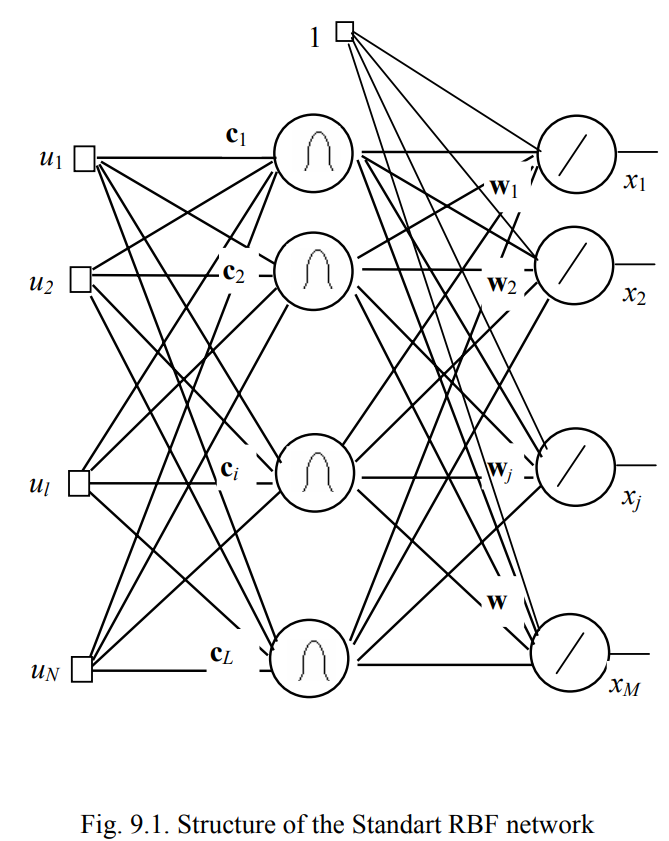
**ABSTRACT & INTRODUCTION**

The topic we will introduce is Radial Basis Function Neural Network(RBFNN) and General Regression Neural Network (GRNN). Those are kind of Artificial Neural Network which is a kind of machine learning algorithm. RBF is found a kind of artificial neural network in late 80’s. GRNN was suggested in early 90’s. Although the two methods are similar, they differ from each other due to some differences. There are many studies and works done by many researchers. Firstly there are informations about RBFNN then GRNN after that we showed the similarities and differences at last we made end up with a conclusion.

**RBFNN**

Radial Basis Function is a feedforward supervised neural network algorithm.It has a basis activation function between input and output layer and it is similar to backpropagation neural network architecture. Unlike backpropagation algorithm it uses Gaussion and some basis activation function instead of S-shaped and sigmoid activation functions.

Each input vector behaves like a local optimized processor that calculates the match between the link weights and centers parameters of that vector. Therefore, basis units are also called highly specialized pattern descriptors.



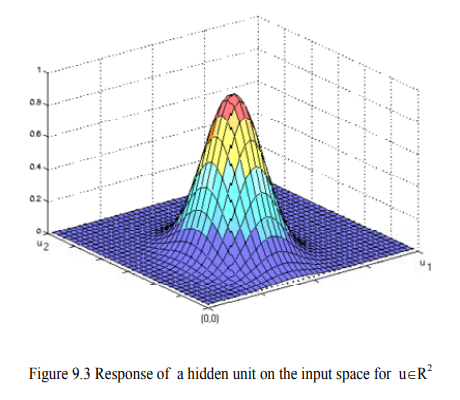
**Radial Basis Function Multivariable Interpolation**

Input layer is the source node which size equals to size of input vector **u.** Second(hidden) layer is connected to all nodes of input layer, and it is large that it can serve as more than a multi-dimensional perceptron. Each hidden unit uses components of input layer as input of it. Each these units use basis functions which has parameters center and width.

-> Euclidean distance

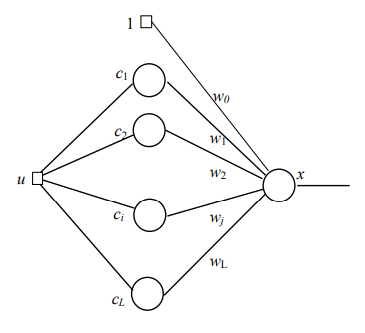
-> Output of each hidden unit

For input vector



Mathematical equation of output layer:

One input and one output RBFNN:



**Training of RBF Network**

If Cİ is known then the problem becomes quadratic.

