A mobility-assisted protocol for supervised learning of link quality estimates in wireless networks

Eduardo Feo Flushing, Jawad Nagi and Gianni A. Di Caro
Dalle Molle Institute for Artificial Intelligence (IDSIA)
Lugano, Switzerland
{eduardo,jawad,gianni}@idsia.ch

***Abstract*—In this paper we propose MAPPLE, a novel method to learn *link quality estimates* in wireless networks. The method is a two-step process that combines a online *distributed protocol*, for gathering link quality measurements, with a *supervised learning* approach, for offline data processing and model building. The distributed protocol exploits *channel probing* and *node mobility*, while the offline learning is based on *Support Vector Regression* (SVR). The core idea is to use the online protocol to dynamically reshape a given network to generate a large number of different network configurations from which we sample, through the transmission of probe packets, link quality measures. Each measure is associated to a vector of network features, related to interference, traffic loads, and local topology, that jointly contribute to the definition of the observed link quality. Quality measures and network features are used to train the SVR model for link quality prediction. We validate our approach by extensive simulation tests, showing the good link quality prediction accuracy of the system, as well as its ability to generalize to networks much larger than the ones used to gather the training data.**

I. Introduction

To provide fast and reliable data delivery in multi-hop wireless networks, such as sensor and mobile ad hoc networks, network protocols need to able to identify *good wireless links*. The effective quality of a wireless link, in terms of expected packet losses and bandwidth, depends on the interplay of a number of factors related both to *hardware*, *software*, and *traffic generation* aspects (e.g., antenna characteristics, and PHY and MAC layer protocols), and to the specific configuration of what we term the *local wireless environment* of the two end-point nodes of the link. The local wireless environment is characterized by the following elements having an impact on the effective quality of the link: (i) the *distance* between the two nodes, that determines the path loss of the transmitted signal (that might be as high as the fourth order exponent of the distance [1]); (ii) the number, relative positioning, and traffic characteristics of the *neighbor nodes*, that determine interference, one of the major factors of performance degradation in wireless multi-hop networks [28]; (iii) additional factors such as the *presence of obstacles*, causing reflections, and *weather conditions*, affecting electrical radio transmissions.

The complex interplay among all these factors of different nature make precise link quality estimation a challenging task in wireless networks. The task is made even more challenging in presence of mobility, since the local wireless environment of a node constantly changes. An extensive body of work has addressed the link quality estimation problem using both *analytical models* and *online statistical estimation* strategies. Since link quality estimation is functional to link selection, most of the work has been carried out in the context of *multi-hop routing*, with the aim of identifying routing paths of better quality than those that could result by using conventional metrics, such as minimum hop count or minimum latency (e.g., [8], [11], [35]). However, also other important applications can benefit from accurate link estimation models, such as agent co-ordination [24], sensor placement [21], topology control [34], load-balancing [36] and relay node placement [10].

In this paper we propose a novel method to estimate link quality in terms of packet reception probability, which we refer to as *mobility-assisted proactive probing and learning estimates* (MAPPLE). It is a combination of a distributed protocol for data gathering based on *channel probing*, with a *supervised learning* algorithm for offline data processing and model building. The method assumes the presence of mobile nodes. It consists of two stages, described in the following.

In the first stage (mobility-assisted proactive probing), the objective is to collect link quality measurements through controlled mobility and data generation. During this stage, we assume the network nodes are neither generating application data nor moving according to application requirements. For instance, this condition could be easily imposed at network deployment time, before starting user operations, or at other times when user network operations can be temporarily put on hold. To collect link quality measurements, the nodes recreate expected network traffic conditions by locally broadcasting *probe messages* at different rates. Moreover, by periodically *switching on and off* and by *moving*, the nodes modify the topological layout of the network, creating both at local and global level a number of short-lived different topological structures, that is, a number of different local wireless environments. The objective is to exploit a given network of n nodes to generate snapshots of a relatively large number of networks with $m \leq n$ nodes, artificially creating in this way many different local wireless environments. By measuring the reception rates of the generated probe messages, it is possible to collect online data samples of the quality of the different links (i.e., of the local wireless environments) that can be present in a network with the same characteristics for number of nodes, traffic loads, and communication interfaces.

