In Proc. of 6th IEEE International Workshop on Signal Processing Advances for Wireless Communications (SPAWC), New York, NY, USA, June 2005

ROBUST REAL-TIME CHANNEL PREDICTION BASED ON INACCURATE INSTANTANEOUS MEASUREMENTS: AN APPROACH

Jirka Klaue and Ana Aguiar

Technical University Berlin
Telecommunication Networks Group
Einsteinufer 25, Berlin, Germany

ABSTRACT

The goal of this work is to show the feasibility of real-time signal strength prediction on the base of instantaneous channel measurements. In real-world systems, like currently available commercial WLAN equipment, the instantaneous signal levels can only be measured with little accuracy. They are affected by a lot of noise from different sources, the measurement hardware is often not calibrated and the sampling interval is coarse and varying. Channel prediction should therefore be able to deal with noisy and non-equidistant data. Furthermore, the algorithms should be relatively simple, since no complex hardware can be afforded for simple WLAN, Bluetooth or sensor systems.

In this paper we introduce a prediction algorithm based on a simple artificial neural network. The algorithm is tested on signal strength values measured with a standard 802.11b WLAN card in various environment conditions. The prediction results are compared with a state-of-the-art channel prediction algorithm. We show that a simple artificial neural network can be used as a robust and comparatively accurate predictor.

1. INTRODUCTION

Channel-adaptive mechanisms like adaptive modulation and coding or adaptive scheduling are very popular means to cope with the short term time varying characteristic of wireless channels (fast fading). These techniques require prediction of future channel behavior to be able to adapt to it. As has been shown in [1] or [2], the more accurate the prediction and the further in the future it lasts, the better for the applications using it. This lead to the development of powerful prediction algorithms for Rayleigh fading channels [3, 4].

Actually, WLAN, Bluetooth or the upcoming sensor networks device drivers usually do not have the possibility to use these algorithms due to many different reasons: accurate measurement hardware is not affordable or necessary, there is not enough computer power for complex calculations, channel samples are not equidistant, the channel characteristics are different from the ones the algorithms are designed for (e. g. not perfect Rayleigh or Rice), more noise is present than expected. However, prediction errors can significantly degrade the performance of channel-adaptive techniques [5], so that finding accurate simple prediction algorithms is an important challenge.

In this paper we evaluate the performance of channel prediction algorithms in non-idealized conditions: the input data to the prediction algorithms result from channel measurements performed with current 802.11b wireless LAN cards. We introduce a prediction algorithm based on a simple multi-layer perceptron (MLP) and compare its performance with that of the modified covariance forecast scheme from [4]. We evaluate the performance in terms of normalized estimation errors.

In the next section, the WLAN channel measurements we used to test the prediction algorithms are briefly described. In Section 3 we explain the necessary preprocessing of the raw measurement data. In Section 4 the performance results — in terms of normalized estimation errors — of the prediction algorithms are presented and compared, and finally we conclude the paper in Section 5.

2. CHANNEL MEASUREMENTS

As mentioned above, we want to evaluate the performance of channel prediction algorithms in real-world conditions. For this purpose we used traces from a wireless channel measurement campaign described in [6] and publicly available at [7]. The measurements were made in low-mobility environments that were thought typical for WLAN users. Figure 1 shows the measured received signal values for example scenarios. The two curves in each graph show the instantaneous signal strength at the base station and at the mobile. Altogether there are measurements in 9 different environments. The average sampling rate of these measurements is 1.3 ms. If we assume a maximum speed of the mobile user (or the environment) of 60 km/h, the channel must be sampled with at least 133 Hz (sampling interval of 7.5 ms). Considering the maximum speed of objects in our

