Using Neural Networks for Octree generation

Prashant K Gupta,
Ph.D Scholar, South Asian University, New Delhi, India
Tel: +919911418900, Email: guptaprashant1986@gmail.com

Abstract

Artificial Neural Network (ANN) is a new data mining technique that is finding applications in a number of areas. ANN is inspired from the biological nervous system. We propose through this paper a new application of ANN which is the octree generation. An octree is a tree data structure in which each internal node has exactly eight children. Octrees are most often used to partition a three dimensional space by recursively subdividing it into eight octants. Octrees are the three-dimensional analog of quadtrees[1],[2].

Key Words-Octrees, artificial neural networks, affine arithmetic

I. INTRODUCTION

The field of data mining has seen fast developments in the recent times. It has opened up new areas of research & development and encompasses a variety of techniques. Neural Networks is one such area that has found a lot of importance in recent times due to its use in data mining[32],[31]. Neural networks have found motivation from the human brain. They have found a lot of application in the field of machine learning. They contain a large collection of learning units called the artificial neurons which can be trained & put to use in the desired applications. One practical area that finds close correspondence to the neural networks is that of affine arithmetic. Affine arithmetic is a technique which is motivated by the standard interval arithmetic. Standard interval arithmetic technique suffers from a lot of shortcomings that are overcome by the affine arithmetic[22]. Here the idea is to represent an ideal quantity x by an affine form

$$\hat{x} = x_0 + x_1 e_1 + x_2 e_2 + \cdots + x_n e_n \ldots\ldots\ldots\ldots\ldots (1)$$

In the above equation (1), the x0 is the central frequency of affine form $\hat{x}$, the coefficients xi are the partial deviations of $\hat{x}$ & ei are the noise symbols[3][21]. The xi are finite floating-point numbers and the ei are symbolic real variables whose values are unknown but assumed to lie in the interval U = [−1, 1]. Each of the ei are an independent component of the total uncertainty of the ideal quantity x and the corresponding coefficient xi gives the magnitude of that component[15],[16]. In the next section we give a brief introduction of the neural networks. Subsequent section gives details of the octrees. Following that section, we give the idea about how to use neural networks for octree generation.

II. NEURAL NETWOKS (NN)

NN are a simplified model of biological neuron system. It is a massively parallel distributed processing system made up of highly interconnected neural computing elements. The mechanism by which NN acquire knowledge is called the training & solving the problem using the knowledge acquired as inference. The actual real application of the NN is the artificial neural network (ANN). Human brain consists of the processing unit called the neurons. There are approximately 1010 neurons in the human brain & approximately 104 connections between two neurons[4],[10]. Form childhood till the death of a person, a human constantly learns by an activity called the gain of experience. Just as human brain remembers certain experiences & does not others depending on the fact whether certain experiences are reinforced or not, similarly the artificial neuron also works[11],[12]. Even then the artificial neuron cannot match the processing capabilities of the human brain. Here we have shown a simplified mathematical model of an artificial neuron.Here there are four basic components: the inputs, the weights, summation unit & the thresholding unit. There are n inputs X1, X2, …….Xn. depending on the importance of the information, certain information are strengthened & certain are dropped[25].

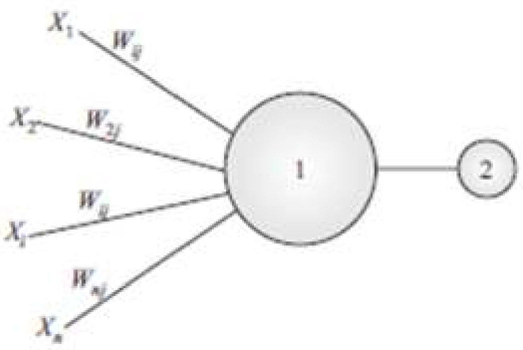

Fig1: Simple model of an artificial neuron Thus, certain inputs are strengthened or retarded. The unit 1 shown is the summation unit & unit 2 is the thresholding unit. The output of the unit 1 is a shown:

