GSM RSSI-based positioning using Extended Kalman Filter for training Artificial Neural Networks

Koteswara Rao Anne, K.Kyamakya, F.Erbas, C.Takenga, J.C.Chedjou
Institute of Communications Engineering (IANT)
University of Hannover
Appel str.9A,Hannover,Germany,D-30167
{ raoanne, kyandogh, erbas, takenga, chedjou }@ant.uni-hannover.de

Abstract

The precise position of the mobile station is critical for the ever increasing number of applications based on location. In this paper, we introduce a novel positioning technique for positioning a GSM mobile phone in real-time. This technique is based on the GSM mobile phone feature, that it can measure the signal strengths from a number of nearby base stations. In this approach we use the GSM signal strengths measured in real environment to train an artificial neural network. The neural network is trained using the second order learning algorithm (Extended Kalman Filter) because of its superiority in the learning speed and mapping accuracy. The mobile position can be determined with good accuracy by providing the current signal strength data to a previously trained neural network. The EKF shows its superiority to the Back Propagation (BP) in both the General Feed Forward (GFF) and the Multi Layer Perception (MLP) neural network architectures. The good accuracy of the calculated position with either an EKF training in a General Feed Forward (GFF) or a Multi Layer Perception (MLP) neural network is shown.

Keywords

Mobile positioning; Neural network training; Extended Kalman filetr; Multi Layer Perception.

I. INTRODUCTION

The phenomenal growth in the ubiquitous computing and location based services fields is making the location-awareness of the mobile device with good precision more important. There are various means of mobile positioning, which can widely be divided into two major categories – network based and handset based positioning methods. In this paper as we are dealing with the positioning of a GSM mobile device. It is worth mentioning that the network based methods result in high signalling load especially in case of continuous tracking. That's why handset based approaches which also have better accuracy than the network based ones are the most recommended. This paper addresses the positioning of a mobile station using radio signal strengths from the serving cell and its neighbouring cells, which are continuously measured by the GSM mobile phone (they are collected in the so-called GSM measurement reports). The block diagram showing the signal flow is shown in Fig.1. The collected data are transferred to a previously trained neural network. In the training phase corresponding GPS positioning data are also provided as target (see Fig.2). In the testing phase there is no need of the GPS signals any more.

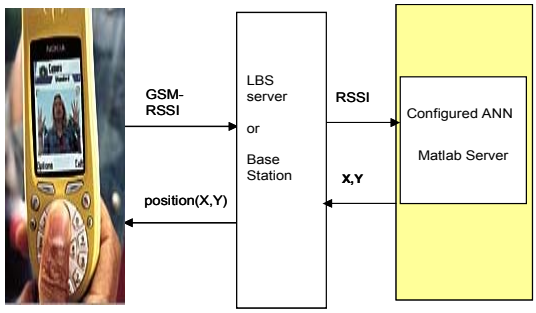

Figure 1. Schematic description of position using signal strengths and ANN.

Even though GPS is a dominant player in outdoor positioning, it has some well known disadvantages. On the other hand, GSM signal strength based positioning requires no additional hardware in the mobile devices and practically every mobile device do measure the signal strengths, even in the idle mode. Many methods have been proposed to calculate the position from the signal strength measurements [1-4]. Even hidden Markov models and pattern recognition models [1-2] have been used. However, the accuracy reached by such approaches is not yet good enough for a good number of interesting location-based services. Some adaptive methods do use fuzzy logic processing of signal strength data [3], however, with still poor accuracy in the range of 0 to 575m. More recent approaches involve Artificial Neural Networks (ANN) for positioning [4, 7]. Some published works use classification networks and adjusted weight initialisation using backpropagation learning rule [4]. They could improve the average accuracy that is however still in the range of 205.3m. In [7], we used a four layered static network with two hidden layers with feed-forward and backpropagation and Levenburg-

Marquardt as the training algorithm and achieved a good accuracy in the range of 50 to 200m. This shows the promise of the ANNs. The key contribution in this paper consists of assessing and optimising the use of the Extended Kalman Filter (EKF) as a training technique for both the Multi Layer Perception (MLP) and General Feed Forward (GFF) ANNs.