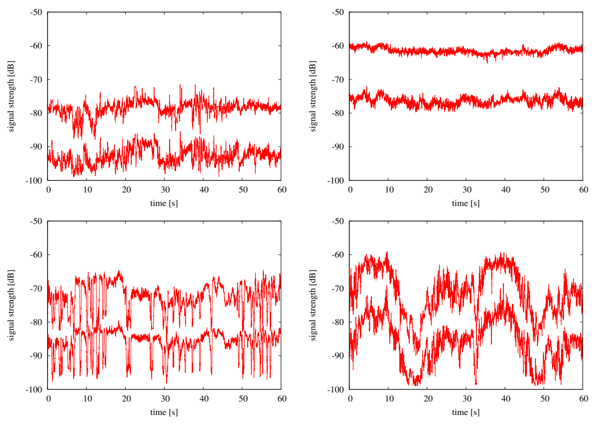


Fig. 1. Example measurements of signal strength (top left to bottom right: scenarios Archi, Mensa, Road and Stadium2)

scenarios we have oversampled the channels. The 1.3 ms are the value of the interval between sent packets. Although sometimes the inter-sample time (which is measured at the receiver) is greater than 10 ms, 80 % of the inter-sample times are below 3 ms as shown in [6]. The measurements also show a high noise level.

3. DATA PREPROCESSING

The measurement data had to be pre-processed in order to provide an useful input for both the modified covariance algorithm and the MLP. These steps are explained in detail in the subsequent paragraphs.

3.1. Interpolation, Normalization and Filtering

First we re-sampled all measurement data to 1 KHz using linear interpolation. This process yields equidistant time series which are required by most statistical evaluation like FFT or autocorrelation. The measured data is available in dB, filter and classical prediction algorithms often work with natural units. For the Modified Covariance forecast algorithm (ModCov) the interpolated measurement data are converted into natural units; for the MLP-algorithm this conversion is not required.

The measured noise should of course be removed from the data before prediction as we are interested only in the underlying fading process. In order to reduce the noise we applied a 30-point moving average filter on the data. Though this filter is not ideal for frequency separation issues, it is fast, simple and has optimal noise reduction capabilities [8]. The frequency response of the filter is expressed with the Dirichlet function:

$$\operatorname{diric}(x) = \begin{cases} -1^{k(n-1)} & x = 2\pi k, k = 0, \pm 1, \pm 2, \dots \\ \dfrac{\sin\left(n\frac{x}{2}\right)}{n\sin\left(\frac{x}{2}\right)} & \text{otherwise} \end{cases}$$

It's solutions for 0 and -3 dB yield the cut-off frequencies as pictured in Figure 2. As can be seen, the 30-point filter also affects frequencies that can belong to the signal we are interested in, and not necessarily be only noise. However, we make this compromise to achieve an adequate noise reduction. For use with the ModCov predictor the data we

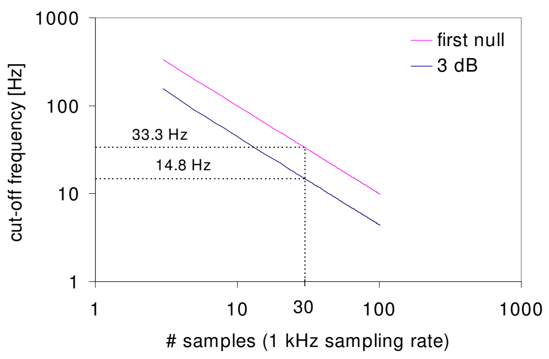


Fig. 2. Cut-off frequency of moving average filter

converted to natural units. The MLP predictor uses the measurement values in dB. The interpolated and filtered data are only scaled to the range [-1, 1], which is the desired input range for MLPs. In this case, -1 corresponds to -100 dB and 1 to -50 dB (the boundaries of the measured data).

3.2. Clustering

MLPs, as defined in [9], are able to learn the patterns of almost any function. In this work, the patterns consist of measured and pre-processed channel data, both “historical” and “future” values. That means, the patterns are created by moving a sliding window over each measurement, extracting the past 10 values as input values and the following 15 values as desired output values. All possible patterns for the measurements are created in this way adding up to 6072752 different patterns.

Learning all these patterns would be a first approach. The main flaw of it would be the impossibility to say something about the performance with similar but new patterns. Over-training is another danger of this method. The usual way to overcome these flaws is to generate training and validation pattern sets from the input. [9] and [10] give deeper insight into these issues.

To create the training set for the MLP, we applied a clustering algorithm for time series on all measurement data sets except the ones from scenario “Bike”. This was done to have some validation patterns which the MLP has never learned.