$$I= \sum_{i=1}^{n} XiWij \ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

The inputs to the unit 2 is the I& the threshold value. The output of the unit 2 can be expressed as

$$y= \Phi((\sum_{i=1}^{n} XiWij)- \square)\ldots\ldots\ldots\ldots\ldots(3)$$

Here, Φ is the step function:

$$\Phi(I)= \begin{Bmatrix} \mathbf{1}, & \boldsymbol{I} > 0 \\ \mathbf{0}, & \boldsymbol{I} \leq \mathbf{0} \end{Bmatrix}\ldots\ldots\ldots\ldots\ldots(4)$$

Other possible thresholding functions possible are sigmoid, signum function, hyperbolic function, etc. Based on the application & implementation, there are number of ANN architectures such as single layer feedforward network, multilayer feedforward networks, recurrents networks, etc. The ANN have a great capability of processing information in parallel & highly disturbed manner[26],[14].

III. OCTREES

Octrees are used to store data about certain objects or images where the situation demands information storage about object interiors. It finds major usage in medical imaging[5],[6]. The technique used to encode the 3-D data is based on the principle used by quadtree encoding technique for 2-D data. In quadtrees, each dataset to be encoded is divided into four regions. Corresponding to the four regions; data is stored in a particular node as shown in fig2:

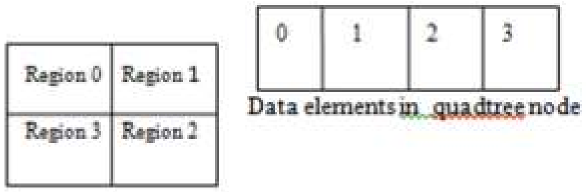

Fig 2: Data representation for a quadtree

The method of storing the information about the image is shown in fig 3.

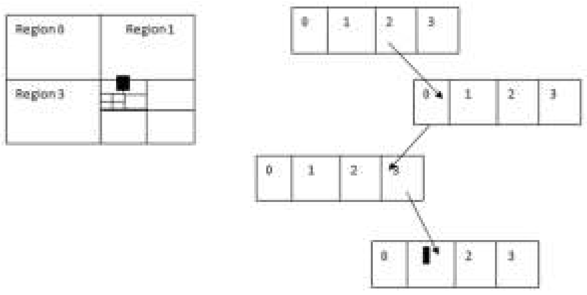

Fig 3: Quadtree representation for data storage

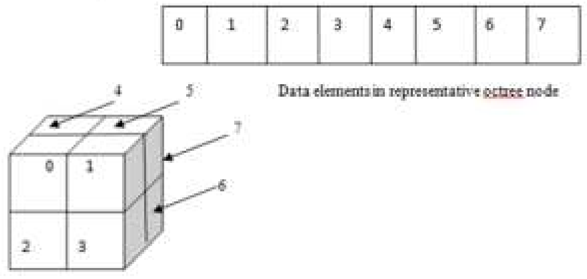

Fig 4: Data representation for an octree

Each individual element of 3-D space is called volume element or voxel. Here also, if the octant is homogeneous, its colour code entry is made into the data element else it is further divided into octants until homogeneous octants are obtained[8][9]. To generate an octree of any object, a 3-D space (generally a parallelepiped) containing the object can be defined around the object. The object is then tested octant by octant to generate octree representation of the object. Once octree representation is obtained for any object, a number of operations can be applied on it such as union, intersection, etc.