In section II, Model description, we describe the ANN models for MLP and GFF. Section III contains the simulation context used for a systematic performance evaluation. The results presented in Section IV do underline the superiority of the EKF.

II. MODEL DESCIPTION

In this section we present the major issues in training a neural network and the ANN models that we used in this work. However, the basic theory of the models in particular and neural networks in general is omitted as extensive literature is available dealing with that. Like in nature (see brains of animals or human beings), the network function is largely determined by the connections between the neurons. ANNs are trained by adjusting the values of the connections (weights) between elements or neurons, in order to get the specific relation between a target outputs and a particular input as shown in fig.1.

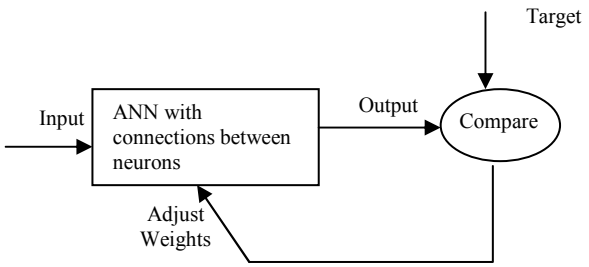

Figure 2. Illustration of the training process of an ANN.

The major factors while using the ANNs are the training method employed to train the network and the network architecture type. In this work we consider two architectures, namely the Feed forward and the static MLP.

A. Artificial Neural network Models

In Fig.2 we see a typical feed forward neural network topology. Data enters the neural network through the input units on the left. The input values are assigned to the input units as the unit activation values. The output values of the units are modulated by the connection weights, either being magnified if the connection weight is positive and greater than 1.0, or being diminished if the connection weight is between 0.0 and 1.0. If the connection weight is negative, the signal is magnified or diminished in the opposite direction.

Mathematically the function of the hidden neuron is described as

$$\sigma\left(\sum_{j=1}^{n} w_j x_j + b_j\right),$$

where the weights $\{w_j, b_j\}$ are symbolized with the arrows feeding into the neuron (see Fig.3).

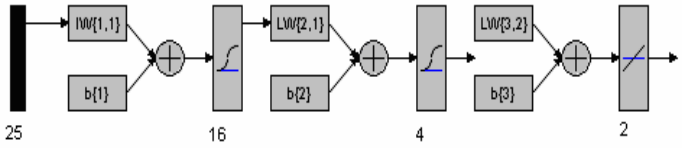

Figure 3. Feed Forward ANN.

The network output is formed by another weighted summation of the outputs of the neurons in the hidden layer. This summation on the output is called the *output layer*. In training the network the parameters are incrementally adjusted till the desired mapping is performed. The GFF network differs from the MLP by not allowing a back propagation and feeding the information only forward. The back propagation feature is however an important one for MLP, [4], where it is taken to improve the performance by knowing the past knowledge. However, it is quite different from the so-called recurrent neural networks where one or more neurons feeds data back into the network, so that they can alter their own input. This is of course used when there is something wrong with the output of the system. Recurrent networks are not considered in this work.

Many standard optimization methods (back-propagation, line search, quasi-Newton) are based on the calculation of gradients. For feed-forward neural networks this is performed in the partial derivatives of the error function with respect to the network outputs (before thresholding); these derivatives form the basis for the back-propagation algorithm. In this work, we used the gradient descent method to compare the results with the EKF. Gradient descent is an incremental hill-climbing algorithm that approaches a minimum or maximum of a function by taking steps proportional to the gradient (or the approximate gradient) at the current point.

B. Extended Kalman filter

The performance of the network (in the gradient descent NNs) is deteriorated by the coupling between the weights in back propagation algorithm. Apart from that appropriate learning parameters should be chosen.