The clustering algorithm is a variation of the method proposed in [11] for finding similar time series. Two criteria for similarity are defined: the mean absolute deviation $\bar{a}$ and the maximum deviation a_{max}.

$$\begin{array}{rcl} v, w & : & \text{pattern vectors} \\ N & : & \text{vector length} \\ \bar{a} & = & \frac{\sum_{i=1}^{N} |v_i - w_i|}{N} \\ a_{max} & = & \max_{i=1..N} |v_i - w_i| \end{array}$$

Two patterns are declared similar if both $\bar{a}$ and a_{max} are smaller than some threshold. Any pattern that meets the criteria above is put into the cluster where it differs least. For any pattern which does not meet the criteria for an existing cluster, the algorithm creates a new cluster. This first pattern in a new cluster is called the seed of the cluster. Figure 3 shows the result of the clustering algorithm using the thresholds $\bar{a} = 0.01$ and $a_{max} = 0.05$. The size of the bubbles represents the number of patterns in the cluster. The figure clearly shows that the patterns with the smallest standard deviation are most frequent. With these parameters 3316

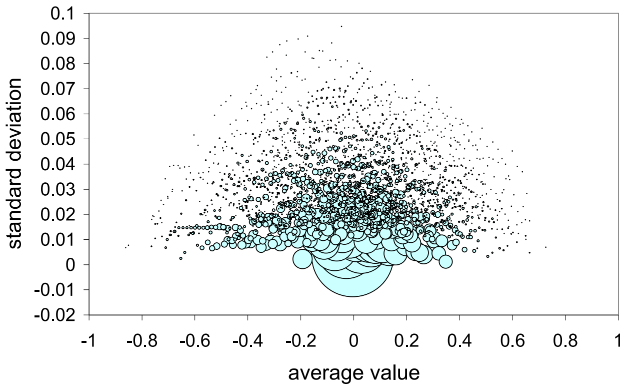
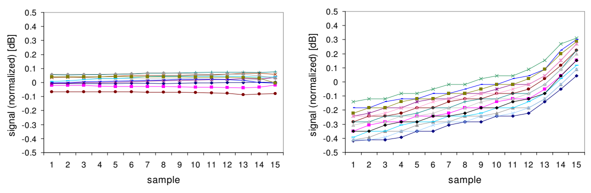


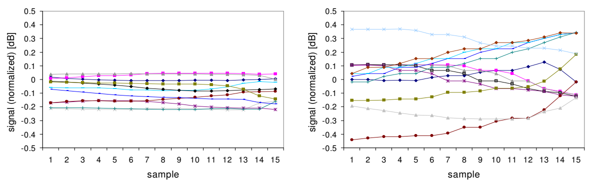
Fig. 3. Number of patterns (bubble size) with certain standard deviation and average value

cluster originated from the 6 million patterns. These values are a compromise between precision (1 cluster per pattern leading to 6 million clusters) and generalization (1 single cluster for all patterns). Optimal values for this trade-off can only be achieved by trying all possible cluster numbers for a wide range of training and validation patterns. That would be very time consuming and would pay-off only in case of real implementation, whereas here we simply do a proof-of-concept.

In order to reduce the complexity of the tests we did not use all 15 future values as desired output of the net. Instead only the values with the index 1, 2, 5, 10, 15 from the current index are used. That means the MLP predicts the signal strength in 1, 2, 5, 10, and 15 ms using the values of the last 10 ms as input. Thus a pattern actually consists of 15 values (10 historical inputs and 5 future outputs covering a range of 15 ms).



(a) No offset correction



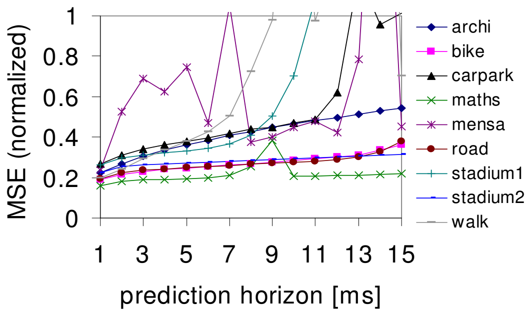
(b) With offset correction

Fig. 4. Patterns with least and highest standard deviation respectively

In Figure 4(a) the seeds of the 10 biggest clusters and the seeds of the 10 smallest clusters are pictured. As the difference between the 10 biggest clusters is only a constant offset, they represent essentially the same function. An offset correction prior to the clustering would significantly reduce the number of clusters while maintaining the differentiation between dissimilar patterns. Figure 4(b) again shows the 10 most frequent and most seldom patterns after offset correction. These figures show that the most frequent patterns are linear (even if different offsets are canceled out), whereas the patterns with random fluctuations are very seldom. This is of relevance because neural networks can approximate well any function, but cannot learn random patterns. Until now we only used the clustering of not offset-corrected patterns.