IV. USING NEURAL NEWORK FOR OCTREE GENERATION

Octrees are most often used to partition a three dimensional space by recursively subdividing it into eight octants. Octrees are the three-dimensional analog of quadtrees.Octree generation is basically a partitioning scheme where vertices are divided according to their spatial location. The partitioning starts with the application of neural networks. The input to the neural network system is the complete data set[19], [23]. The information can be provided in the form of a collection of the data points. For example, for an image, the information can be provided in the form of a collection of pixel data values. Thus, our concern is that part of the image where information is present. The starting is the error calculation. If this error is unacceptably high, the cell is subdivided and the process repeats for each sub-cell. For the termination criteria, the information comparison for the cell can be done. It can be followed by a singular value decomposition and quantization using neural network[20],[24]. So, the basic idea is that the space is recursively divided into 8 equal portions until some termination criteria stop the process. Thus, using ANN, the termination criteria can be the threshold value of the interval defined such that as long as the information value is more than the threshold, division process continues & when the information falls below the threshold, the division process stops automatically.The impact of this is that the algorithm will be able to discard empty regions of space earlier in recursion or else the result is the dropping of more cells before giving up on the recursion. The result of this is the reduced size of the final octree[27],[29]. Also it must be kept in mind that in many applications, real-time processing is a requirement that cannot be undermined. We provide here an algorithm for the octree generation using ANN:

1. First step is to apply a tightly bounded cube around all the vertices in the previous frame.
2. Second step is to calculate the neural network information representation between all vertices in the bounding cube and the corresponding vertices from the current frame.
3. Third step checks that if the error in the previous step is too large, then partitioning of the bounding cube into eight smaller sub-cubes is done. Then the steps (2) and (3) above are repeated for each of the sub-cubes.
4. If the termination criterion is met, stop the division process.

Once the current frame representation is completed satisfactorily, the algorithm proceeds to the next frame. That is, now the reconstructed current frame becomes the “previous” frame and the next frame becomes the “current” frame and steps are repeated until the last frame in the sequence is encoded. The basic concept is that only the positions of the vertices for the first frame are recorded at the sender side and transmitted to the receiver in the case of 3D video streaming[28],[30]. After the transmission of the first frame, only motion vectors related to each cube of the octree need to be transmitted to the receiving end.

V. CONCLUSION AND FUTURE SCOPE

In the nutshell, ANN can be considered as a good compromise between complexity and efficiency. The network however requires training first then only can be used for drawing inferences. Also, depending on the situation, different architectures can be used.But, the algorithm needs to be improved a lot & also dependencies between data items need to be taken into account.

REFERENCES

[1] Neural Networks by Simon Haykin

[2] Fuzzy Logic and Neural Networks: Basic Concepts & Applications by

Chennakeseva R. Alavala, New Age international publications

[3] Neural Networks, Fuzzy Logic and Genetic Algorithms, Synthesis & Applications, S.Rajasekaran& G.A.V Pai, PHI

[4] Youyou Wang and Guilherme N. DeSouza, "A New 3D Representation and Compression Algorithm for Non-Rigid Moving Objects Using Affine-Octree", Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, October 2009

[5] Jin Sun, Jun Li, Dongsheng Ma, "Chebyshev Affine-Arithmetic-Based Parametric Yield Prediction Under Limited Descriptions of Uncertainty"

[6] Eva Dyllong, "A Reliable Convex-Hull Algorithm for Interval-based Hierarchical Structures

[7] Katja Buhler, "Taylor Models and Affine Arithmetics- Towards a More Sophisticated Use of Reliable Methods in Computer Graphics"

[8] M.Dearing, "Geometry compression," ACM AIGGRAPH'95, 1995.

[9] A.Szymczak, "Optimized edgebreaker encoding for large and regular triangles meshes," IEEE Proceedings of Data Compression, p. 472, 2002.

[10] T. Lewiner, "Efficient edgebreaker for surfaces of arbitrary topology," IEEE Proceedings of Computer Graphics and Image Processing, pp. 218–225, 2004.

[11] J. E. Lengyel, "Compression of time-dependent geometry," ACM Symposium Interactive 3D Graphics, pp. 89–95, 1999.

[12] J.Zhang and C.B.Owen, "Octree-based animated geometry compression," Proceedings of Data Compression Conference(DCC'04), pp. 508–517, 2004.

[13] S. Gupta, K. Sengupta, and A.A.Kassim, "Compression of dynamic 3d geometry data using iterative closest point algorithm," Computer Visionand Image Understanding, vol. 87, pp. 116–130, 2003.