A variety of second-order methods began to be developed using weight updates based on quasi-Newton, Levenburg-Marquardt [7] and conjugate gradient techniques. However, all these methods use the batch-oriented update policy of the weights, an approach that does not allow an incremental and progressive training of the network.
In this paper, we do also use the Extended Kalman Filter (EKF) [5] as the training algorithm to train the ANN. In

contrast to other second order updating methods, EKF updates the weights sequentially and it maintains the error covariance matrix of the training problem. The first use of a global EKF algorithm dates back to 1980 [6]. Our approach is mainly based on the EKF though adapted to the positioning problem of concern. The training of the weights in a neural network can be treated as an estimation problem. The corresponding signal flow is shown in Fig.3.

As shown in Fig.3 the EKF training is carried out in a sequential fashion, with the input training pattern $\mathbf{u}_k$ propagating through the network to produce an output vector $\hat{\mathbf{y}}_k$. The error vector ξ_k is computed by comparing the $\hat{\mathbf{y}}_k$ with the reference output $\mathbf{y}_k$ (expected output). The Kalman gain matrix $\mathbf{K}_k$ is computed using the covariance noise matrix $\mathbf{R}_k$, error covariance matrix $\mathbf{P}_k$, and also the derivative matrix $\mathbf{H}_k$ obtained from back propagation. The weights are updated using the Kalman gain matrix, the error vector, and also the current values of weight vector $\hat{\mathbf{W}}_k$. The forward propagation and back propagation depends on the network type, here in this paper we are using the Multi Layer Perception (MLP) network to take the advantage of back propagation knowing the history of network before training with the new set of parameters.

A successful application of EKF for non linear problems requires the selective choice of the covariance noise matrix $\mathbf{R}_k$, error covariance matrix $\mathbf{P}_k$ at each time step. Unlike the ordinary learning algorithms the EKF can be used as on-line algorithm which facilitates better accuracy as shown in results.

III. SIMULATION SCENARIO

The GSM signal strengths and the position information are measured in set of streets in Hanover, Germany by connecting a GPS and GSM receiver to a PDA. The GPS data obtained in polar coordinates is converted in to Gauß-krüger coordinates. The GSM and GPS data are merged into a single file with respective their time. The data is divided into two parts, one for training the network and other for testing the network. The training and testing data are separated in order to reach a generalization for the network, as it is costly to take measurements on all the streets of a town and train the network for future use.

The various parameters for the neural network are assigned depending upon the performance expected and the architecture chosen as given below:

- Network model (GFF or MLP)
- Number of inputs 12 (6 Cell Identifiers) but can vary
- Number of outputs 2 (X,Y)
- Number of hidden layers 3 ,but can be changed
- Wait update method (batch, online)
- Training function (EKF, trainlm)
- Transfer function in hidden layers(LOGSIG)
- Transfer function at output layer (PURELIN)
- Normalization at channel and normalization at output
- Total number of iterations

The number of neurons in the hidden layer is fixed by the random approximation.

After assigning the different parameters, the network should be trained separately before putting it into test. After training the network, it can be tested with the known data or it can predict the position for a new set of data. The simulation is run for 1500 iterations while training to generalize the network.

In batch updating of weights, the true weight is used to update the parameters of a model. The true weight is usually the sum of the weights caused by each individual training sample. Therefore, batch updating requires one sweep through the training set before any parameters can be changed.

In on-line updating, the true weight is approximated by the weight of the cost function only evaluated on a single training sample. Therefore, the parameters of the model are updated after each training sample. For large data sets, on-line updating can be much faster than batch updating.

