Node Localization in Wireless Sensor Networks: Physical Layer based Methods and the Role of AI

Mohd Abbas Zaidi, *Student, EE670* and Shivam Utreja, *Student, EE670*.

***Abstract*—Wireless Sensor Networks are the heart of most IOT systems today. One of the central features of WSNs is to collect data and send it forward. However, the application of transmitted data is highly limited if the location of the nodes is not known. Many applications which involve visual, thermal, acoustic, seismic, or other type of data rely heavily on location information to be attached with the data to ensure that it is significantly assimilated and responded to. Node localization plays a major role in the performance of WSNs. In this work, a summary of localization techniques has been provided with special focus on Physical Layer based Algorithms and the role of Artificial Intelligence in Node localization.**

***Index Terms*—Wireless Sensor Networks, Physical Layer Algorithms, Localization, Neural Networks, Deep Learning.**

I. Introduction

THE process of Localization includes estimating the coordinates of a sensor in absolute or relative sense in a network. Since a WSN has a large number of nodes which may also be mobile depending on the application, it becomes impractical or irrelevant to store the node locations at the installation time.

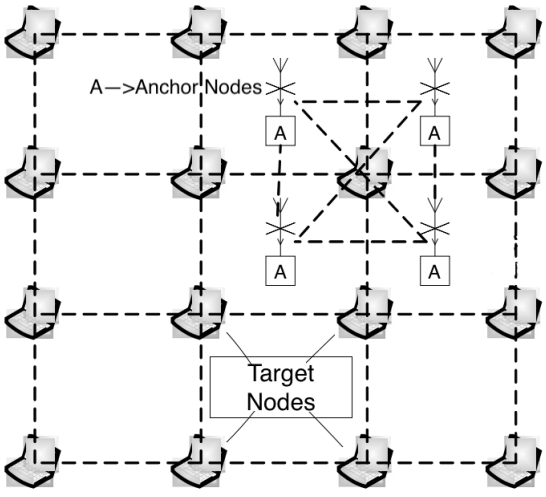

Fig. 1. A typical WSN Network with Anchor and Target Nodes

The research carried out under WSN Node localization has included various range based and range free algorithms. These algorithms include Angle of Arrival, Received Signal Strength Indicator, Time of Arrival, Time Difference of Arrival, and Hop-based Reconstruction. The techniques which find absolute location include the use of GPS, and give the exact latitude and longitude of the node. With inputs from 4 satellites, it is possible to determine the location accurately. The relative techniques estimate the coordinates based on the knowledge of the location of other nodes called anchor nodes. These include the *Time of Arrival* approach where the distance is estimated using the delay in signal propagation. The *RSS* based approach uses the received signal strength to estimate the relative position. We will discuss some techniques based on it later in the paper. We will start with localization techniques at the **Physical Layer**, we will then move on to more recent algorithms based on deep learning and neural network approaches.

A network might have a set of nodes called anchor nodes whose location is known with high accuracy, say using GPS. A number of approaches rely on finding the location of nodes with respect to the anchor nodes.

Authors are with the Department of Electrical Engineering, Indian Institute of Technology, Kanpur, UP, 208016 India

Manuscript received November 16, 2018.

II. Physical Layer Localization Technique

Most of the conventional techniques we discussed so far require special hardware to implement them. This may not be possible in applications where the sensor/ nodes are too small. Therefore, we move on to Physical Layer techniques which try to address this issue.

A. Physical layer network coding (PNC) method

Previous research work on physical network coding (PNC) has lead to improvements in bandwidth usage efficiency when compared to the more traditional techniques. Traditionally, if two nodes depend on a third intermediate node to exchange packets; the exchange will either have to occur on different carrier frequencies or in different time slots (see figure 1(a)).

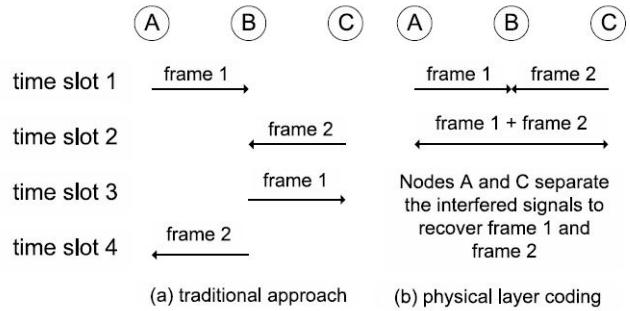

Fig. 2. Traditional vs PNC approach.

