the different class conditional densities separately. No other adjustment of the EM algorithm is necessary to account for the classes, when the class variable is discrete. [6] discussed about Improved Particle Swarm Optimization. The fuzzy filter based on particle swarm optimization is used to remove the high density image impulse noise, which occur during the transmission, data acquisition and processing. The proposed system has a fuzzy filter which has the parallel fuzzy inference mechanism, fuzzy mean process, and a fuzzy composition process. In particular, by using no-reference Q metric, the particle swarm optimization learning is sufficient to optimize the parameter necessitated by the particle swarm optimization based fuzzy filter, therefore the proposed fuzzy filter can cope with particle situation where the assumption of existence of "ground-truth" reference does not hold. The merging of the particle swarm optimization with the fuzzy filter helps to build an auto tuning mechanism for the fuzzy filter without any prior knowledge regarding the noise and the true image. Thus the reference measures are not need for removing the noise and in restoring the image. The final output image (Restored image) confirm that the fuzzy filter based on particle swarm optimization attain the excellent quality of restored images in term of peak signal-to-noise ratio, mean absolute error and mean square error even when the noise rate is above 0.5 and without having any reference measures.

4 CONCLUSIONS

The goal of this work has been to construct a very general and robust classification method which is directly applicable to real world problems with all the complications this implies in terms of limited training data, noisy inputs, and poorly understood interactions.

To this end, inspiration has been taken from several areas: statistics for the probability estimations and statistical inference, information theory to detect the dependency structure of the domain, and mixture models to incorporate continuous valued attributes. The components are combined into one consistent framework, and made to work together, while still maintaining the simplicity in computational structure inherent in the neural network approach.

Thus, most of the components used in this neural network model build on well known methods and algorithms. Still there are some parts which could be considered more original, and which may find use outside of the Bayesian neural network as well. One such thing is the Bayesian version of the expectation maximizationalgorithmwhichispresentedin section3.3.2. Standard EM has some problems, for example with component densities getting zero probability or zero variance. There are several possible adjustments of the method which can

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

overcome one or another of these problems. The method used here, Bayesian estimation of the parameters of the component functions, is a single quite natural variation which solves many of the major problems at once.

Also somewhat original is the use of mixture models in dependency graphs. Various probabilistic graphical models, e.g. belief networks, have been used to propagate discrete, Gaussian, or other simple distributions, through a dependency graph. The mathematics behind the overlapping complex columns in the Bayesian neural network is very similar to that used in probabilistic graphical models. Here we have shown that it is possible to use Gaussian mixture models to code continuous attributes and to combine these mixtures in dependency graphs (i.e. by using overlapping columns of continuous attributes). Since mixture models are very general models, which can be used for most distributions when no other simple model is known, this may also be very useful in the context of belief networks or other probabilistic graphical models. One important question, which has not been treated in this work, is how to select the number of component functions in a mixture model. Here all mixtures of the same input dimension are more or less arbitrarily given the same number of components. This is however an important parameter, and there should be some principled way of selecting it; especially since there may be a quite large number of different mixtures in a single Bayesian neural network, and each potentially requires a different number of components.

In general it is not a simple task to find a good number of components, but there may be a few possible approaches. It is a future task to look at the different possibilities of doing this in the Bayesian neural network.

5.REFERENCES

[1] Abend K., Harley T. J., and Kanal L. N. (1965). Classification of binary random patterns. IEEE Trans.Information Theory11:538–544.

[2] Abramson N. (1963). Information theory and coding. McGraw-Hill, New York.

[3] Ackley D. H., Hinton G. E., and Sejnowski T. J. (1985). A learning algorithm for Boltzmann machines. Cognitive Science9:147–169.

[4] Barto A. G., Sutton R. S., and Andersson C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Trans. Systems, Man, and Cybernetics13:834–846.

[5] Battiti R. (1994). Using mutual information for selecting features in supervised neural net learning.

[6] Christo Ananth, Vivek.T, Selvakumar.S., Sakthi Kannan.S.,

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

Sankara Narayanan.D, "Impulse Noise Removal using Improved Particle Swarm Optimization", International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE), Volume 3, Issue 4, April 2014,pp 366-370

[7] BellmanR. (1961). Adaptive Control Processes: A Guided Tour. PrincetonUniversityPress, New Jersey.

[8] Bishop C. M., Svens´en M., and Williams C. K. I. (1997). GTM: A principled alternative to the selforganizingmap.

[9] In Mozer M. C., Jordan M. I., and Petche T. (eds.), Advances in Neural Information Processing Systems, volume 9, pp. 354–360.MIT Press, Cambridge, MA. Proc.of Neural Information Processing Systems, Denver, Colorado, December 3–5, 1996.

[10] Block H. D. (1962). The perceptron: A model for brain functioning. Reviews of Modern Physics 34:123–135.

[11] Blum A. L. and Rivest R. L. (1992). Training a 3-node neural network is NP-complete. Neural Networks 5:117–127.

[12] Breiman L., Friedman J. H., Olshen R. A., and Stone C. J. (1984). Classification and Regression Trees.

[13] Wadsworth, Belmont, CA.

[14] Broomhead D. S. and Lowe D. (1988). Multivariable functional interpolation and adaptive networks.

[15] Complex Systems2:321–355.

[16] Brown D. T. (1959). A note on approximations to discrete probability distributions. Information and Control2:386–392.

[17] Bruner J. S., Goodnow J. J., and Austin G. A. (1956). A Study of Thinking. John Wiley, New York.

[18] Chandrasekaran B. (1971). Independence of measurements and th he mean recognition accuracy. IEEE Trans. Information Theory17:452–456.

[19] Chow C. K. (1966). A class of nonlinear recognition procedures. IEEE Trans. Systems, Science, and Cybernetics2:101–109.

[20] Chow C. K. and Liu C. N. (1968). Approximating discrete probability distributions with dependency trees. IEEE Trans. Information Theory14:462–467.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 4, February 2018

[21] Cichocki A. and Unbehauen R. (1993). Neural Networks for Optimization and Signal Processing.John Wiley, New York.