1

Localization in Ad-Hoc Sensor Network: A Machine Learning Based Approach

Vineet Abhishek, CS229 Project Report, Fall 2005

Abstract—We present a new approach to Ad-Hoc sensor network localization from pairwise received signal strength (RSSI). Localization problem is viewed as a classification problem. We map the output of Support Vector Machine (SVM) to probability and present various schemes for estimating nodes' locations from it. We show that application of Laplace Eigenmaps followed by an appropriate Affine Transformation gives a good esitmate of nodes' position.

Index Terms—Support Vector Machine (SVM), Radio Signal Strength Indicator (RSSI)

I. INTRODUCTION

N a wireless sensor network, the data collected from different sensor nodes needs to be correlated together for a meaningful interpretation and application. This often requires relating sensor data to its physical location. Localization form the very first step in the sensor network based applications.

The majority of the localization algorithms use information such as time of arrival of signal, angle of arrival of signal (using directional antennas) or RSSI measurements for localization. Typically, RSSI is not the most accurate indicator for device location as it can be subject to various multi-path and fading effects, which may also be varying randomly over time. The other parameters like the angle of arrival and timing based indicators provide more accurate observations relating to the location of the node. However, they require a dedicated hardware co-located with the sensor node for logging and providing this information. In case of RSSI, the information is already available to the sensor node and no extra hardware is required. This is particularly useful for energy constraint nodes.

The traditional localization approach use *range* estimate as the first step in the localization process. Using signal propagation models, as estimate is formed about the distance between a pair of points. However, accuracy of such scheme is limited by the ranging errors. *Range free* localization from RSSI data is a non standard problem and there have been only few attempts so far in this diretion. In [1], authors described a simple nearest neighbour classification algorithm to obtain coarse localization of objects. A kernel based localization algorithm is proposed in [2].

Motivated by the nature of the problem and some recent work in this area, we explore the use of a learning algorithm for solving this problem. We present a new approach to this problem and discuss and analyze some schems for estimating locations from RSSI data.

This report is organized as following. In section II, the problem statement is introduced. We give a deatiled description of our approach in section III which forms the major part of this report. The performance of various proposed schemes is analyzed in section IV. Conclusions are given in section V.

II. LOCALIZATION IN AD-HOC SENSOR NETWORK

A. Problem Statement

Let $X_1, X_2 \dots X_m, X_{m+1} \dots X_{m+n}$ denote the locations of m+n sensor nodes in R^3 . We assume that position of first m nodes are unknown and represent this set by $X_u = \{X_1, X_2 \dots X_m\}$. Also, we assume that position of last n nodes are known and represent this set by $X_k = \{X_{m+1}, X_{m+2} \dots X_{m+n}\}$. We call the nodes in X_k as Becon nodes. In general n << m. For every pair of nodes (i,j) we are gievn S_{ij} , the signal strength that sensor j receives from node i. Our problem is to recover the positions of all the nodes $i \in X_u$.

B. Brief description of Algorithm

Given any pair of nodes (X_i,X_j) , we train SVM to assign label 1 to it if $||X_i-X_j||^2 < R_o$ or -1 otherwise, where the choice of R_o is up to us. Training is done on the set of becon nodes. We form a feature vector which is robust to signal variations. Once we have this pairwise classification output, we map the SVM output to probability by fitting the data to a logistic regression as described in [4]. Our intution is that a pair of points which are very close will be assigned label 1 with probability close to 1 and points which are very far apart will be assigned label 1 with probability very close to 0. We then use this pairwise probability to estimate the postion of unknown nodes by various schemes as described in the next section.

III. DETAILED DESCRIPTION OF THE ALGORITHM

A. Training SVM and Classification

Consider a pair of nodes (i,j). Our intution is that two nodes are close in space if they receive the similar signal strengths from all the other nodes. Thus, the feature vector for nodes (i,j) is given by $\phi(i,j)$ where it's k^{th} element $\phi(i,j)_k = exp\left(\frac{||Sik-Sjk||^2}{T}\right)$. This choice of feature vector makes system robust to fading and interference, which is commonly observed in wireless signals. So, if there is some obstruction in the line of sight between node i and node k and if node k is close to node k then k will also show the variations similar to that observed in k Note that to reduce the training complexity, we can conside only those nodes k for which at least one of k are k have value greater than

some pre determined threshold, though, we haven't done this in our current implementation.

