IJARST

International Journal of Advanced Research in Science and Technology

journal homepage: www.ijarst.com

ISSN 2319 – 1783 (Print)

ISSN 2320 – 1126 (Online)

Training and classification of Epilepsy Detection using EEG

G.Rajendra Kumar[1]*, Kuda Neswara Rao[1] and Santhosh Kumar Dhavala[2]

1 Computer Science & Systems Engineering Départment, Andhra University, Visakhapatnam, India.
2 Department of Physics, School of Engineering, GVP Degree and PG College Visakhapatnam, India.
****Corresponding Author's Email: rajendragk@rediffmail.com***

A R T I C L E I N F O

Article history:
Received 14 Jun. 2013
Accepted 26 Jun. 2013
Available online 30 Jun. 2013

Keywords:
EEG,
Entropy,
Neuron,
Images,
Classification,
Epilepsy.

A B S T R A C T

An electroencephalogram (EEG) is a test used to detect abnormalities related to electrical activity of the brain. This procedure tracks and records brain wave patterns. Small metal discs with thin wires (electrodes) are placed on the scalp, and then send signals to a computer to record the results .The automated detection of epilepsy using EEG signals. Epilepsy is nothing but, the human brain that affected by the syndrome during the fix. We cannot cure the problem fully. But we can prevent from the epileptic syndromes. Use of this paper we can identify the Epileptic syndrome using EEG signals. If someone has affected from the problem means the EEG report shows from the wave signals which comes in that machine. Affected brain signal waves should be differs from the normal brain. Based on the entropy value we can say the person is to be effected with epilepsy or not. Hence this analysis not only enables the correct diagnosis of the patients with epileptic seizures but also predicts the probability of risk in future for the normal persons.

1. Introduction to Epilepsy:

Epilepsy is the one of the most dangerous disease in the human body. There are different types of Epilepsy are effecting by the people. Approximately there 1% of people are suffering with the epilepsy in the world. Here to detect the epilepsy we are using the electroencephalogram (EEG) signal. Epilepsy is characterized by the occurrence of recurrent seizures in the EEG signal. Epilepsy can develop in any person at any age. Almost 1% of people will develop epilepsy during their lifetime. That's 60million people worldwide. In other words, out of a 60,000 person stadium, about 500 will have epilepsy.

1.1. Detection of Epilepsy with traditional Methods:

Detection of epilepsy with traditional methods of analysis is complicated and very time consuming process. There are many automated epileptic EEG detection systems are developing from years. The electroencephalogram (EEG) signal is used for the purpose of the epileptic detection and it is condition related to the electrical activity[1] of the brain. Epilepsy is characterized by the occurrence of the recurrent seizures in the EEG signal.

With the advantage of the technology we are using for detection is possible to store and process the EEG signal digitally. The digital signal I.e. EEG is the data that can be led to an automated seizure detection system that can detect the seizures present in the EEG data. Using this EEG, we can automatically reduce the time for the detection due to automation. There are so many systems for epilepsy detection that have been developed for different approaches. In this present work, we are discussing the automated epileptic EEG detection system using two different neural networks, namely, Elman network and probabilistic neural network using a time-domain[2] feature of the EEG signal called approximate entropy. ApEn is a recently formulated statistical parameter to quantify the regularity of a time series data of physiological signals.

2. Existing System:

In this Existing system, the EEG signal as an input to a learning quantization network and this was proposed by Pradhan et al and the other new neural network model called LAMSTAR network was proposed by Nigam and Graupe and the two time-domain attributes of EEG namely, relative spike amplitude and the spike rhythm city have been used as inputs for the purpose of the detection of epilepsy. And finally uses a back propagation neural network with period gram and autoregressive features as the input for the automated detection of epilepsy. Finally the proposed techniques are used for detecting the epilepsy according to their approach. All the above techniques may not get the accurate values and may not detect the epilepsy accurately.

3. Proposed System:

In this proposed system, we are using the Artificial Neural Network that will increase the computational complexity. The high overall detection accuracies achieved with this system surpasses its disadvantage as in any automated seizure detection[4] system; the detection of the seizure with high accuracy is of primary importance. Approximate Entropy shows clear discrimination between the normal and epileptic EEG signals.

The optimum Approximate Entropy obtained based on this data may not hold good for a general case. Hence, using a linear separator with known Approximate Entropy parameter values may not give good results in situations where a large number of different subjects are involved. This problem will not arise in the proposed ANN-based method as it has performed well irrespective of the Approximate Entropy used.

In this present work, we are describing that the Approximate Entropy having very good characteristics such as robustness in the characterization of the epileptic patterns and low computational burden. Hence, an automated system using Approximate Entropy as the input feature is best suited for the real-time detection of the epileptic seizures.

The proposed system is based on two types of EEG, namely, EEG signals of awake and epileptic subjects. It can be made more robust by acclimatizing it to the other manifestations of EEG like sleep EEG.

In this present work, we are finally using two different types of neural networks are considered. To detect the Epilepsy 100% ApEn is used in this proposed[5] system using neural networks. The will show that the overall accuracy values as high as 100% that can be achieved in this proposed system.

4. Statement of the Problem:

The EEG is not necessarily a “harmless” investigation. Although it may be relatively non-invasive and therefore physically harmless, it may be harmful in terms of its interpretation. An unsatisfactorily recorded EEG, undertaken[6] by technical staff who have never been trained to perform such investigations in children, which is reported by a clinician who has not been taught the normal maturational as well as abnormal appearances of children’s EEGs. may result in inaccurate diagnoses of both epilepsy and the specific epilepsy syndrome. Clearly, this may have serious medical, psychological, and social consequences. Unfortunately, appropriately trained technical staff and paediatric neurologists or clinical neurophysiologists are not ubiquitous, particularly within the UK.