**Setting Width**

Simply, width must be set with respect to dimension of input data

**GRNN**

What is it?

● This is basically a network-based function approach or function prediction algorithm. Predicts the output of an input data.

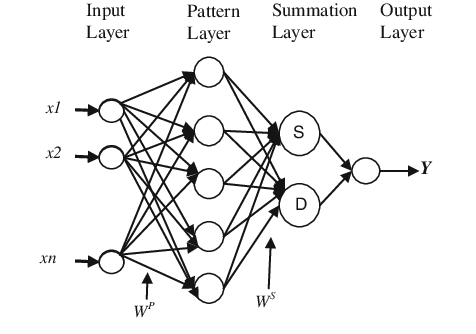
● Unlike feedback artificial neural networks, it can perform all calculations in a single pass and produce results quickly.

Procedure

●For the principle of basif of Neural Network, GRNN needs training data for training itself. Training data must includes input-output matches. If a GRNN model is trained with a training data, model will a give an output or a prediction which is based on a new test data.

●To predict output of GRNN is used output of weighted average of training data. The weight is calculated by using euclidean distance between training data and test data.If weight or distance is large then weight will be small and small distance gives more weight to output.

Architecture of Network



GRNN is a feedforward artificial neural network which is generated by four layer those are input layer, pattern layer, summation layer and output layer. Unlike backpropagation ANN, GRNN does not require iterative training process. Each layer in structure of GRNN has different neuron numbers and layers are sequantially connected to next layer.

1-)**Input Layer:**

**●** In this layer number of neurons depend on features of data in other word number of neurons are equal to dimension of data.

● Transfers the input to the next layer.

2-)**Pattern Layer:**

**●** In pattern layer number of neurons are equal to number of data in training set.

●The distances between training data and test data are calculated in the neurons in this layer and the results are obtained passing through radial basis function to get weight values.

3-)**Summation Layer:**

**●**Summation layer has two sub parts one is Numerator and the other one is Denominator.

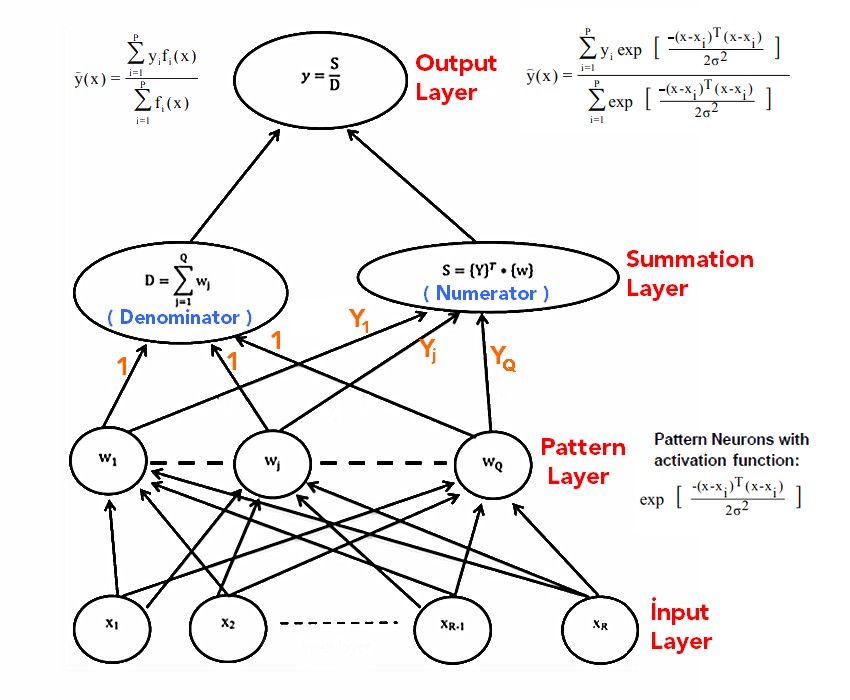
**●**The weights obtained from pattern layer will feed the numerator and denominator layer in summation layer.

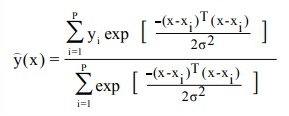
● All the weights are summed in denominator neuron.

● All the weights coming from pattern layer and output of training data in related neuron are multiplied in the numerator neuron, and summation of these multiplications is considered as the output of the numerator neuron.

4-)**Output**:

● In the output layer final output is obtained from value of coming from numerator neuron is divided by value of coming from of denominator.