MAPPLE's second step (learning estimates) consists in the use of a *machine learning* technique to process offline the gathered data and build a prediction model for link quality, which can be then deployed to the nodes and used to derive link quality estimates online. In practice, node data can be collected at a central processing/sink node, processed, and the results can be sent back to the nodes to be used by other applications needing link quality estimates (e.g., routing protocols). The prediction model is built using a supervised learning framework based on *Support Vector Regression* (SVR) [25]. SVR is a specific instantiation of a *Support Vector Machine* (SVM) [26], a kernel-based class of techniques that in recent years has emerged as one of the best approaches for classification and prediction. Compared to artificial neural networks (ANNs), to which they are related to, SVMs are designed to minimize the structural risk by minimizing an upper bound of the generalization error rather than the training error. In this way, they can provide better generalization capabilities and the ability to better handle large multi-dimensional training sets. Therefore, SVMs are particularly suitable for the task we have at hand, where we want to learn a mapping from many different configurations of local wireless environments to link qualities, using the large set of training data obtained during MAPPLE's first step. An important advantage of using the SVR precisely consists in its *generalization* capabilities. The built model, not only provides accurate estimates for the specific links (i.e., local environments) generated during the online procedure, but, to some extent, it also allows to robustly predict the performance of links that were not present in the network either because they were not generated while executing the online procedure or because they could not be generated at all (e.g., if we use a 10-nodes network for training, the maximum number of interfering neighbors a node could happen to have by exploiting node mobility is 9; however, the model has the capability to generalize also to cases in which a number of neighbors higher than 9 is present in the neighborhood). This characteristic allows the link quality prediction model to be effectively used by applications needing to adapt to topology changes, or to plan controlled topology changes, which may take the form of placement of additional nodes or removal and movement of existing ones.

We validate our approach by extensive simulation tests. We use a small set of 20-nodes networks for gathering data, and we show that MAPPLE is able to build a model that can very reliably predict link qualities for networks ranging from 10 to 100 nodes. As a by product of the training procedure, we also learn which features of a local wireless configurations are really needed to build a reliable model. Finally, we show the beneficial effects of using mobility compared to a version of MAPPLE where only node switching on and off is used.

The rest of the paper is organized as follows. In the next section we discuss related work. In Section III we identify the elements of a local wireless environment that can affect link quality, which we use for model training. In Section IV we describe the online protocol for data gathering, while the supervised learning procedure for model learning is described in Section V. Experimental results are reported in Section VI. In Section VII we draw conclusions and discuss future work.

II. Related Work

In general terms, the approaches used to characterize the elements affecting the performance of a wireless network, such as link quality, can be classified as *analytical* and *empirical estimators*. Analytical estimators [13], [14], [28] try to capture wireless channel properties by building theoretical models, mostly based on the modeling of radio propagation and interference [18]. On the other hand, empirical estimators use statistics collected during the network operation in order to predict current characteristics or future behavior.

According to the way they collect network measurements, empirical approaches can be further distinguished into: *passive measurement* and *active probing* schemes. Passive measurement schemes use the performance history of the network, obtained by overhearing local traffic, without interfering with the network environment or generating additional traffic [20], [31]. The information used by these techniques is limited to the configurations that have occurred in the past and to the characteristics of the wireless links that have been used so far. They usually fail to adapt to highly dynamic environments, despite of their attempts to capture short-term variations. Active probing approaches [2], [7]–[9], [17], [32], collect performance statistics from *probe* messages which are periodically sent out. Compared to passive measurement strategies, the additional overhead due to the injection of probe messages is compensated by a main advantage: the ability to examine areas and configurations of the network that otherwise would not be have been used for data traffic. This characteristic allows to proactively monitor large portions of the network and be able to better react to changes.

We propose a probing-based approach which, in contrast to previous active probing techniques, does not rely on the continuous transmission of probe messages while data traffic is being transmitted. MAPPLE's probing phase is executed in absence of application traffic, to gather training data under a controlled situation and to not create an unwanted traffic overhead. Moreover, none of the previous approaches explicitly rely on node mobility and/or node switching on and off to artificially generate multiple local topologies.

In addition to the mobility-assisted probing scheme, the main contribution of this paper is the definition of a machine learning approach to predict wireless link quality. While machine learning strategies have been employed since long to learn link qualities in the context of learning routing paths (e.g., [5], [9]), only a few works have proposed to use machine learning specifically for link estimation in wireless networks. In [12], the authors propose a pattern matching-based approach to predict link quality as a function of Signal to Noise Ratio (SNR) variations, based on historical measurements. In contrast, we consider a much larger set of measurable features of the local wireless environment, rather than focusing on a single one (the SNR, that we do not take directly take into account, also considering that it may lead to inaccurate predictions [37]). Moreover, we aim to learn a prediction model that is general, not related or depending on a specific link, and that can be applied on the spot, without the need of per-link historical measurements.