In the PNC technique, the intermediate node receives the interference of the signals from the two nodes and then rebroadcasts this to each of the nodes. The two nodes then use the knowledge of their own signal to separate out the signal from the other node from the interference signal (see figure 1(b)).

1) Basic Idea: The PNC approach can be further exploited to localize nodes in a wireless sensor network. The network is assumed to contain anchor nodes that already know their location. The nodes are also assumed to use pairwise keys for secure exchange of information. The technique also requires the nodes to have identical pseudo random bit generators (PRBG), so that the exchange of information regarding the seed of the PBRG is enough to generate identical bit sequences at the nodes.

The technique makes use of the interference property of PNC to derive a locus for a given node. The target node (which needs to be located) computes the difference in arrival times of signals from two anchor nodes(t_{diffA}). This is done by identifying the symbol in the received sequence where the interference begins and combining it with the known value of the speed of radio wave propagation in the medium. This information is also computed for a node whose location is known (t_{diffE}, E being the third anchor node), in order to eliminate the need of synchronized clocks. Finally, this information is combined with the prior knowledge of the locations of the three anchor nodes to derive a hyperbolic locus on which the target node lies. The equation of the locus is given below.

$$d_{AD} - d_{AC} = (d_{ED} - d_{EC}) + s \times (t_{diffA} - t_{diffE})$$

C,D,E are anchor nodes, of which C and D are transmitters. A is the target node. E is the third anchor node in the above description. The velocity of radio wave propagation is s, which is a known constant. d_{XY} denotes the separation between the nodes X and Y. t_{diffX} denotes the difference in arrival of the two sequences at node X from the two anchor nodes.

Finally, one can keep using different pairs of sender nodes (keep one node fixed and change the other sender node) to derive multiple hyperbolic loci for the target node. The target node lies on the intersection of these loci.

2) Practical Issues and their Solutions: Several issues come in the way of making the above idea a physical reality. These issues and their solutions are presented below:

- Collision detection-
  The receiver should be able to distinguish between a state of no input signal (I), one input signal (II), and an input signal comprising of two colliding sequences (III). This is crucial to be able to measure the time difference between the arrivals of the two sequences. Assume we are using the MSK encoding scheme and that signal power is much higher than noise power. Then the state (I) can be differentiated from states (II) and (III) by measuring the average received energy at the receiver. This will be much higher in states(II) and (III) compared to state (I), which is only receiving noise. Now states (II) and (III) can be differentiated by measuring the variance in the incoming energy levels. The energy of the single input sequence will almost be constant and that of the colliding signals will be much more varying.
- Decoding the interfered sequences-
  Up until the point of collision, the receiver needs to run standard MSK decoding. But after the collision, the receiver needs to be able to separate the two incoming sequences and decode each of them. Note that it is crucial for the receiver to be able to verify the authenticity of the incoming sequences from the transmitters and differentiate them from other (potentially random) signals in order to ensure high accuracy of the positioning system. There are two possible, context specific ways to achieve this;
  - *Using pseudo-random sequences:*
    This method requires all the nodes to have the same PRBGs, and the nodes to have pairwise keys. The anchor nodes can then deliver their respective seeds of the PRBGs to the target node (to be located). The target node can then use this information to reconstruct the sequences, hence also being able to reconstruct their interference. This reconstruction is compared to the incoming sequence to verify authenticity.
  - *The target node has no prior information on the type of data being transmitted:*
    In this case, the target node alone cannot separate the interfering sequences. Here we need two receiver nodes with the difference between their t_{diff}s being large enough. Such a pair of nodes can combine their information to separate the interfered sequences for each of them. Following the same node labelling from section II, let the collision at node A begin at the 4th bit, and that at node E begin at the 7th bit. Es knowledge of the 4th bit of sequence from C will help A decode the 1st bit of the sequence from D. This new knowledge of A can now help E decode the 7th bit from C. This exchange between A and E will keep happening up until both completely separate their interfered sequences (see figure 2).

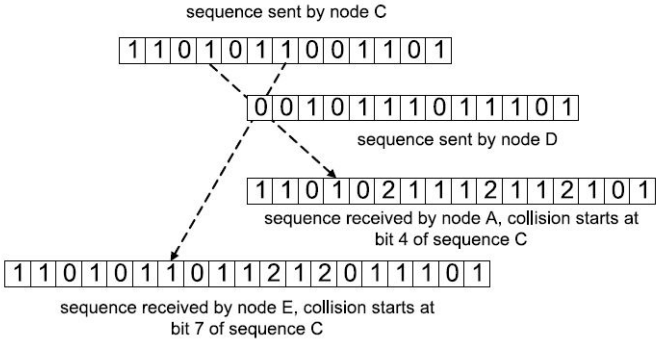

Fig. 3. Illustrating the separation of interfering sequences.