The main subject of this work is the human brain through the analysis of electromagnetic signals coming from EEG recordings. These techniques have been largely used in the past in order to detect brain activity for both research and clinical purpose. In particular, we focus on the emerging class of methods which is based on Nonlinear Dynamical System analysis. In fact, recently, new studies support the idea of the brain as a complex neural network in which nonlinear synchronization among different regions make possible the formation of complicated temporal patterns dynamics. It follows that the brain function can be described by a state-space model in which dynamical properties can be modeled by nonlinear manifolds. Identification and characterization of these manifolds represents one of the goals of this works. We consider the analysis of the from different types of brain dynamics due to both normal and abnormal activity. Normal patterns are interesting[7] because such a study can lead to a better understanding of brain functionality. Among abnormal brain function, we consider the case of epilepsy which is a common neurological disorder that develops irregular and uncontrolled electrical activity. This disease seems to be important because its deviant behavior reflects a low dimensional pattern formation. Implementing efficient algorithms for epilepsy detection and prediction is an important goal in the field of neuroscience.

5. Methodology:

5.1. Approximate Entropy:

Entropy Measures signal complexity. EEG with low entropy is due to a small number of dominating processes. EEG with high entropy is due to a large number of processes. Relatively simple measure of complexity and system regularity[8]. Quantifies the predictability of subsequent amplitude values of the EEG based on the knowledge of previous amplitude values.

As a relative measure depends on three parameters

1. The length of the epoch.
2. The length of the compared runs.
3. The filtering level.

Approximate entropy and Shannon entropy are two entirely different measures. Approximate entropy measures the predictability[9] of future amplitude values of the EEG based on the one or two previous amplitude values, increasing anesthetic concentrations are associated with increasing EEG pattern regularity.

EEG approximate entropy decreases with increasing anesthetic concentration. At high doses of anesthetics, periods of EEG silence with intermittent bursts of high frequencies occur. For example median EEG frequency method fails to characterize concentrations because of these bursts. Brain's EEG approximation Entropy value is a good candidate for characterizing different extents of cerebral ischemic injury. First, in the early stage of ischemia, the EEGs' approximate entropy difference between ischemic region and normal region increase.

Second, after ischemia 18 minutes, the approximate entropy of ischemic region become lower than that before ischemia (normal state), which may indicate an emergent injury being induced. Last, the approximate entropy of ischemic region (left brain) is lower than that of normal region (right brain). In this figure, the filtered EEG-signals from the left hemisphere compared to mean signal from that hemisphere are shown. As it can be seen the methods find the local brain activations and spatial artifacts from the second EEG-channel.

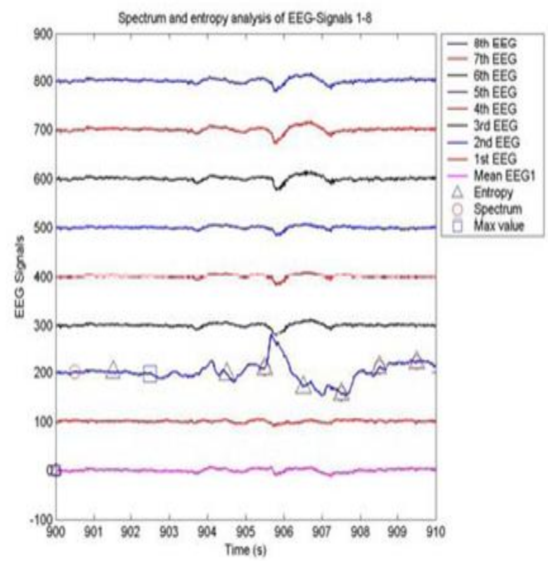

Figure1.1: Entropy analysis of EEG Signals

5.2 Elman Neural Networks:

Elman networks are often adopted to identify or generate the temporal outputs of nonlinear systems. It is well known that a recurrent network is capable of approximating a finite state machine and thus can simulate any time series. So recurrent networks are now widely used in fields concerned with temporal problems. In published literature[10], however, all the initial weights of recurrent networks are set randomly instead of using any prior knowledge and thus the trained networks are vague to human and their convergence speed is slow. In addition, the temporal generalization capability of simple recurrent networks is not so good. These two major problems make the applications of recurrent networks with temporal identification and control of systems more difficult. It is a special type of recurrent neural network. It is a two layered back propagation network with a feedback connection from the output of the hidden layer to its input. This feedback connection allows EN to recognize and generate temporal patterns, as well as spatial patterns. Following figure shows the Elman neural network structure.

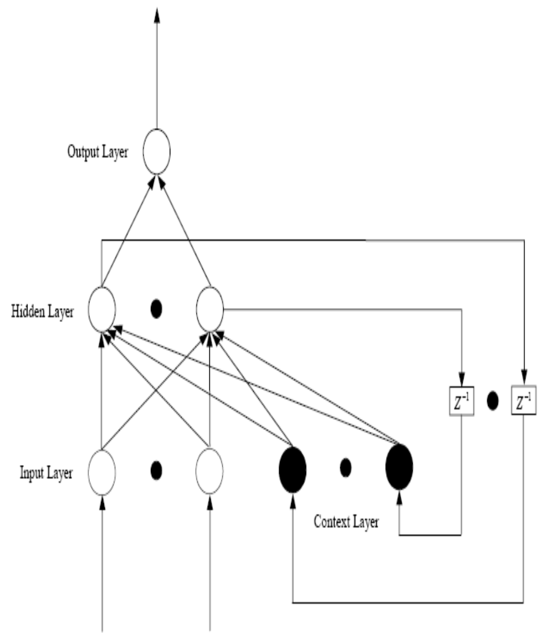

Figure1.2: Structure of Elman Neural Networks

6 Artificial Neural Networks:

Without going into detail about the history, let's just say neural networks were invented in the sixties by mathematicians (back in those days, computer[11] scientists were often mathematicians; my how things have changed) looking for ways to model processing in the human brain at the lowest level. They took what they knew about a human neuron as a model for creating an electronic neuron.