With this choice of features, we train the SVM to answer if the distance between a pair of nodes is less than R_o . All possible pairs of becon nodes are used for training. We implement this using SMO algorithm with L1 regularization [3]. Once trained, we store the classification output in matrix \mathbf{Y} and value of SVM before thresholding in a matrix \mathbf{S} where $y_{ij}=1$ if SVM output for nodes (i,j) is >0 and $y_{ij}=-1$ otherwise. Similarly, $s_{ij}=w^t\phi(i,j)-b$ denotes SVM output for nodes (i,j) before thresholding.

B. Mapping SVM output to probability

We use the approach given in [4] to map SVM to probability. This is done by first fitting the parameters of a logistic regression function $g(s) = \frac{1}{1 + exp(As + B)}$ and then evaluting $g(s_{ij})$ for each $s_{ij} \in \mathbf{S}$. This gives us a matrix \mathbf{P} where $P_{ij} = g(s_{ij}) = P[||X_i - X_j||^2 < R_o]$.

For fitting the parameters A and B, we form following log-likelihood function and maximize is with respect to to A and B.

$$(A,B) = \arg\max_{A,B} \sum_{f} (t(f)log(g(f)) + (1 - t(f))log(g(f)))$$

where f is SVM output for a pair of becon nodes after it has been trained on the becon nodes and t(f) is the corresponding lable which takes value 0 and 1 instead of -1 and 1.

In order to get an unbiased estimate of the probability, we split the training data into three parts, trained SVM on all combinations of two out of three, and used the output of that SVM on the third part and corresponding labels of third part for fitting logistic regression parameters.

C. Estimating locations

We use the matrix \mathbf{P} obtained earlier for estimating the locations. One possible approach is to write the position of node i as the weighted sum of its neighbours.

$$X_i = \sum_{j \in N(i)} w_{ij} X_{ij} \tag{2}$$

where N(i) denoted the neighbours of node i which can be taken as those nodes j for which entries $C_{i,j}$ is 1. An obvious way method to choose w_{ij} is to take them in proportion to p_{ij} . A sequential algorithm can be formed as follows: We first find the position of nodes which has maximum number of neighbour with known locations (for the very first pass, the node with maximum number of becon nodes in its neighbourhood). We compute the location of this node as the weighted sum of locations of its neighbours with known locations. As further node locations becomes known, they help in locating other nodes. This scheme is very simple but is prone to error propagation. Fig. 2 shows the outcome of such scheme.

An alternative would be to use joint detection which tries to minimize

$$\sum_{i=1}^{m} || \sum_{i=1}^{m+n} w_{i,j} X_j - X_i ||^2$$
 (3)

If P_u, Q_u, R_u and P_k, Q_k, R_k are column vectors denoting the x,y and z co-ordinates of nodes in X_u and X_k respectively, and if we partition weight matrix W as $W = [W_u \ W_k]$ then it can be shown that minimizing (3) gives

$$P_u = (W_u^T W_u - W_u^T - W_u + I)^{-1} (W_k P_k - W_u^T W_k X_k)$$
 (4)

and similar expressions can be obtained for Q_u and R_u . We call this as $MinNorm_2$ scheme.

Though, we are doing a joint detection, when positions of all the nodes are unknown, a trivial solution for (3) is given when all the nodes are at single point. In presence of some becon nodes, nodes won't stick to one point but they will still have the tendency of converging towards the center since number of becon nodes are very small compared to unknown nodes as shown in fig. 3. An alternative would be to minimize first norm instead of second norm, but this improves the results only by a small margin.

A better solution is given by an application of Laplace Eigenmaps [5]. This involves minimizing the cost function

$$S_{LE} = \sum_{i,j} w_{i,j} ||Z_i - Z_j||^2$$
 (5)

subject to constraints

$$\sum_{i} Z_{i} = 0; \ and \ \sum_{i} ||Z_{i}||^{2} = 1; \tag{6}$$

The constraints remove the translation ambiguity and the tendency to put all the points at origin. We take the weights $w_{i,j} = P_{i,j}$. The intution behind using this optimization problem is that nodes for which $P_{i,j}$ is close to 1 are likely to be close and nodes for which $P_{i,j}$ is close to zero are likely to be far apart.