- Anchor distribution-
  Though, at first glance it seems that a high percentage

of nodes need to be anchor nodes (since 3 anchors are needed to locate one target node); this can be circumvented in the scenarios described below;

- *Wireless mesh network scenario:*
  This scenario assumes that the network has several special nodes (like desktops) that have access to high speed internet, know their positions (hence anchors) and are connected to wall sockets. Hence, they can use much higher power levels for transmission (allowing them to act as anchors for nodes within a larger range). We can now use any 3 of these special nodes as anchor nodes for any target node lying in the region at the intersection of the anchor nodes respective ranges.
- *Self-organized ad-hoc network:*
  This scenario does not need any special nodes. Every node has the same transmission power. To localize in this scenario, a small set of anchor nodes are deployed near the central region of the network. These help locate the nodes which are in the immediate vicinity of all the anchor nodes. These nodes, now knowing their location, can act as anchor nodes to help locate nodes lying further out. previous target nodes as anchor nodes to help locate nodes lying further out. This can be tackled by scattering groups of anchor nodes (constituting 3 to 4 anchors in each group) throughout the network and then computing the location from multiple sources and cross-referencing.

3) Performance Analysis:

- Computational overhead-
  Major computation to be done is that of finding the hyperbolic loci and solving them (finding the intersection point). The authors of [1] propose a method that uses simple add and shift operations to find the intersection points. Therefore it is computationally simple and easily implementable.
- Communication overhead-
  The main contributors to the communication overhead are the exchange of received sequence information for separating interfering sequences, and the information (like their own location) that the anchor nodes need to transmit for the derivation of the hyperbolic loci at the target node. As derived in [2], the overhead comes out to be of the order of 9Kbytes, which can be handled by laptops and smart-phones.
- Accuracy of location-
  Error in the detection of the starting point of the collision can propagate into the error of location prediction. This can be corrected by appending a fixed length (say, 64bits)of known bit sequence at the start of each data packet. In this way, any delay in the detection of the collision starting point can be corrected.
  Another factor that could introduce error is frequency jitters. It can affect the calculation of t_{diff} from the sequence length before the collision. It can also increase the BER, which can affect the sequence separation procedure. This can be compensated for by using error control coding techniques.
  Finally, in the scenario of the ad-hoc network, the error in computing the location can accumulate when using

4) Conclusions: This technique makes use of physical layer network coding to locate nodes with the help of anchor nodes (whose position is already known). The major benefit of this technique over the older methods is that it does not require any specialized hardware such as synchronized clocks, directional antennas or RSS indicators. Another desirable property of this method is that it does not require a centralized controller. The communication and computational overheads can be handled at the nodes themselves. Finally, this approach can be implemented in a progressive manner, i.e., the location accuracy can be successively increased (as need be) by increasing the number of hyperbolic loci computed.

III. Deep Learning Techniques

With the advent of better processing and storage units, deep learning has provided significant improvements in various estimation and prediction problems. Deep Learning Algorithms have had tremendous impact in the fields of Mobile and Wireless Networking [3]. Deep Nets are very general in their approach and the burden to learn the unknown patterns or relations between variables is shifted to the processing units. Moreover, they also work decently well with huge datasets. It is expected that they will be more efficient as time progresses, since the processing and storage capacities are sure to improve. Neural Networks are machine learning models

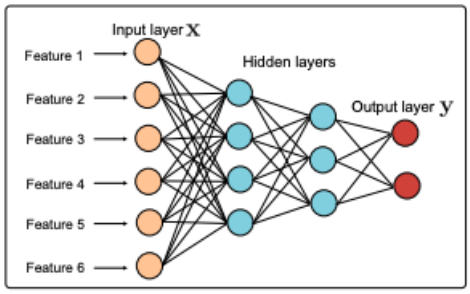

Fig. 4. A traditional Neural Network

that try to mimic human brains in terms of the patterns of learning. Given enough training samples, they can themselves learn the relation between the input variables and the desired output. This is particularly useful since in the case of WSN localization we can exploit the metrics used as inputs for conventional localization techniques and use them to estimate the location of nodes. Therefore NN based approaches can easily be implemented for most of the conventionally used methods given the right set of inputs and calculations.