[14] P.-F. Lee, C.-K. Kao, B.-S. Jong, and Y.-W. Lin, "3d animation compression using the affine transformation matrix and pca," IEICE Transacations on Infromation and Systems, pp. 1073–1084, 2007.

[15] J.Zhang and J.Xu, "Optimizing octree motion representation for 3d animation," ACM Session:Graphices and Real-time Systems, pp. 50–55,Mar. 2006.

[16] J.Zhang, J.Xu, and H.Yu, "Octree-based 3d animation compression with motion vector sharing," IEEE International Comference on InformationTechnology, pp. 202–207, 2007.

[17] I.Guskov and A.Khodakovsky, "Wavelet compression of parameterically coherent mesh sequences," Proceedings of the ACM SIGGRAPH/ Eurographics symposium on Computer animation, pp. 136–146,2004.

[18] F.Payan and M.Antonini, "Wavelet-based compression of 3d mesh sequences," Proceedings of IEEE ACIDCA-ICMI, 2005.

[19] K.Muller, A.Smolic, M.Kautzner, and T.Wiegand, "Rate-distrotion optimization in dynamic mesh compression," IEEE International Conference of Image Processing, pp. 533–536, Oct. 2006.

[20] M.Sattler, R.Sarletter, and R.Klein, "Simple and efficient compression of animation sequences," ACM Siggraph Symposium on Computer Animation, pp. 209–217, 2005.

[21] M.Aleca and W.Muller, "Representaing animations by principal components," Computer Graphics Forum, vol. 19, no. 3, pp. 411–418, 2000.

[22] R.Amjoun and W.StraBer, "Efficient compression of 3d dynamic mesh sequences," Journal of the WSCG, no. 1-3, pp. 99–107, 2007.

[23] Q.Gu, J.Peng, and Z.Deng, "Compression of human motion capture data using motion pattern indexing," Computer Graphics Forum, vol. 28(1), pp. 1–12, 2009.

[24] J. Yang, C. Kim, and S. Lee, "Compression of 3d triangle meshe sequences based on vertex-wise motion vector prediction," IEEE Trans.Circuits and Systems for Video Technology, vol. 12, no. 12, pp. 1178–1184, 2002.

[25] L.Ibarria and J.Rossignac, "Dynapack:space-time compression of the 3d animations of triangle meshes with fiexed connectivity," SCA '03:proceedings of the 2003 ACM SIGGRAPH/EurographicsSysmposium on Computer animation, pp. 126–135, 2003.

[26] M. Tournier, L. Reveret, X. Wu, N. Courty, and E. Arnaud, "Motion compression using principal geodesics analysis," in ACM Siggraph/Eurographics Symposium on Computer Animation, SCA (Poster), July 2008. [Online]. Available: http://perception.inrialpes.fr/Publications/2008/TRWCA08

[27] C. Jackins and S.L.Tanimoto, "Oct-tree and their use in representing three dimensinal objects," Computer Grapics and Image Processing, vol. 14, p. 29, 1980.

[28] Y.Huang, J.Peng, and M.Gopi, "Octree-based progressive geometry coding of point clouds," Eurographics Symposium on Point-based Graphics, pp. 103–110, 2006.

[29] R.Schnabel and R.Klein, "Octree-based point cloud compression," Eurographics Symposium on Point-based Graphics, pp. 111–120, 2006.

[30] M. Berz and G. Hoffstätter, Computation and application of Taylor polynomials with interval remainder bounds, Reliable Comput. 4(1) (1998) 83–97.

[31] L.H. de Figueiredo, Surface intersection using affine arithmetic, in: Proc. of Graphics Interface'96, 1996, pp. 168–175.

[32] E. Hansen, A generalized interval arithmetic, in: Interval Mathematics, ed. K. Nickel, Lecture Notes in Computer Science, Vol. 29 (Springer, New York, 1975) pp. 7–18.