6.1 The Neuron:

These researchers started by boiling down to how they viewed a neuron as working in light of existing knowledge. A neuron, they said, may have many inputs ("dendrites") that a hooked into the single output ("axon") of another neuron. Signals[12] received by some dendrites would tend to activate the neuron, while signals on others would tend to suppress activation. Figure 1.2 below shows one kind of model of this:

Many essays on how neural networks work are available online. I've found all of the ones I've skimmed difficult to follow; again, because I'm not a mathematician. Let me skip the proofs, then, and summarize the exact algorithm I used in my own neurons for the "thinking" done by a single neuron:

```
Public Sub Think()

Dim Sum As Single, D As Dendrite

For Each D In Dendrites

D.Value = D.FromNeuron.AxonValue

Sum += D.Value * D.Weight

Next

AxonValue = -2 / (1 + Math.Exp(2 * Sum)) + 1

End Sub
```

To be sure, I've left out some extra lines of code that are to deal with odd cases that are not essential for this discussion.

I should also point out that there are a variety of ways of creating a neuron. Some use Boolean values for inputs and outputs, while others use linear (floating point) values. Some range their values from 0 to 1, others from -1 to 1, and still others from -infinity to infinity. The model I chose is referred to as having a "sigmoid" output function.

This means that it uses linear input values that range from -1 to 1 and whose outputs, which after summation could be anywhere from -infinity to infinity, are instead "crunched" down to fit in a range from -1 to 1. They are not truncated, but are squashed using the equation on the last line of the *think* () function above.

6.2 Pattern Matching:

Here is where a lot of introductions stop short. Let me explain what it means for one of these neurons to "know" something before explaining how they work in concert with other such neurons.

The goal of a neuron of this sort is to fire when it recognizes a known pattern of inputs. Let's say for example that the inputs come from a black and white image 10 x 10 pixels in size. That would mean we have 100 input values. For each pixel that is white[13], we'll say the input on its corresponding dendrite has a value of -1. Conversely, for each black pixel, we have a value of 1. Let's say our goal is to get this neuron to fire when it sees a letter "A" in the image, but not when it sees any other pattern. Figure 3 below illustrates this:

"Firing" occurs when the output is above some threshold. We could choose 0 as the threshold, but in my program, I found 0.5 is a good threshold. Remember; our only output (axon) for this neuron is always going to have a value between -1 and 1. The key to getting our neuron to recognize an "A" in the picture is to set the weights on each dendrite so that each of the input-times-weight values will be positive. So if one dendrite (for a pixel) is expecting to match white (-1), its weight should be -1. Then, when it sees white, it will contribute -1 * -1 = 1 to the sum of all inputs. Conversely, when a dendrite is expecting to match black (1), its weight should be 1. Then, its contribution when it sees black will be 1*1 = 1. So if the input image is exactly like the archetype our single neuron has of the letter "A" built into its dendrite weights, the sum will be exactly 100, the highest possible sum for 100 inputs. Using our "sigma" function, this reduces to an output of 1, the highest possible output value, and a sure indication that the image is of an "A".

Now let's say one of the input pixels was inverted from black[14] to white or vice-versa. That pixel's dendrite would contribute a -1 to the sum, yielding 98 instead of 100. The resulting output would be just a little under 1. In fact, each additional "wrong" pixel color will reduce the sum and hence the output.

If 50% of the pixels were "wrong" - not matching what the dendrites say they are expecting as input - the sum would be exactly 0 and hence the output would be 0. Conversely, if the entire image of the "A" were inverted, 100% of them would be wrong, the sum would be -100, and the output would be -1. Since 0 and -1 are obviously below our threshold of 0.5, we would say this neuron does not "fire" in these cases.

The most important lesson to take away from this simple illustration is that knowledge is encoded in the weights on the dendrites. The second most important lesson is that the output of this sort of thinking is "fuzzy". That is, our neuron compares input patterns and generates variable outputs that are higher the closer the inputs are to its archetypical pattern. So while there are definitive "match" (1) and "non-match" (-1) outputs, there is a whole range in between of somewhat matching.

6.3 Training:

One might argue that a neuron as described above can be quite useful as it is for fuzzy matching of patterns, but a little summation and threshold trickery isn't enough to qualify a program as a neural network. One key concept is that a NN can be "trained".

In our example above, we might assume that some programmer hard-coded the "knowledge" - the weights for each dendrite - into the neuron[15] at design time. What makes NNs interesting is that they can be given examples of patterns and hence learn to recognize them thereafter, without any designer explicitly calculating what the weights should be. Most explanations of NNs speak of learning as a collective function of a network of neurons, but I want to introduce the idea of learning in a single neuron instead. I find it's easier to grasp that way.

Let's take our previous example of a single neuron with 100 dendrites taking input from the pixels of a 10 x 10 black (1) and white (-1) image. Let's say we start with weights of zero for all dendrites to start with. The goal of training will be to get those dendrite weights to morph their way to being able to match the desired pattern.

Knowing what we do about this case, we could just find the difference between the results of each weighted input and the desired outcome and just set the weights in one step. The problem with this, though, will become apparent when we start considering matching or not matching different patterns. That is, we will gradually adjust weights in successive steps instead of all at once because we don't want to wipe out previously acquired knowledge. It's important to stress that each time we morph the dendrite weights of a neuron in order to advance its ability to recognize one pattern, we degrade its acquired knowledge of other patterns it should also recognize.

In the sample program I wrote, each neuron can have its own set of what I call "training cases". Each training case identifies a desired output value given a set of input values. In the case of the character recognition example, I have a set of images[16] of letters from "A" to "F". As with our current example, I have one neuron for each letter I want to be able recognize. I use these images to build training cases such that each letter-recognizing neuron gets one positive-output case for its own pattern and one negative-output case for each of the other letters' patterns. For example, the neuron I want to match the letter "B" in an input image has one test case for the letter

"B" that says the output should be as close as possible to 1 if the neuron sees a "B" in its input. It also has test cases for "A", "C", "D", "E", and "F" that says the output for this same neuron should be as close as possible to -1 if it sees any of these.