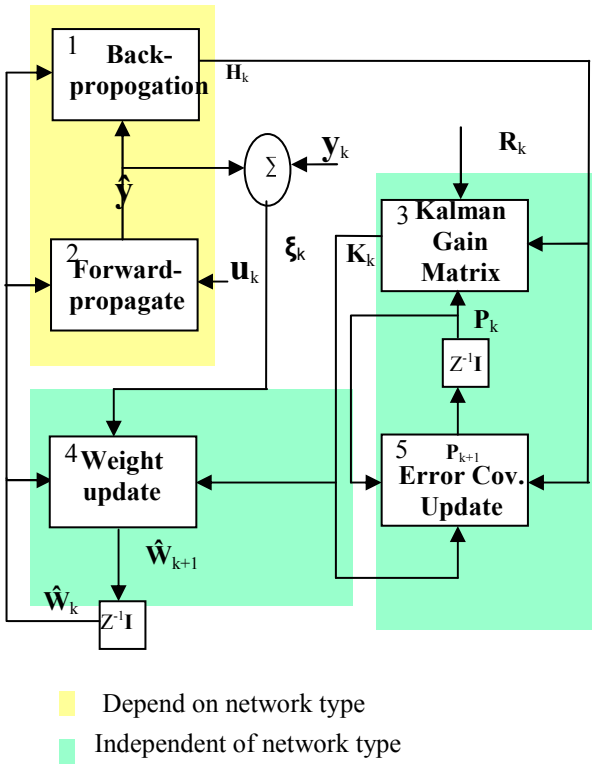

Figure 4. Signal flow diagram for EKF neural network training

IV. RESULTS ANALYSIS

In this section we do analyze the results obtained for a set of streets in Hanover, Germany. Here, the results are presented for cases where the neural networks are tested using data different from those used for training. For obvious reasons the test performance of a neural network is always high if data used for training and for testing are the same. This has been for example the case for part of the experiments reported in [7].

In this work, the emphasis is laid on selecting both the suitable ANN model and the training algorithm. The entire simulation is carried for the GFF network with 1 hidden layer and MLP network with 2 hidden layers. All other parameters are the same for both network types. Besides, the same data are used to test both models in either batch update or online update of the weights. The default training algorithm used is EKF if any other scheme is not specially mentioned.

First we present the active cost involved in the network as shown in Fig.5.The active cost (also called training error) is defined as the minimum error obtained for training the data set. The initial value is set to 1E+009. It is also used to stop the iteration process during the training. The active cost involved in training the network is clearly high for the batch wise updating of the weights, whether it is for GFF or MLP. However, the difference between the MLP and GFF in online updating is very low and it does not need to be taken into consideration as the actual active cost changes in each iteration. From Fig.5 we can say that, while opting for the online updating of the weights the training can be performed quickly irrespective of the network type.

The quadratic error computed in the testing phase is shown in Fig.6 and Fig.7 for GFF and MLP respectively. From Fig.6 the quadratic error is again high for batch wise updating of weights, while compared to the online updating. The batch wise scheme reaches a quadratic error that is almost the double of the one reached by the online updating. For MLP, the online updating is still better, whereby the difference in performance is however much lower. A very interesting result is that the quadratic error for the online updating scheme is only in the range of 48 to 64 m.

The most critical point to be noted while analyzing any neural network is the error or in other words the learning curve of the network changes for each training, even with the same set of data. If the variation in the learning curve is high, then the reliability of the training is very low or in other words the position information obtained can't be trusted. This phenomenon is however significantly reduced in cases of online updating (see Fig.6).

The EKF scheme proved to deliver much better results in the worst scenario cases. This is shown in Fig.8 where the worst performance from the EKF is plotted against the best performance of the Step Gradient Descent training algorithm. The variation in the error for the EKF is very small where as for the gradient descent method it is comparatively high.

V. CONCLUSION

In this paper we have shown that an ANN training using EKF does provide a very well performing positioning system for outdoor (refer to the US FCC requirements for cellular network based positioning). A systematic analysis of the suitable neural network for the positioning, based upon GSM-RSSI, has been identified. Besides, extensive tests and fine tuning experiments have been conducted. Apart from a suitable architecture, a suitable training algorithm and the weight updating method have also been identified.