Neural Networks try to model the topology of the network and hence estimate the location of the sensor nodes. The complexity of the topology is mapped on to complex neural architectures which learn the correct weights and biases corresponding to each connection accordingly. It requires

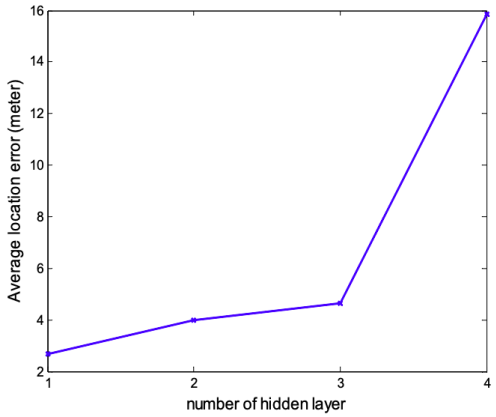

Fig. 5. Variance of number of hidden layers

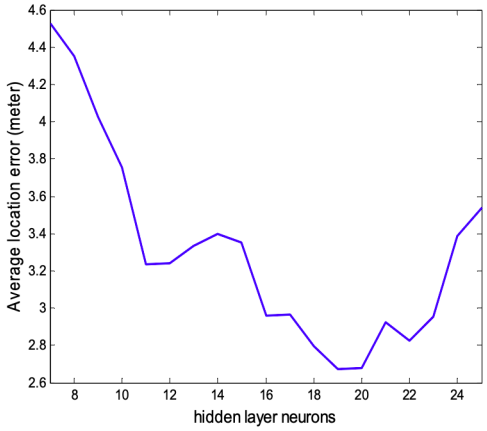

Fig. 6. Variance of hidden layer neurons

an architecture(no of nodes, layers and connections between them), activation functions, error metrics, epochs and stopping criteria). We will discuss a couple of networks based on different mechanisms and then a joint network which tries to merge them and provide superior results.

A. RSSI based Neural Network

Dana et al [4]: There have been conventional methods which try to estimate the distance of node from anchor nodes based on the received power strength. However, these methods already assume a relation between the received power and the distance, which is termed as 'bias' in AI terminology. The conventional model assumes a unique path between sensors and anchor nodes which is a strong assumption. Also, the actual relation depends on the environment and varies from one setting to another. This work aims to estimate the node locations from anchor nodes based on the RSSI(Received Signal Strength Indication) values in a general setting without any bias about the model.

1) Model: The Input to the first layer included the RSSI values from three anchor nodes and the coordinates of the reference nodes. The output layer gives two outputs x,y corresponding to the sensor location. The hidden layer is the layer nodes between the output and input layer. This layer has at least 2 nodes.

2) Training: Various meta-learning tasks were also performed and the model was run for 10 different network architectures and 9 different training algorithms, which basically include different back-propagation or gradient descent techniques. In this sense, this work also provides an analysis of the applicability of various neural networks based methods in the domain of Wireless Sensor Node Localization.

3) Results: Since the setting is very general, and the bias is low we may run into the problem of over-fitting to the noise in the data. This is highly possible since we have a strong bias in terms of the input that we provide to the network. This is evident from the Fig. 1. where we see that increasing the number of hidden layers leads to an increase in the Average Localization Error. Moreover, Fig. 2. shows that the Error reduces as the number of nodes in the hidden layer increases, but increases after a point, which is in line with the expectations.

The final model used had one hidden layer with 19 nodes in it. Levenberg Marqurdt (LM) algorithm was the fastest of the nine algorithms employed.

The network works well for dense networks with nodes being close to anchors. It tends to accumulate errors for sparse network topology. This is an inherent problem with neural networks since they lack distributive learning. As discussed earlier the accuracy grows with the increase in the size of the training set.

Since, the Range based approaches rely on the distance or relative position of different nodes, they need extra hardware and machinery for that task. The large computations are taxing in terms of both cost and time. We next discuss a Range-Free approach which tries to address this issue.

B. Range Free Approach: BP Localization

Runjie et al [5]: This is a bi-layer network to estimate the position of nodes. It employs a centralized localization algorithm based on Range-free approach. The network is based on Hop Counts instead of RSSI values.

The beacons transmit the hop count information; each node stores the minimum of present and former hop counts from the beacons. They transmit the information further by incrementing it. Let us assume a WSN with N nodes, out of which M are beacon nodes. The task includes estimating the location of $N - M$ nodes with respect to the M beacons.