Returning to our "A" neuron, let's consider what training means. Each neuron goes through its own individual training cycle based on its set of training cases. We take each training case. For each, we ask the neuron to "think"; to generate an output based on the input values that training case holds. We then find the difference between the desired output for that training case and the actual output resulting from thinking. We use that difference as an adjustment rate for each of the dendrites' weights. We only adjust a small percentage of the distance - 1% in my program - to the desired outcome.

We repeat the training cycle several times. With each cycle, the ability of our "A" matching neuron to properly match an "A" image and to not match a "B", "C", etc. image gets better and better. Because with each step, we only morph the dendrite[17] values a small part of the way towards the "ideal" weights, we guarantee that each training case is factored in. The resulting set of weights will be a compromise that equally balances the interests of all the training cases.

Following is the actual code for a neuron to execute one training cycle:

```
Public Sub AutoTrain()
Dim T As TrainingCase, Td As TrainingCaseDendrite
Dim ErrorTerm As Single
Const LearningRate As Single = 0.01
'Learn from each training case in this one cycle
For Each T In Me.TrainingCases
  'Preset my dendrite's source axons' values to reflect
For Each Td In T.Dendrites
Td.ForDendrite.FromNeuron.AxonValue = Td.Value
      Next
'Given the training inputs, generate an output
      Think()
      'Morph toward the desired output
      ErrorTerm = T.AxonValue - Me.AxonValue
     For Each Td In T.Dendrites
    Td.ForDendrite.Weight += Td.Value * ErrorTerm * LearningRate
        Next
      Next
    End Sub
```

After a few dozen invocations of this routine for this neuron, the ability to match even a sloppily drawn letter is surprisingly good. My sample program shows the matches as checkboxes that suggest definitiveness, but in fact every letter's neuron has its own output from -1 to 1 indicating how closely it believes the input value matches its view of how such a letter should appear.

6.4 The Network:

I hope the explanation above gives a sense of the power of an individual neuron to recognize and even "learn" to recognize patterns and to fire (or not) based on recognizing such known patterns. We get into the heart of neural networks' power when we start to link them together.

I must at this point remind the reader that I have not done much experimentation beyond this point. I've technically created a neural network program that supports stacks of layers of neurons and even clusters of stacks that can all interact, but I haven't really gone beyond the point of having a single set of neurons in one "layer". I have knowledge of these concepts, but again, please take what I say with a grain of salt.

A layer in this context, if it's not obvious by the diagram, is just a set of neurons that all share the same inputs. That is, every neuron[18] in one layer has dendrites that extend to all the axons of neurons in a prior layer. Technically, an NN may have any number of "layers" of neurons. But it seems in practice, there are usually only two or three.

The first layer, referred to generally as the "input" layer, is actually just there to make programming easier. Its neurons don't actually have any dendrites. Instead, the input of some input matrix - say, an image of a letter - gets pumped straight into the axon values for each neuron. The neurons themselves are just placeholders so the next layer can tap into these input values in the same way each subsequent layer does.

The last layer is referred to as the "output" layer. Layers between the input and output layers are generally referred to as "hidden" layers[19]. These names sound mysterious, so let me try to ground the thinking behind this. In my sample program (at least, for the character recognition demo), I have two layers. The first layer is just for dumping input values into the network. The second is the output layer, and its neurons are tasked with identifying characters from an input image. Each neuron is for matching a specific character: "A", "B", "C", and so on.

If two layers is good enough for recognizing a pattern, then you might well ask what the point of adding another layer is. The point, generally, is to add layers of abstraction. Let's take the character recognition example and say that the neurons that recognize characters are now in the hidden layer and we add a new output layer.

Perhaps our goal is to determine when a character recognized represents an even or odd character, in an ordinal sense. That is, for "A", "C", and "E", we want to report that we see an odd character and for "B", "D", and "F", we'll report that we see an even character. To do so, we will add two neurons to the output layer - one for even and one for odd - and provide them with training cases, just like we did for the hidden layer to recognize characters. The training cases for the "even" neuron would indicate that the neuron should fire when the previous layer fires to indicate one of the even characters and should suppress any urge to fire when it indicates one of the odd characters.

It should now be apparent that we could extend this concept further and add any number of layers to add new levels of abstraction and behavior. But I should also point out that there's no reason that an entire layer must be devoted to recognizing one class of entities. In fact, we could have neurons in our hidden layer in the above example recognizing[20] characters and have other neurons in the same layer recognizing frequency bands in a music stream. The next layer could then provide responses appropriate to the combination of both kinds of information. So long as each neuron (outside the input layer) has its own well-crafted set of training cases, they'll all learn to cope well with the goals the designer sets out for such a "stack" of neuron layers.

6.5 Back-Propagation:

When researchers speak of learning in neural networks, they often refer to the concept of "back-propagation". Technically, the concept of giving each neuron its own training case is not back-propagation. In back-propagation, one would give training cases where the desired inputs would be for the input layer and the desired outputs would be for the output layer. All the layers would then go through a self-organizing process to figure out how best to achieve the goals of the training cases.

Thus far, I have not reproduced this feat in my own sample program. I'm not sure I want to go there, either, because this seems to be an area where things get complicated. Researchers refer in this realm to the problem of the network seeking a maximal[22] level of fidelity but perhaps getting stuck during training in local maxima and never reaching their global maxima. I suspect this sort of problem gets magnified with each new hidden layer that's added. The potential richness of features increases, but the difficulty of getting training to work right may also.

6.6 Other Interesting Points:

What I've described here is, again, just one kind of neural network and one conception of how to use them. One other interesting type uses linear inputs and outputs - values from -infinity to infinity - and can be used to learn to approximate[23] linear and perhaps even nonlinear mathematical functions. As such, these can be used, for example, to help control tricky manufacturing processes like chemical vapor deposition (CVD).

One thing that's interesting to me is that, whereas my sample and most other neural networks today use gradual linear progression functions to ease an NN into its proper behavior, some researchers use genetic algorithms to evolve knowledge. The basic idea behind this is to take an NN that is untrained, to duplicate it hundreds of thousands of times. For each copy, one assigns random values to all the weights for all the dendrites. Then each one is tested using the same set of test cases. A smaller subset - maybe 10% - of the NNs that get close to approximating the desired behavior are kept and the others get thrown out. These are used to repopulate the "world" by mating pairs of them. By "mating", I mean that some of the weights of parent A go to an offspring and the rest of the weights for the offspring come from parent B instead.