The performance of the positioning system obtained is very good in terms of accuracy and reliability. Even though the accuracy slightly changes with the training iterations, it is still within a reasonable stable range with variations of not more than only 10 m.

VI. REFERENCES

[1] O. Kinsman, "Pattern Recognition by Hidden Markov Models for Supporting Handover Decisions in the GSM System," in *Proc. 6th Nordic Seminar Dig. Mobile Radio Comm.*, Stockholm, Sweden, 1994, pp.195–202.

[2] O. Kennemann, "Continuous Location of Moving GSM Mobile Stations by Pattern Recognition Techniques," in *Proc. 5th Int. Symp. Personal, Indoor, Mobile, Radio Comm.*, Den Haag, Holland, 1994, pp.630–634.

[3] H.L. Song, "Automatic Vehicle Location in Cellular Communication Systems," *IEEE Transactions on Vehicular Technology*, Vol. 43, pp. 902–908, 1994.

[4] Z. Salcic, E. Chan, "Mobile Station Positioning Using GSM Cellular Phone and Artificial Neural Networks", wireless personal communications 14(3):235-254;sept200

[5] Symon Haykin, *Kalman Filtering and Neural networks*, New York ;John willey &sons, 2001.

[6] S. Singhal and L.Wu, "Training Multilayer Perceptions with the extended Kalman algorithm," in D. S. Touretzkey (Eds.), *Advances in Neural Information Processing Systems 1*, San Mateo, 2001.

[7] K. Kyamakya, I. K. Adusei, F. Erbas, M. Perez, *"A Fingerprint based positioning method using GSM Signal Strength Data and involving a neural network,"* presented at ICWN'03, Las Vegas, USA, 200**3.**

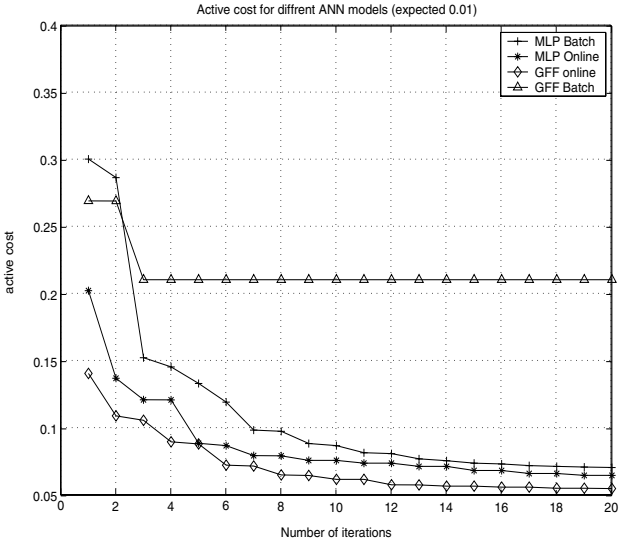

Figure 5. Active cost for training the network models (in online and in batch).