4. PREDICTION RESULTS

We applied the prediction schemes on all available measured channel data and normalized the prediction errors in order to make the schemes comparable. In the following paragraphs the results from the ModCov scheme and from our MLP approach are presented and compared according to the normalized mean squared error (NMSE) [12]. The prediction errors are normalized with the mean squared value of the channel in use to make the errors in different scenarios comparable. Figure 1 illustrates the necessity of this



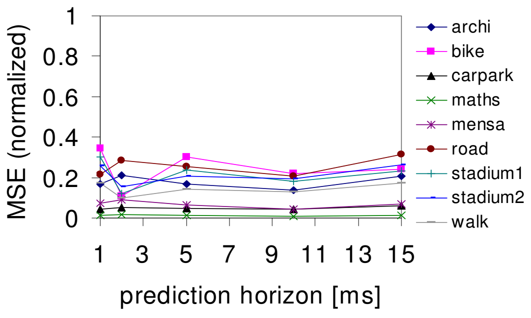
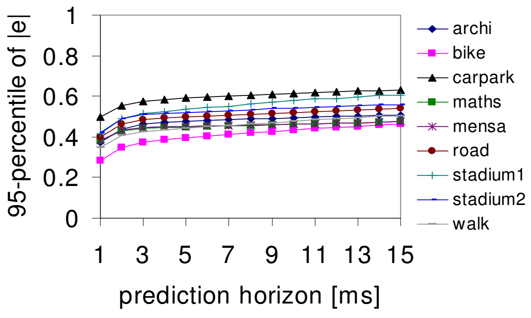


Fig. 5. The MSE of the normalized prediction error for all measured scenarios, left: ModCov, right: HybNet



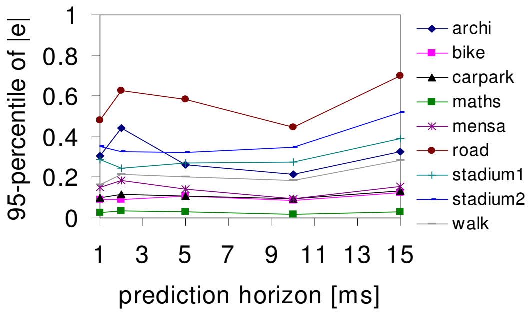


Fig. 6. The 95-percentile of absolute errors for all measured scenarios, left: ModCov, right: HybNet

normalization. From the normalized errors the mean square error (MSE) is calculated for each environment and per prediction horizon.

4.1. ModCov Scheme

The ModCov scheme was applied on all pre-processed (interpolated, converted and filtered) data. For this purpose we used the modified covariance forecast scheme implementation provided in [13]. At any time, the last 30 values were used to predict the next 15 samples. The predicted values were compared to the original (not filtered) measured values, since we cannot say what is noise and what not.

On rare occasions, the ModCov method became unstable, producing enormous errors probably as a result of an overshooting filter, since the algorithm is not designed for non-Rayleigh channels. However, the errors with very big values invalidated the usage of MSE. To avoid this we decided to include a sanity check in ModCov: if the prediction error is greater than the 95-percentile of the last 30 ms of the channel, it is considered to be an outlier. In these cases, the predicted signal strength value is replaced by the last measured (pre-processed) value.

4.2. HybNet Scheme

The core of the proposed prediction method is an artificial neural network with ten inputs, one hidden layer with ten neurons and five outputs. As activation function $\tanh$ was used. The learning process was done using standard backpropagation. The MLP was created and simulated with JavaNNS from the University of Tübingen in Germany [14].

At a first try the network was trained with the seeds of all 3316 clusters from Section 3. Subsequent tests with the actual measured data revealed that the trained net, though yielding good overall results, had problems with some of the measured data. A possible solution could be to increase the number of hidden neurons. But, as we want to keep the complexity as low as possible, we tried another way. Using the fact that most patterns are of almost linear shape (see Section 3.2), we grouped the clusters into two groups: one consisting of the 256 cluster seeds with the lowest standard deviation (these 256 cluster seeds represent over the half of all available 6 million patterns), the other group contains all other clusters. One MLP of the above described shape was trained with the cluster seeds from the group with low standard deviation and one MLP of the same shape was trained with the seeds of the more varying clusters. Subsequently, the standard deviation threshold used to separate the clusters into groups was also used to decide which network should do the prediction.