The result is cross-breed[24] somewhere between A and B. Then the tests begin again and again the poor performers are again thrown out so room is made for a new generation of cross-breeds. Random "mutation" -

changing some of the off springs' weights - occurs to help ensure the system continues to find novel approaches to solving the problem of finding the best combination of weights for the neurons' dendrites.

6.7 Not Like Your Brain:

Ever since the concept of neural nets was invented back in the sixties, there has been a bit of a mystique and aura surrounding them. They are often sold in the popular press as pared down electronic versions of the way the human brain works. They are also envisioned by some as the panacea to the artificial intelligence problem. I suspect that serious researchers in this field know better.

Let me voice my own criticisms of such bold claims. First, one could argue that a sigmoid neuron of the sort I created is not a bad model for how an individual neuron in the human brain functions; I'd be willing to buy that. But I have no doubt that neurons in the brain are not lined up in layers like I illustrated in figure 4 above, with each layer serving a different functional purpose and each neuron only talking to neurons in adjacent layers. Evolution is just not that good of a planner. With each new phase of experimentation[25], evolution makes subtle adjustments that serve some short-term good of a given individual. They are always "band aid" solutions. The result is a very function but hideously complicated tangle of neurons.

Second, neurons in the brain are not all the same. Some have short axons and dendrites that only span perhaps millimeters and thus propagate local information, while others can have axons that run many inches or perhaps even several feet. The neurons in your spinal column are structurally and functionally different from the ones in your brain. The ones in your brain vary by where they are, and they all certainly have differing functions[26]. Most artificial neural nets, by contrast, are composed of just one type of neuron, and they are all in a very homogeneous kind of arrangement.

Third, the ideas of training cases and back-propagation seem intuitive enough as a learning paradigm for NNs, but I've never seen any indication that such mechanisms exist in the human brain. Neurons don't take time off every now and then to undergo training and then get switched back into a performance mode. They seem to learn and perform as they go.

A person should take away from this exercise an understanding that a human brain could not start off "tabula rasa", having no knowledge from the beginning. A mass of billions of neurons with no initial weights would do nothing and have no chance of learning anything. Such an amorphous brain would require a long period of orientation before it could even begin to do even the most basic things like regulate breathing and bowel functions, let alone get a person crawling and interpreting what it sees. I suspect our genes are responsible not only for telling cellular machinery how to specialize a stem cell into a neuron and where certain specialized types of neurons should develop and how far and where their axons should reach out, but perhaps even provide initial biases for synaptic connections and their effects on how a neuron responds to input. New research is showing that the 90% of DNA we previously thought was junk because it doesn't code for the proteins that make up the body may actually be largely responsible for differentiating humans from other species.

Many of them do code RNAs that we are starting to understand are responsible for various activities. Could controlling initial neural wiring be one of them? Of course, there's no denying that our brains do learn in ways that are not directly[27] defined by our genes. Once the development process is set in motion and the first neurons emerge in an embryo, interconnections and reinforcements are surely beginning to take shape based on variables in the environment. Such is the essence of learning.

So I think I can fairly say that, no matter how many layers deep and how many neurons wide one makes a neural network, no such network will ever be able to function like a human brain or even become vaguely sentient. That said, I think it's fair to also say that NNs can play a role in AI research. Perhaps an intelligent being can be borne out of an assemblage of just NNs, but it can't just be one big stack. It would have to be lots of individual stacks- what I call a "cluster" of stacks - interacting. Most of the parts of the cluster would have to be pre-trained so they can perform largely unchanging, autonomous functions. They would also have to be able to function in such a way that learning and acting can happen in parallel, or at least in alternating sequences. More realistically, NNs can be used as just one kind of machine in a heterogeneous platform of technologies to provide practical value. Perhaps one part of an AI might decide it wishes to learn to recognize certain patterns in its vision - perhaps different faces or machine parts - and allocate a new neural net to do the job and control when and what it learns, for example.

It seems the science of neural networks is largely dead and NNs have settled into being an esoteric, if rarely-used, tool in various practical applications like cameras and manufacturing processes.

Artificial neural networks have now been applied to a wide variety of real-world problems in many fields of application. The attractive and flexible characteristics of ANNs, such as their parallel operation, learning by example, associative memory, multi factorial optimization and extensibility, make them well suited to the analysis of biological[28] and medical signals. In this study, we review applications of ANNs to brain signal analysis, for instance, for analysis of the electroencephalogram (EEG).

Artificial neural networks are computational framework inspired by our expanding knowledge of the activity of networks of biological neurons in the brain. ANNs cannot hope to reproduce all the still not well-understood complexities of actual brain networks. Rather, most ANNs are implemented as sets of nonlinear summing elements interconnected by weighted links, forming a highly simplified model of brain connectivity. The basic operation of such artificial neurons is to pass a weighted sum of their inputs through a nonlinear hard-limiting or soft "squashing" function.

To form an ANN, these basic calculating elements (artificial neurons) are most often arranged in interconnected layers. Some neurons, usually those in the layer furthest from the input, are designated as output neurons. The initial weight values of the interconnections are usually assigned randomly. The operation of most ANNs proceeds in two stages. Rules used in the first stage, training (or learning), can be categorized as supervised, unsupervised, or reinforced. During training, the weight values for each interconnection in the network are adjusted either to minimize the error between desired and computed outputs (supervised learning) else to maximize differences (or to minimize similarities) between the output categories (unsupervised or competitive learning). In reinforced learning, an input-output mapping is learned during continued interaction with the environment so as to maximize a scalar index of performance (Haykin, 1999).

The second stage is recall, in which the ANN generates output for the problem the ANN is designed to solve, based on new input data without (or sometimes with) further training signals. Because of their multi factorial character, ANNs have proven suitable for practical use in many medical applications. Since most medical signals of interest are usually not produced by variations in a single variable or factor, many medical problems, particularly those involving decision-making, must involve a multi factorial decision process.