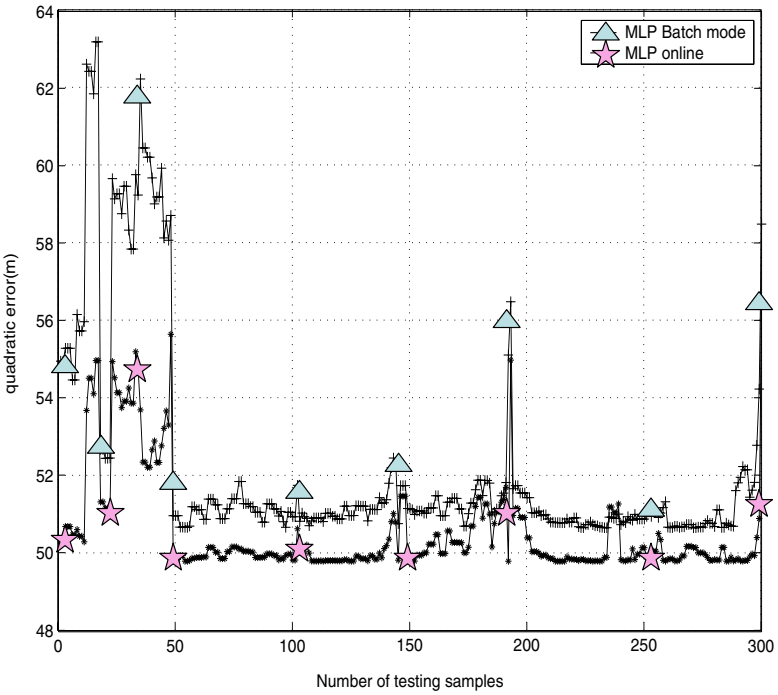

Figure 7. Quadratic error analysis for MLP. Comparison of the performance of Online Training Vs Batch Training.

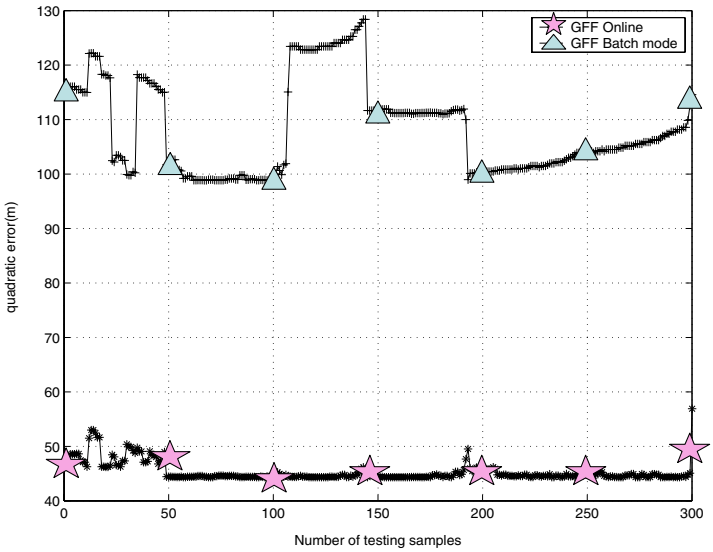

Figure 6. Quadratic error analysis for GFF. Comparison of the performance of Online Training Vs Batch Training.

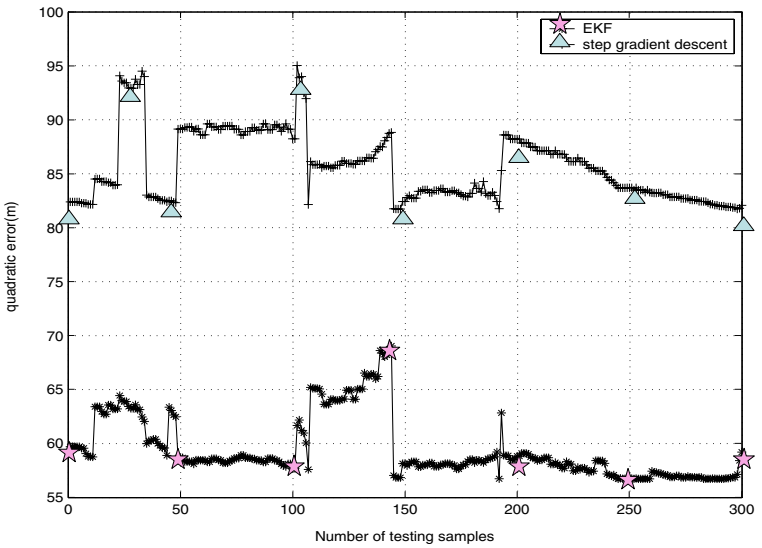

Figure 8. Quadratic error analysis for MLP network while using EKF versus Step Gradient Descent training schemes.