Let W denote the weight matrix. We define $u_i = \sum_j w_{ij}$ and $L = diag[u_1, \dots, u_N] - W$. Let (λ_k, bfv_k) be the eigne value decomposition of L which are arranged in the increasing order by magnitude of eigenvalues. Then the optimal lowest cost 3-dimsensional solution to (5) is given by $Z_i = [\mathbf{v_2}(\mathbf{i}, \dots, \mathbf{v_4}(\mathbf{i})]^T$.

However, the co-ordinate system for Z_i 's and X_i s are different. We map Z_i to X_i by the following way. We assume that in a noise free environment X_i are obtained from Z_i by an affine transformation

$$X_i = AZ_i + b \tag{7}$$

where A is 3×3 matrix and b is a 3×1 matrix. We estimate A and b by posing a least square error problem over becon nodes estimate given by X'_is and Z'_is .

$$(A,b) = \arg\min_{A,b} \sum_{i \in X_b} ||AZ_i + b - X_i||^2$$
 (8)

Once we know A and b, we map back to the original coordinate system by using (7). The results are shown in fig. 4.

A similar approach based on nodes connectivity has been tried in [6] but our approach differ in a sense that we have a definite scheme for choosing weights by using SVM probability values and we propose the use of an Affine transformation to go back to the original co-ordinate system.

IV. TEST RESULTS

We test the validity of the algorithm by simulation. We use the signal model described in [2] for evaluting the performance. Each sensor location x is assumed to receive from a sensor located at x' a signal value following a fading channel model: $s(x,x')=exp(\frac{-||x-x'||}{\sigma_1})+N(0,\sigma_2)$, where $N(0,\sigma_2)$ is the independently generated Gaussina random variable with zero mean and variance σ_2 . This has nothing to do with the choice of the feature vector and a polynomial decay function was found to give similar reuslts.

Simulations are carried out for three different values of noise power $\sigma_2=0.01,0.05,0.1$ while $\sigma_1=1.7$ is kept fixed. The percentage classification errors for these values of noise were found to be 2.82,3.1,3.85 respectively. The parameters for fine localization error is presented in table 1-3 where the values are normalized, so they repesent the error that would have occured in an area of 1×1 . As expected, the performance deteriorates with the increase in the noise power. Moreover, Laplace Eigenmaps perform much better than the other schemes.

Fig. 1 shows the outcome of a typical initial calssification stage. Here the all the points which are within R_o distance from center node are marked withing the large circle. A blue point means a classification label 1 while a red point means a classification label -1. Figs. 2-4 show the outcome of the *sequential*, $MinNorm_2$ and Laplace Eigenmaps schemes respectively for the worst case noise power of 0.1. Blue dots denote the true position of sensor nodes while red dots denote their estimated position. A cross denotes a becon node.

We also tried to test out algorithm on the actual data [7] for Chipcon CC1000. Since our data consisted had only 16 nodes which were partially connected only, we were not able to perform the inital classification with good accuracy. We obtained percentage classification error to be arond 30. But this was mainly due to very small number of nodes and an even smaller fraction of becon nodes on which training was done. We intend to evaluate our alogrithm on a large network of actual sensor nodes in near future.

V. CONCLUSIONS AND FUTURE WORK

We present a new learning algorithm for Ad-Hoc sensor network localization and show that mapping output of SVM to probability followed by application of Lapalce Eigenmaps and Affine transformation gives a good estimate of node locations from RSSI data only. Our algorithm is particularly useful in a dense sensor network.

Our contribution in this project is to show a completely different approach to solving a difficult problem of localization from RSSI data. We do not claim that our algorithm is better than the existing ones but we show that a learning algorithm can be applied for this problem.

Our immediate future goal is to test the algorithm on a large actual sensor network. We would also like to use apply thin plate spline transformation instead of Affine transformation to map back to the original co-ordinate system once we have the outcome of the laplace eigenmaps. Current implementation of SVM has high computational complexity for the training.