In these cases, changing one variable at a time to find the best solution may never reach the desired objective (Dayhoff and DeLeo, 2001), whereas multi factorial ANN approaches may be more successful. In this chapter, we review recent applications of ANNs to brain signal processing, organized according to the nature of brain signals to be analyzed and the role that ANNs play in the applications.

To date, ANNs have been applied to brain data for the following purposes:

1. Feature extraction, classification, and pattern recognition: ANNs here serve mainly as non-linear classifiers. The inputs are preprocessed so as to form a feature space. ANNs are used to categorize the collected data into distinct classes. In other cases, inputs are not subjected to preprocessing but are given directly to an ANN to extract features of interest from the data.

2. Adaptive filtering and control: ANNs here operate within closed loop systems to process changing inputs, adapting their weights "on the fly" to filter out unwanted parts of the input (adaptive filtering), or mapping their outputs to parameters used in online control (adaptive control).

3. Linear or nonlinear mapping: Here ANNs are used to transform inputs to outputs of a desired form. For example, an ANN might remap its rectangular input data coordinates to circular or more general coordinate systems.

4. Modeling: ANNs can be thought of as function generators that generate an output data series based on a learned function or data model. ANNs with two layers of trainable weights have been proven capable of approximating any nonlinear function.

5. Signal separation and deconvolution: These ANNs separate their input signals into the weighted sum or convolution of a number of underlying sources using assumptions about the nature of the sources or of their interrelationships (e.g., their independence).

6. Texture analysis and image segmentation: Image texture analysis is becoming increasingly important in image segmentation, recognition and understanding. ANNs are being used to learn spatial or spatial-frequency texture features and, accordingly, to categorize images or to separate an image into sub images (image segmentation).

7. Edge detection: In an image, an edge or boundary between two objects can be mapped to a dark band between two lighter areas (objects). By using the properties of intensity discontinuity, ANNs can be trained to “recognize” these dark bands as edges, or can learn to "draw" such edges based on contrast and other information.

7. Roles of ANNs In Brain Signal Process:

ANNs are considered to be good classifiers due to their inherent features such as adaptive learning, robustness, self organization, and generalization capability. ANNs are particularly useful in situations where enough data are available for training and where the simpler classification algorithms fail. The results obtained for the epileptic EEG detection using several types of ANNs.

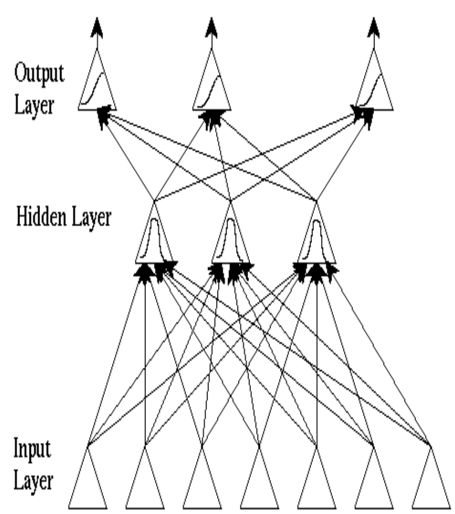

Figure1.3: Three Layers of Neural networks

However, to the best of our knowledge, the performances of the EN for the epileptic EEG detection have not been investigated so far. Two different types of ANNs, namely, EN and PNNs are employed in this present work for the detection of epilepsy. To form an ANN, these basic calculating elements (artificial neurons) are most often arranged in interconnected layers. Some neurons, usually those in the layer furthest from the input, are designated as output neurons. The initial weight values of the interconnections are usually assigned randomly. The second stage is recall, in which the ANN generates output for the problem the ANN is designed to solve, based on new input data without (or sometimes with) further training[29] signals. Because of their multi factorial character, ANNs have proven suitable for practical use in many medical applications.

Below figure shows the brain activity signals consisting of the epilepsy signals. It will be going to training as well as classification.

7.1 Brain Images:

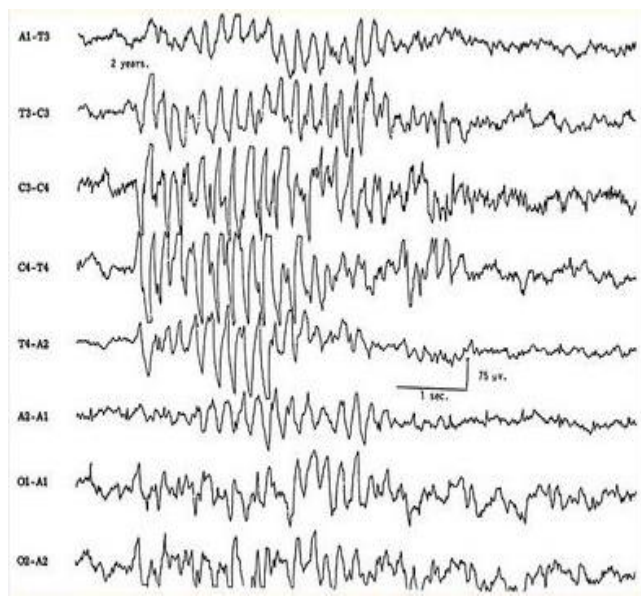

Figure3.4: Epilepsy signal Images

7.2 Structural Images:

In structural brain image analysis, ANNs may play roles in image segmentation, image labeling and/or edge detection. Image segmentation is the first, and probably the most important step in digital image processing. Segmentation may be a labeling problem in which the goal is to assign, to each pixel in a gray-level image, a unique label that represents it's belonging to an anatomical structure. The results of image segmentation can be used for the image understanding and recognition, three-dimensional reconstruction, visualization, and for measurements including brain volume changes in developmental brain diseases such as Alzheimer's disease and autism.