In [33], link quality estimation is approached as a classification problem. The authors propose to categorize links using a limited set of classes by means of a supervised learning approach, where a model is trained using a set of samples collected by the nodes. To characterize each sample, the authors identify a set of features that are believed to have an impact on link quality. In the paper, it is shown that the proposed link estimation strategy, combined with the ETX protocol [8], improves network performance. Rather than classify links, we use a support vector regression technique to directly estimate the packet reception rate of a given link given a characterization of its local environment. Moreover, the set of link features considered in [33] does not take into account the impact of traffic/interference generated by neighbor nodes, as we do. Finally, we propose a protocol to maximize the efficiency of data collection in terms of different samples, which has a significant impact on the accuracy and generalization capabilities of the model.

In [3] the authors propose a fuzzy logic link quality estimator. It employs membership functions to estimate the quality based on four features: packet delivery ratio, link asymmetry, stability, and SNR. In [23], three different machine learning techniques are used for link quality prediction. The authors combined hardware based estimators, readily available from the physical layer, together with a software based estimator, in order to obtain a more accurate model which predicts the current link packet reception probability. In general terms, apart from the differences related to the used learning techniques, our work can be distinguished from these previous works by the considered set of link characteristics. The features included in our modeling capture a larger number of elements affecting the performance of a wireless link. Also, most of these machine learning related approaches build statistics performing active probing concurrently with application traffic. In our framework, data collection is kept separated from the use of the network for data applications, to minimize the overhead and have a full control on mobility and traffic patterns.

III. LINK QUALITY AND FEATURE SELECTION

In order to define a model to estimate the quality of a wireless link, first, we need to identify the features of the local wireless environment that can have an impact determining the link performance. These features also depend on the adopted MAC protocol. In this paper, we focus on CSMA/CA protocols. However, our approach can be applied also when different MAC protocols are used, by selecting the appropriate set of features. The selected features define the data that are gathered during MAPPLE's first stage, and that are used as input for training the SVR in the second stage.

Since it is widely understood that the packet reception rate between two neighbor nodes is directly related to their physical distance [35], the first feature we consider is precisely the *distance* between the two end nodes of the wireless link. Distance estimations can be obtained through a variety of methods, such as GPS, wireless signal measures [19], and range measurements based on ultra-wideband or infrared [30].

The second feature we consider is the expected link *traffic load at the sender*, r_S. The traffic rate at which data flows through a link from sender to receiver has a clear impact on the percentage of expected packet losses. These losses might be caused by MAC contentions, and queue management failures, among others. For simplicity, we assume that the traffic load r_S belongs to one of three possible *traffic profiles*: $\{r_{low}, r_{med}, r_{high}\}$.

The *Received Signal Strength Indicator* (RSSI) is a hardware based estimator embedded in many wireless devices. It is usually available directly from the PHY layer and is commonly included in link quality prediction models. We also include it, even if its efficacy as a link quality estimator by itself has been disputed in many research works [4], [29].

The next feature that we consider is the *transmission rate at the receiver*, r_R. In typical scenarios, nodes both generate data to be transmitted and act as relays for packets generated by other nodes. In single-channel MAC protocols, this condition may cause to *destination busy* collisions [27]. These types of packet losses occur whenever the destination node of a packet is "busy" (i.e., performing a packet transmission). To account these losses in our model, we also include the receiver transmission rate as an element that affects link quality. As for the sender traffic rate, we assume that r_R can be characterized by one of the three mentioned traffic profiles.

In wireless networks, *interfering transmissions* contribute to poor packet delivery performance. Although MAC protocols are designed to prevent transmission interference, packet collisions might still occur (e.g., presence of hidden terminals in CSMA/CA protocols [28]). To consider this factor in our model, we include the neighborhood state of the receiver node in the set of link features by using a vector V_R of elements $(n_i, d_i,\ r_i)$, where n_i is an *active neighbor* (simultaneously transmitting) of the receiver, located at distance d_i from the receiver and transmitting at rate r_i.