●The above graph tells the general structure of GRNN and the above equation is the last expression of the output of GRNN model.It is clearly to see smoothing factor specify based on error between output of GRNN neural network and training samples GRNN sinir ağının çıktı verileri ile eğitim örnekleri arasındaki hatanın esas olarak yumuşatma faktörü tarafından belirlendiği görülebilir. That’s why, GRNN has the basit performance control procedure and with setting the smoothing factor it is possible to get better performance.

● is the prediction value of x.

* is activation weight of pattern layer at k.

**Differences of RBFNN and GRNN**

The difference is between GRNN and RBFNN, GRNN has a summation layer and removing weight connections of hidden layer and output layer.

RBF algorithm has shorter training time against to GRNN. But this turns around in clustering algorithms by GRNN.

**Similarities of RBFNN and GRNN**

Both techniques learn in one pass through the data and can generalize from examples as soon as they are stored. The means of the testing period forecasts of GRNN are closer to the observed values compared with the other two ANN methods for long-range prediction.

While RBFNN, GRNN and PNN have simpler architectures and they can train data faster than BPNN, BPNN is a robust model and it can provide competent results in various problems

GRNN is tuned model of RBF which has similar structure.

**CONCLUSION**

We talked about working procedure of both algorithms. We showed the similarites and differences. Clustering with help of c-means clustering, time varying systems are the sectors GRNN is used.

Chaotic time-series modeling, control engineering, electronic device parameter modeling channel equalization, speech recognition, image restoration shape-from-shading, 3-D object modeling, data fusion those are sectors where RBFNN is used.

Thank you for reading.

**REFERENCES**

* <http://users.metu.edu.tr/halici/courses/543LectureNotes/lecturenotes-pdf/ch9.pdf>
* <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5340935&casa_token=zHS0N6F4sBMAAAAA:u9fqmUcMdfjRgXJLmVYUHwbZgUJB54ONnTv5zcJsYz2Pc8IxQQ0ZxqRr16aIzX8kgmk9EsT3hbab>
* <https://www.tandfonline.com/doi/pdf/10.1080/10286600600888565?casa_token=X8dHVpWt15MAAAAA:e1mepvcrnPmVT1WV7JCWQpq_dc-Ew2lp8fpOVGQZlJCX5JOORqHAk2V1mBr183lJVZbg4gfWSInckHg>
* <https://minds.wisconsin.edu/bitstream/handle/1793/7779/ch2.pdf?sequence%3D14>
* <https://github.com/muhendis/Generalized-regression-neural-networks-library-from-scratch>
* <https://easyneuralnetwork.blogspot.com/2013/07/grnn-generalized-regression-neural.html>
* <https://jag.journalagent.com/pajes/pdfs/PAJES_24_5_857_863.pdf>
* <https://www.researchgate.net/publication/280445892_Introduction_of_the_Radial_Basis_Function_RBF_Networks>
* <http://mccormickml.com/2013/08/15/radial-basis-function-network-rbfn-tutorial/>
* <http://www.eng.auburn.edu/~wilambm/pap/2011/Advantages_Yu_Paszczynski.pdf>
* <https://core.ac.uk/download/pdf/322551455.pdf>
* <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.48.651&rep=rep1&type=pdf>
* <https://d1wqtxts1xzle7.cloudfront.net/56278595/grnn.pdf?1523302618=&response-content-disposition=inline%3B+filename%3DA_General_Regression_Neural_Network.pdf&Expires=1607508305&Signature=KStCD-xLG1rZTUiYq320d0khRHvFbO8mznznFtIDENBYTOiSnSAy2kIwxE103dAgBGPXzozzVkDSS511LABulLl9T0zF1Hy9hqOVVXIUdgs3VIfYswtjSxBWwWsiDRXnEYMkGC5mODwXfuUOlTD8g~py3Vipx6A~n5u7KaIr3tF9hTQH7tr9A6H-DvCHkZLrRwBL5ORc7hcKQiNaNZHIDLh0Cg3hOXvxp~Irz7q-XW0Ijndy4EazhHJzLDT3uP~MRgYkIYt32qHKC9-Fse5EK9qbYGZjSmxf83IYCQI0ux8-oF6eW0S27e-u2prYLGWv7wuQ7fVBYT-VbtuZeGq3rA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA>
* <https://github.com/federhub/pyGRNN>
* <https://www.kaggle.com/residentmario/radial-basis-networks-and-custom-keras-layers>