The rapid pace of development of medical imaging devices such as magnetic resonance imaging (MRI) and computerized tomography[30] (CT), allows to better understanding of anatomical brain structure without, prior to, or even during neurosurgery. However, results are highly dependent upon the quality of the image segmentation processes. Here, we give some examples using ANNs in image segmentation: Dunant et al. (1991) presented a back propagation (BP) neural network approach to the automatic characterization of brain tissues from multi-modal MR images.

The ability of a three-layer BP neural network to perform segmentation based on a set of MR images (T1-weighted, T2-weight and proton density weighted) acquired from a patient was studied. The results were compared to those obtained using a Maximum Likelihood Classifier. They showed there was no significant difference in the results obtained by both methods, though BP neural network gave cleaner segmentation images. By using the same analysis strategy, Riddick et al. (1997) first trained a self-organizing map (SOM) on multi-modal MR brain images to efficiently extract and convert the 3-D inputs (from T1-, T2- and PD-weighted images) into a feature space and utilized a BP neural network to separate them into classes of white matter, gray matter, and cerebral spinal fluid (CSF). Their work demonstrated high intra class correlation between the automated segmentation and classification of tissues and standard radiologist identification as well as high intra subject reproducibility.

7.3 Functional Images:

Nowadays, not only does medical imaging device provide impressive spatial resolution and details of the fine structure of the human brain, it is also able to reveal changes in brain status while awake subjects perform a task or even daydream by measuring ongoing metabolic changes including cerebral blood flow (CBF), cerebral blood volume (CBV) (by Positron Emission Tomography, PET), and blood oxygenation level-dependent (BOLD) signal levels (by functional MR imaging, fMRI).

We will give some examples mainly from fMRI analysis. Functional brain imaging emerged in the early 90's based on the observation that increases in local neuronal activity are followed by local changes in oxygen concentration. Changing the amount of oxygen carried by hemoglobin changes the degree to which hemoglobin[9] disturbs a magnetic field. The subsequent changes in the MRI signal became known as the blood-oxygenation-level-dependent or BOLD signal. This technique was soon applied to normal humans during functional brain activation, by cognitive task performance, giving birth to the rapid growing field of functional magnetic resonance imaging. Theoretically, the fMRI BOLD signal from a given brain voxel can be interpreted as a linear combination of different sources with distinguishable time courses and spatial distributions, including use-dependent hemodynamic changes, blood or central spinal fluid flows, plus subject movement and machine artifacts.

Recently, ANNs (especially independent component analysis, ICA), applied to fMRI data, have proven to be a powerful method for detecting and separating task-related activations with either known or unanticipated time courses (McKeown et al., 1998) that could not be detected by standard hypothesis-driven analyses. Duann et al. (2002) have given further details of applying ICA to fMRI BOLD signal showing that the hemodynamic response to even widely spaced stimulus presentations may be trial, site, stimulus and subject dependent. Thus, the standard regression–based method of applying a fixed hemodynamic response model to find stimulus- or task-related BOLD activations needs to be reconsidered.

8. Neural Network Model:

The neural network mathematical model was born in the Artificial Intelligence (AI) research sector, in particular in the 'structural' one: the main idea is to reproduce the intelligence and the capability to learn from examples, simulating the brain neuronal structure on a calculator. The first result was achieved by McCulloch and Pitts in 1943, when the first neural model was born. In 1962 Rosenblatt proposed a new neuron model, called perceptron[8], which could be trained through examples. A perceptron makes the weighted sum of the inputs and, if the sum is greater than a bias value, it sets its output as '1'. The training is the process used to tune the value of the bias and of the parameters which weight the inputs. Some studies underline the perceptron training limits. Next studies, otherwise, show that different basic neuron models, complex neuron networks architecture as suitable learning algorithms, ensure to go beyond the theoretical perceptron limits.

Proposed Algorithm:

From the above discussions we can say that the approximate entropy plays major role to detect the epilepsy. We have proposed a new algorithm to detect the epilepsy 100% accurately without any noise in the detection.

(1) Given a sequence S_N , consisting of N instantaneous data points.

(2) m is input parameter which specifies the pattern length used to compute approximate entropy.

(3) Let r, defines the criterion of similarity and represent the noise filter level.

(4) We denote a subsequence of m beginning at measurement i within S_N, by the vector $C_m(i)$. Two patterns, $C_m(i)$ and $C_m(j)$, are *similar* if the difference between any pair of corresponding measurements in the patterns is less than r. In this process, we defineas follows:

$$C_{im}(r) = \frac{n_{im}(r)}{N - m + 1}$$

Where $n_{im}(r)$ is the number of patterns in P_m that are similar to $P_m(i)$. consider the set P_m of all patterns of length m within S_N.

(5) We can calculate $C_{im}(r)$ for each pattern in S_N, and we define $C_m(r)$ as the mean of these $C_{im}(r)$ values. Finally, we define the approximate entropy of S_N , for patterns of length m and similarity criterion r, as

$$ApEn(S_N, m, r) = \ln\left[\frac{C_m(r)}{C_{m+1}(r)}\right]$$

Summary:

Epilepsy is characterized by the occurrence of recurrent seizures in the EEG signal. This chapter describes the Detection of Epilepsy with traditional Methods, existing system and proposed system. Based on the Elman neural network model, one algorithm is proposed. The code is developed in Visual Basic and .Net. The system employing pic microcontroller and personal computer system. The user interface consists of several screens depicting the graphic plots the behavior of the brain also this explains the interfacing between the sensor, signal processing unit to the microcontroller. Another user interface is used for the patient is effected with epilepsy or to be effected based on entropy value. Epilepsy is characterized by the occurrence of recurrent seizures in the EEG signal. This paper describes the Detection of Epilepsy with traditional Methods, existing system and proposed system. Based on the Elman neural network model, one algorithm is proposed. The system employing pic microcontroller and personal computer system. The user interface consists of several screens depicting the graphic plots the behavior of the brain also this explains the interfacing between the sensor, signal processing unit to the microcontroller. Another user interface is used for the patient is effected with epilepsy or to be effected based on entropy value.