Simultaneous transmissions from nodes in the vicinity of the sender also affect link quality. Due to the shared nature of the wireless channel, contention-based MAC protocols prevent transmissions from nodes within a certain distance in order to reduce packet collisions. However, when the neighborhood of the sender is congested (due to high number of active neighbors or heavy local traffic), packet losses may still occur because the sender's MAC protocol is unable to acquire access to the wireless channel. For this reasons, the *sender's neighborhood state* is another feature that we consider in our modeling. We use as feature the vector V_S, including the set of concurrently transmitting nodes in the sender's neighborhood. Each element of V_S is a tuple (n_j, d_j, r_j) representing a neighbor node n_j, located at distance d_j from the sender and transmitting at rate r_j.

It is important to remark that all the information needed to define the feature vector for a given link can be made available at the receiver node using its local sensors (i.e., radio transceiver, range and bearing) and the probe packets received from node the source node. Therefore, in practice, collecting link quality measurements with their respective feature values can be done in a fully decentralized and online way.

IV. Mobility-assisted channel probing

With the aim of gathering a large and diverse set of samples of local wireless configurations, we propose a three-state distributed protocol. At any given time, a node can be in one of the following three states:

- *Idle:* The node is inactive. This may be implemented by switching off its radio or stopping sending and receiving;
- *Probing:* In this state a node periodically broadcasts probe packets containing information about its neighborhood and the wireless channel. At the same time, the node receives and precesses probe packets sent by other nodes;
- *Moving:* In this state a node performs a controlled movement to enforce changes in the network topology.

Every time a node makes a state transition, it broadcasts a *state change* message to inform its neighbors about the new state. As the protocol is running, each node uses this message information to keep updated its neighbor table.

A. Wireless channel probing

When a node switches to a probing state, it randomly chooses a traffic profile in $\{r_{low}, r_{med}, r_{high}\}$. According to the profile, it periodically generates and locally broadcasts *probe* packets. Each probe packet from a node a contains: (i) a message counter; (ii) the rate at which a is transmitting packets; (iii) information about a's neighborhood, represented by the status vector V_S; (iv) the time stamp of the last change of a's neighborhood state (the counter value at the moment of receiving a *state change* message from any of its neighbors).

If $a \rightarrow b$ is a directed wireless link from node a to node b, then samples from link $a \rightarrow b$ are collected by a whenever both a and b are in *probing* state. At the reception of the first probe packet from a, b records the message counter and the time stamp and initiates a *sampling session* for a. Every new probe packet is examined and counted. If a probe packet contains a different value for the time stamp, it indicates that node a's neighborhood has changed. When this happens, the current sampling session is closed: session's packet reception rate is computed using the first and the last message counters successfully received, and the state of node a and b is recorded and used to define the wireless enviroment of the sampled data. A new sampling session is then initiated for link $a \rightarrow b$. Another situation leading to a restart of the local data collection procedure is the reception of a state change message. For instance, node b receives a state change message which changes its set of active neighbors while it was sampling data regarding link $a \rightarrow b$. This event implies a change in the local environment of link $a \rightarrow b$. Therefore, the current sample must be closed. This condition holds for any other link being sampled at that moment. Data collected during a sampling session are stored only if the counter value (i.e., the number of packets sent during the session) is greater than a threshold value. This constraint is defined to ensure sampling intervals that are long enough to provide representative measures of the packet reception rate of a given link.

B. Topology variations by node mobility

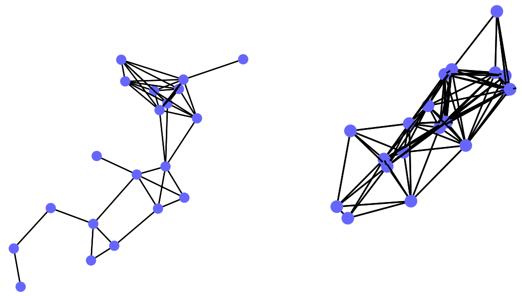
By switching to *idle* and *moving* states, nodes alter the network topology and change the local environment of sampled links. In idle state, a node excludes itself from the network, reducing in this way the number of nodes and creating a locally sparser network. On the other hand, node movements modify the network structure by adding new links, and/or removing existing ones. New links are created if the distance between two nodes which were previously out-of-range decreases to a value that allows both nodes to receive each others transmissions. Vice versa, existing links disappear from the topology if, after moving, the inter-node distance exceeds the transmission range.