4.3. Comparison

Figure 5 depicts the MSE for both schemes for all environments. Even though the maximum error was bounded for ModCov, the MSE becomes very big especially for long prediction horizons. HybNet does not have the problem of such great outliers. Moreover, the overall MSE is smaller in almost all situations.

To overcome the problem of the great influence of outliers, we additionally calculated the CDF of the normalized absolute prediction errors. From this CDF the 95-percentile was extracted. It is depicted in Figure 6. Again, the MLP approach outperforms ModCov except for the environment "Road". Although the data from the scenario "Bike" was not used for training the MLPs (see Section 3.2), HybNet performs very well with these measurements.

5. CONCLUSIONS AND FUTURE WORK

We have presented an approach for prediction of wireless channels using noisy historical channel data. The method is based on a simple MLP trained with very few training patterns. These patterns were extracted from channel measurements in various low mobility environments. The number of patterns was reduced via clustering to enable the net to learn the underlying signal patterns rather than noise.

The performance of HybNet was compared with a state-of-the-art forecast scheme and turned out to be more accurate and more robust under the assumed difficult but realistic conditions. The performance of our approach can be improved if we optimize the training set on the one hand and the MLP shape on the other hand. Since the performance of HybNet is a lot better for some environments than for others, it can be expected that a better choice of the rules for clustering the training patterns will improve the performance, as explained in Section 3.2. Furthermore, tests with synthesized Rayleigh channels will be done in order to proof the general applicability of the concept.

6. REFERENCES

[1] Ye (Geoffrey) Li, Leonard J. Cimini, Jr, and Nelson R. Sollenberger, "Robust channel estimation for OFDM systems with rapid dispersive fading channels," *IEEE Transactions on Communications*, vol. 46, pp. 902–915, July 1998.

[2] Geir Egil Øien, Henrik Holm, and Jørgen Hole, "Impact of channel prediction on adaptive coded modulation performance in rayleigh fading," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 3, May 2004.

[3] T. Ekman, M. Sternad, and A. Ahlén, "Unbiased power prediction on broadband channels," in *Proc. of IEEE Vehicular Technology Conference 2002 — VTC 2002 Fall*, Sept. 2002.

[4] S. Semmelrodt and R. Kattenbach, "Performance analysis and comparison of different fading forecast schemes for flat radio channels," COST Technical Report, Jan 2003.

[5] Vincenzo Lottici, Aldo D Andrea, and Umberto Mengali, "Channel estimation for ultra-wideband communications," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 9, Dec 2002.

[6] A. Aguiar and J. Klaue, "Bi-directional WLAN channel measurements in different mobility scenarios," in *Proc. of Vehicular Technology Conference (VTC Spring)*, May 2004.

[7] Telecommunication Networks Group Technical University of Berlin, "Bi-directional wireless channel measurements," http://www.tkn.tu-berlin.de/˜aaguiar/wlan/wlan.html, 2003.

[8] Steven W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing, San Diego, CA, USA, 1997.

[9] Christopher M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[10] Warren S. Sarle, "AI FAQ / Neural Nets," http://www.faqs.org/faqs/ai-faq/neural-nets/, 2002.

[11] Gautam Das, Dimitrios Gunopulos, and Heikki Mannila, "Finding similar time series," in *Principles of Data Mining and Knowledge Discovery*, 1997, pp. 88–100.

[12] Jeng-Kuang Hwang and J. H.Winters, "Sinusoidal modeling and prediction of fast fading processes," in *Proc. of GLOBECOM*, Nov. 1998, vol. 2, pp. 892–897.

[13] Department of RF-Techniques / Communication Systems University of Kassel, "Spectral analysis and linear prediction toolbox," http://www.uni-kassel.de/fb16/hfk/neu/toolbox/, 2004.

[14] WSI Computer Science Department / Computer Architecture University of Tübingen, "Javanns: Java neural network simulator," http://www-ra.informatik.uni-tuebingen.de/software/JavaNNS/.