References:

1. N. Mc Grogan (1999). Neural network detection of epileptic seizures in the electroencephalogram,[Online]. Available:http://www.new.ox.ac.uk/˜nmcgroga/work/transfer
2. J. Bruhn, H. Ropcke, and A. Hoeft, "Approximate entropy as an electroencephalographic measure of anesthetic drug effect during desflurane anesthesia," Anesthesiology, vol. 92, pp. 715–726, 2000.
3. Pincus SM, Goldberger AL. Physiological time-series analysis: what does regularity quantify? Am. J. Physiol. 1994 Apr; 266(4 Pt 2):H1643-56.
4. N. Pradhan, P. K. Sadasivan, and G. R. Arunodaya, "Detection of seizure activity in EEG by an artificial neural network: A preliminary study," Comput. Biomed. Res., vol. 29, pp. 303–313, 1996.
5. V. P. Nigam and D. Graupe, "A neural-network-based detection of epilepsy," Neurol. Res., vol. 26, pp. 55–60, 2004.
6. M. K. Kiymik, A. Subasi, and H. R. Ozcalik, "Neural networks with periodogram and autoregressive spectral analysis methods in detection of epileptic seizure," J. Med. Syst., vol. 28, pp. 511–522, 2004.
7. N. Pradhan, P. K. Sadasivan, and G. R. Arunodaya, "Detection of seizure activity in EEG by an artificial neural network: A preliminary study," Comput. Biomed. Res., vol. 29, pp. 303–313, 1996.

8. V. P. Nigam and D. Graupe, "A neural-network-based detection of epilepsy," Neurol. Res., vol. 26, pp. 55–60, 2004.

9. M. K. Kiymik, A. Subasi, and H. R. Ozcalik, "Neural networks with periodogram and autoregressive spectral analysis methods in detection of epileptic seizure," J. Med. Syst., vol. 28, pp. 511–522, 2004.

10. J. Bruhn, H. Ropcke, and A. Hoeft, "Approximate entropy as an electroencephalographic measure of anesthetic drug effect during desflurane anesthesia," Anesthesiology, vol. 92, pp. 715–726, 2000.

11. S. M. Pincus, I. M. Gladstone, and R. A. Ehrenkranz, "A regularity static for medical data analysis," J. Clin. Monit., vol. 7, pp. 335–345, 1991.

12. S. M. Pincus, "Approximate entropy as a measure of system complexity," Proc. Natl. Acad. Sci. USA, vol. 88, pp. 2297–2301, 1991.

13. S. M. Pincus, T. R. Cummins, and G. G. Haddad, "Heart rate control in normal and aborted SIDS infants," Amer. J. Physiol., vol. 264, no. 3, pp. R638–R646, 1993.

14. S. M. Pincus and A. L. Goldberger, "Physiological time series analysis: what does regularity quantify?," Amer. J. Physiol., vol. 266, pp. H1643– Hl656, 1994.

15. J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," Amer. J. Physiol., vol. 278, pp. H2039–H2049, 2000.

16. W. S. Kim, Y. Z. Yoon, J. H. Bae, and K. S. Soh, "Nonlinear characteristics of heart rate time series: Influence of three recumbent positions in patients with mild or severe coronary artery disease," Physiol. Meas., vol. 26, pp. 517–529, 2005.

17. S. M. Pincus, "Older males secrete luteinizing hormone and testosterone more irregularly and joint more asynchronously, than younger males," Proc. Natl. Acad. Sci. U.S.A., vol. 93, pp. 14100–14105, 1996.

18. N. Radhakrishnan and B. N. Gangadhar, "Estimating regularity in epileptic seizure time-series data. A complexity-measure approach," IEEE Eng. Med. Biol., vol. 17, no. 3, pp. 89–94, May./Jun. 1998.

19. X. S. Zhang and R. J. Roy, "Derived fuzzy knowledge model for estimating the depth of anesthesia," IEEE Trans. Biomed. Eng., vol. 48, no. 3, pp. 312–323, Mar. 2001.

20. L. Diambra, J. C. Bastos de Figueiredo, and C. P.Malta, "Epileptic activity recognition in EEG recording," Phys. A: Stat. Mech. Appl., vol. 273, pp. 495–505, 1999.

21. R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state," Phys. Rev. E, Stat. Nonlinear SoftMatter Phys., vol. 64, pp. 061907-1–061907-8, 2001.

22. M. Le Van Quyen, J. Martinerie, V. Navarro, P. Boon, M. D' Have, and C Adam et al., "Anticipation of epileptic seizures from standard EEG recordings," Lancet, vol. 357, pp. 83–188, 2001.

23. H. Demuth andM. Beale, Neural Network Toolbox. (For use with Matlab). Natick, MA: MathWorks, 2000.

24. C. J. Huang and W. C. Liao, "A comparative study of feature selection methods for probabilistic neural networks in cancer classification," in Proc. 15th IEEE Intern. Conf. Tools with Artif. Intell. (ICTAI'03), Sacramento, CA, Nov. 3–5, 2003, pp. 451–458.

25. V. Srinivasan, C. Eswaran, and N. Sriraam, "Artificial neural network based epileptic detection using time-domain and frequency-domain features," J. Med. Syst., vol. 29, pp. 647–659, 2005.

26. Pincus SM, Goldberger AL. Physiological time-series analysis: what does regularity quantify? Am. J. Physiol. 1994 Apr; 266(4 Pt 2):H1643-56.

27. Santamaria J, Chiappa KH. The EEG of drowsiness New York: Demos Publications, 1987:1-202.

28. Hasan, Joel, Past and Future of Computer-Assisted Sleep Analysis and Drowsiness Assessment. Journal of Clinical Neurophysiology. Computerized Polysomnographic Analysis. 13(4):295-313, July 1996.

29. Burioka N, Miyata M, Cornelissen G, Approximate entropy in the electroencephalogram during wake and sleep. Clin EEG Neurosci. 2005 Jan; 36(1):21-4.

30. J. Gotman and L. Wang, "State-dependent spike detection: Concepts and preliminary results," Electroencephalogr. Clin. Neurophysiol., vol. 79, pp. 11–19, 1991.