Let s_{im} denote the minimum hop count between the i_{th} node and m_{th} beacon. If the i_{th} node is a beacon itself, then $s_{ii} = 0$(assuming that nodes are ordered giving lower counts to beacon nodes). Let $S_j = \{s_{j1}, s_{j2}, \dots s_{jM}\}$ be the minimum hop-count vector for each node, where $1 \leq j \leq N$.

1) Model: The network consists of two layers, input layer followed by a hidden and output layer. The number of input nodes in the first layer is decided by the number of beacons. The number of hidden nodes is again decided experimentally via meta-learning. Output layer has two nodes, which denote the position coordinates of the output node.

2) Training: The network is initially trained using the beacons. The input is the minimum hop count vector of beacons S_j where $1 \leq j \leq M$. The outputs correspond to the

estimated beacon locations which are denoted by $B_i = (x_i, y_i)$ where $1 \leq i \leq M$. Since the true location of the beacons is known beforehand, the error between the true and estimated location is backpropagated to adjust the weights of the network during the training.

3) Testing: The trained BP network gives us the location of nodes based on the hop counts. During the testing, the input is the minimum hop count vector S_j where $M + 1 \leq j \leq N$. The outputs are the estimated coordinates of the node.

4) Virtual Node based Localization(VNBP): These networks can also be used for other applications. The localization accuracy improves in general with an increase in the number of beacons in the network. However, each beacon taxes in terms of cost, power and other practical constraints. Another method is to add sub-beacons.

Ideally, the nodes whose position has been estimated with high accuracy can be used as sub-beacons. However, there is no way of estimating the accuracy during the test time for non-beacon nodes. Adding virtual nodes to the network can help us select appropriate sub-beacons.

After the network has been trained and tested we have the 'known' and 'estimated' locations of all the nodes in the network. Let us hypothesize some virtual nodes at certain locations in the networks. The network now has P such virtual nodes at location $V_k = (x_k, y_k)$ where $1 \leq k \leq P$. The minimum hop-count vector for the virtual nodes $S_k^v = \{s_{k1}, s_{k2}, \ldots s_{kM}\}$ where $1 \leq k \leq P$, where v denotes that the vector corresponds to a virtual node.

In order to localize the virtual nodes using the trained network, the minimum hop-count vector is required. The distance between the virtual nodes and the beacons is known, this distance has to be converted into hop count. All the virtual nodes within the Wireless range are one-hop nodes. If $D_{AC} \leq R$, then hop-count is one, where R is the Wireless Range Radius of beacon node A. The hop-counts of other virtual nodes need to be computed using their neighbouring nodes(while finding the minimum possible).

After finding S_j^v, the minimum hop-count vector for the virtual nodes, we use the trained network to find the estimated locations of the virtual nodes. We already know the actual positions of these virtual nodes, and hence we also know the accuracy in this case.

The Virtual nodes which have a high accuracy can be used to select the sub-beacons. The assumptions are that firstly, the nodes with similar minimum hop-count vector are nearby, and secondly that the nearby nodes have similar accuracy. Therefore, we find the nodes with similar minimum hop-count vector and use them as sub-beacons.

5) Results: The average value of Localization Error(LE) is used as a metric for the performance of this BP model, which is Euclidean distance normalized by the Wireless Range. It is seen that the introduction of sub-beacons make the localization error of the VN-BP reduce by a considerable value. As seen in the last case, the LE declines as the number of beacon nodes increases. It is noticed that the LE of the BP algorithm goes down more rapidly as the number of nodes increases as compared to other algorithms like DV-HOP and thus outperforms them at higher number of nodes.

C. Effective Neural Network: Joint Approach

The method described by Dana et al is based on offline training and hence requires large amount of training data. It is time-consuming and provides poor performance in terms of localization success rates. The BP/VNBP methods based on minimum HC, but results in large localization errors. To improve the performance a new method was proposed by *Chuange et al* [6]: The model includes online training and correlated topology-trained data. By doing so, the trained model becomes more relevant to the topology. It achieves better localization rates at no additional cost.

The network uses both RSSIs and HCs to estimate the inter-node position. The training data comes from the complete topology, therefore it covers all situations and N_{area} x N_{area} training data is included.