 $\label{eq:table in table in$

| | Norm Max Abs Err | Norm Mean Err | Norm Std Dev |
|-------------|------------------|---------------|--------------|
| Sequential | 0.2574 | 0.1229 | 0.0683 |
| $MinNorm_2$ | 0.4046 | 0.1233 | 0.0969 |
| Lap Eigmap | 0.2232 | 0.0666 | 0.0440 |

TABLE II $\label{eq:performance} \mbox{Performance with noise power } \sigma_2 = 0.05$

| | Norm Max Abs Err | Norm Mean Err | Norm Std Dev |
|-------------|------------------|---------------|--------------|
| Sequential | 0.2808 | 0.1265 | 0.0686 |
| $MinNorm_2$ | 0.4379 | 0.1335 | 0.1034 |
| Lap Eigmap | 0.2502 | 0.0713 | .0482 |

Attempts can be made to reduce the complexity of the training phase by a good choice of Kernel which doesn't require computing feature explicitly, or by limiting the size of the feature vector. A study can be made to find better schemes for choosing the weights or to find the optimal weights. A distribute version of the algorithm can also be developed.

VI. ACKNOWLEDGEMENT

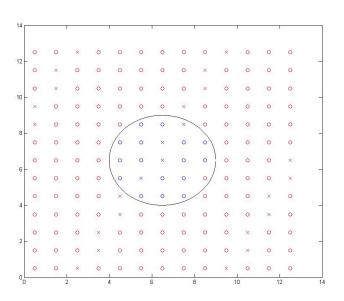
The author would like to thank Professor Andrew Ng and Ashutosh Saxena for insightful discussions. The author is also grateful to Kannan Srinivasan for providing the real RSSI data.

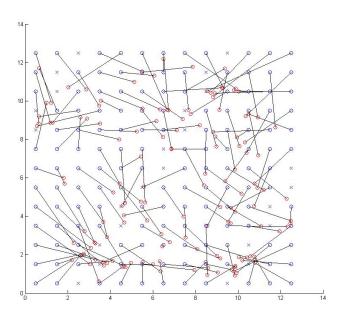
REFERENCES

- Bahl, P. and Padmanabhan, V. N., "RADAR: An in-building RF-based user location and tracking system", INFOCOM (2). 775-784, 2000.
- [2] Xuanlong Nguyen, Miahael I. Jordan, and Bruno Sinopoli, "A kernel-based learning approach to ad hoc sensor network localization", IACM Transactions on Sensor Networks, Volume 1, Issue 1, 134 152, Aug 2005.
- [3] Platt, J., "Fast Training of Support Vector Machines using Sequential Minimal Optimization", Advances in Kernel Methods - Support Vector Learning, 1999.
- [4] Platt, J., "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods", Advances in Large Margin Classifiers, 1999.
- [5] M. Belkin, P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation", Neural Computation, 15 (6):1373-1396, Jun 2003.
- [6] N. Patwari and A. O. Hero, "RAdaptive Neighborhoods for Manifold Learning-based Sensor Localization", in Proceedings of the 2005 Signal Processing and Wireless Communications Conference (SPAWC), New York City, 5-8 June 2005.
- [7] http://cvs.sourceforge.net/viewcvs.py/tinyos/tinyos-1.x/tools/matlab/contrib/kamin/chipconRSSI/

TABLE III $\label{eq:performance} \mbox{Performance with noise power } \sigma_2 = 0.1$

| | Norm Max Abs Err | Norm Mean Err | Norm Std Dev |
|-------------|------------------|---------------|--------------|
| Sequential | 0.3395 | 0.1363 | 0.0730 |
| $MinNorm_2$ | 0.4162 | 0.1259 | 0.0990 |
| Lap Eigmap | 0.2200 | 0.0674 | 0.0450 |

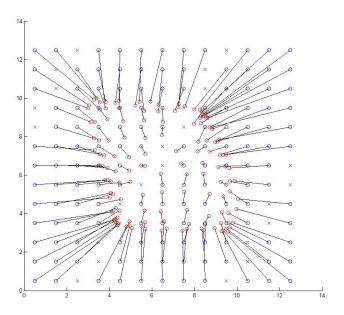




(b) Sequential localization scheme

(a) Coarse Localization: A typical output

Fig. 1. Simulation results



(a) Minimizing second norm $(MinNorm_2)$

Fig. 2. Simulation results

(b) Laplace Eigen value Map