1) Algorithm: Firstly, each anchor node would broadcast its HC as done in the last algorithm. Each anchor node and the unknown node maintains its Hop Count table. The smaller or minimum of the hop-count value is chosen and updated in the table. Therefore, each node records the Hop Count and the RSSI value from the sender. All the anchor nodes and the unknown nodes send their HC tables to the sink.

2) Online Training Phase: In order to realize the training set for the complete topology, we would ideally want to have input table for complete topological coordinates. However, this is not possible. This is very similar to the virtual node hop count estimation problem we encountered in the last case. In this approach, we will use distance, the reciprocals of the physical distance as the input vectors. The distance estimation for unknown nodes uses both HCs and RSSI values. We have N_{area} x N_{area} training data. The training model used is A-16-2, where A stands for the number of anchors, maximum number of epochs(or full data runs) = 50, and MSE threshold for convergence is at 0.001. This means that we will declare the model as trained, only when the mean squared error over the trained data goes below the MSE threshold.

3) Testing: During the test time, we will first need to find the modified input vector for the unknown node, which includes the modification including transforming and modifying HCs to distances. Once the unknown node sends its HC table to the sink, the sink will estimate the distance of the node to each anchor using an estimated average distance of each hop. The sink will then input the reciprocals of these distance to the trained network model. The coordinate of the unknown node is obtained as the output of the trained network model.

4) Results: The log-normalizing shadow model was used to obtain the estimated distances of the RSSI, and the Dijkstras algorithm is used to compute the shortest paths to estimate the distances of HCs. While comparing the performance, the measure of interest is localization rates and localization errors. The experiment was run 10 times with 10 different topologies and node positions. Averaging the results eliminates any bias that creeps in due to initial node placement.

The trends observed with the number of hidden layers, with the number of anchor nodes are same as that in earlier case. The localization error reduces as the number of RSSI samples increase and decreases as the standard deviation of RSSI increases. However, it is seen that the final joint network

model outperforms both the Dana and BP/VNBP models in almost all cases in terms of localization errors.

IV. Conclusion

We presented the Physical Layer algorithms, the main advantage of these techniques is that it can be decentralized, they do not require any external hardware. However, as mentioned earlier it has a high overhead of up to 9 KBytes and needs computational power as much as that of a PDA(Personal Digital Assistants).

On the other hand, Neural Network approaches can simultaneously take advantage of multiple parameters and scale the relative importance to find the location with the best accuracy. An interesting point is that any deep learning approach can also be used to extract sub-beacons if we can construct the input vector of a virtual node. The methodology will be the same as that done for the case of Hop Count as described earlier in the work[5].

Another direction which is relatively unexplored is to design deep-nets very specific to the structure of wireless sensor network which can exactly model the topology and can be used for tasks beyond localization also. This would require a lot of meta-learning while choosing models, activation etc. However, while surveying the advancements due to deep-learning in other fields we felt that this is an area which has great potential.

Acknowledgment

https://www.overleaf.com/project/5bbd7f48e763fa5446ba482d

The authors would like to thank the course Instructor Prof Aditya Jagannatham for this opportunity. We would also thank the course tutors and TAs for support throughout the course. Lastly, we would thank our friends who made the process of learning more meaningful.

References

[1] E. Doukhnitch and M. Salamah, "General approach to simple algorithms for 2-d positioning techniques in cellular networks," *Computer Communications*, vol. 31, no. 10, pp. 2185 – 2194, 2008.

[2] Z. Li, D. Pu, W. Wang, and A. Wyglinski, "Node localization in wireless networks through physical layer network coding," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pp. 1–5, Dec 2010.

[3] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *arXiv preprint arXiv:1803.04311*, 2018.

[4] A. Dana, A. K. Zadeh, and B. Hekmat, "Localization in ad-hoc networks," in *Telecommunications and Malaysia International Conference on Communications, 2007. ICT-MICC 2007. IEEE International Conference on*, pp. 313–317, IEEE, 2007.

[5] R. Liu, K. Sun, and J. Shen, "Bp localization algorithm based on virtual nodes in wireless sensor network," in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*, pp. 1–4, IEEE, 2010.

[6] P.-J. Chuang and Y.-J. Jiang, "Effective neural network-based node localisation scheme for wireless sensor networks," *IET Wireless Sensor Systems*, vol. 4, no. 2, pp. 97–103, 2014.

Mohd Abbas Zaidi Senior Undergraduate student with Major in Electrical Engineering and Minor in Machine Learning at IIT Kanpur

Shivam Utreja Senior Undergraduate student with Major in Electrical Engineering and Minor in Machine Learning at IIT Kanpur