Node mobility is used with the objective of strongly reshaping network topology, generating a number of novel local link environments to be sampled. Several mobility strategies can be used at this aim. To preserve distribution and simplicity of the protocol, we adopt a simple random mobility strategy to select node trajectories. When a node enters into a moving state, it randomly chooses a direction and starts moving according to it. The movement is performed until a neighborhood change is detected, that is, until a topology change is ensured. To prevent nodes becoming disconnected from the rest of the network, we enforce them to return to the last position where the node had at least one neighbor if the node becomes isolated. We set the maximum displacement to be relatively short, to minimize the time a node spend performing movements (this time depends on the mobility capabilities of the node). Figure 1 illustrates the network topology evolution after running the MAPPLE protocol for a few minutes. In the figure, the edges represent the wireless links present in the network. The transmission range is set to 8 meters.

By relying on node mobility, the protocol can be quite energy consuming. On the other hand, the switching on and off of the nodes partially counterbalances the energy consumption due to mobility. The investigation of energy-aware mobility strategies and the trade-off between energy consumption and system performance is left for future research.

C. State transitions

State changes modify the neighborhood of probing nodes, triggering, in turn, the restarts of sampling sessions. If these restarts are frequent, the number of link samples decreases, because of the constraint on the minimum number of per session probe messages required to generate a valid data sample for learning. We define the time between two consecutive state changes as a random variable uniformly distributed in (min_s, max_s). The interval sets the trade-off between number and diversity of the samples, and can accommodate the requirements for energy optimization and execution time of the protocol. In our evaluation, we set $min_s = 10s$ and $max_s = 30s$. The impact on model learning of the different settings for min_s and max_s will be the subject of future work.



(a) Initial topology (b) Topology after 5 minutes

Fig. 1: Topology evolution using MAPPLE's mobility.

V. Building a prediction model with SVR

After collecting the measurements of link quality at the individual nodes, the sampled data can be gathered at a base station, or a specialized processing node, to build a database of samples, which is then used to train a SVR model.

A. Feature preprocessing

Given that the selected features all have numerical values, we first need to find a manageable representation for the information contained in the vectors V_R and V_S.

Definition 1. *Let* $D = [d_{min}, d_{max}]$ *be the range of values for link distances, and* $R = \{r_{low}, r_{med}, r_{high}\}$ *be the set of traffic profiles. Let* $\bar{D} = \{\bar{d}_1, \ldots, \bar{d}_n\}$ *be a partition of the interval* D*. A feature set is defined as:* $F = \{f_{ij} \mid i \in \bar{D},\ j \in R\}$.

Using the definition above, two sets of features, F_R and F_S, are defined to represent the data contained respectively in each vector V_R and V_S. The information is compacted by classifying the vector elements and counting the members for each class. For instance, the value of feature f_{ij} of F_S is the number of elements of V_S that have a distance value less than $\bar{d}_i$ and traffic rate r_j. A summary of all the considered features is reported in Table I.

d	Link distance
r_S	Traffic generation rate at sender
r_R	Traffic generation rate at receiver
$RSSI$	Average of RSSI measures obtained at receiver
F_S	Neighborhood state of sender node
F_R	Neighborhood state of receiver node
$\lvert V_S\rvert$	Number of active neighbors of sender node
$\lvert V_R\rvert$	Number of active neighbors of receiver node

TABLE I: Features used for model building.

B. Feature ranking

Since irrelevant features degrade the prediction performance of the link quality, it is beneficial to perform *attribute selection* to eliminate redundant features. We use the well-established feature attribute selection approach of Weka [15] , namely the $InfoGainAttributeEval$, to evaluate the worthiness (weightage) of an attribute, using the following expression [33]:

$$InfoGain(D, Attr) = H(D) - H(D|Attr), \qquad (1)$$

where $H(D|Attr)$ is the entropy of the training set D given attribute $Attr$, and $H(D)$ is the entropy of D. Entropy has been widely used in machine learning problems to represent the amount of disorder an attribute contains with respect to the target output (link quality). In particular, $H(D) = -(\sum_{i=1}^{m} p_i \log_2(p_i))$ where p_i is the probability that an arbitrary sample in D belongs to the target output T_i. If the samples in D on attribute $Attr$ have v distinct values $\{a_1, a_2, \ldots, a_v\}$, then $H(D|Attr) = -\sum_{j=1}^{v} H(Dj)(|D_j| \cdot |D|)$, where D_j contains samples in D that have a_j as outcome for $Attr$.

From Table II, it results that features d and $RSSI$ are the most critical ones to differentiate good links from bad ones (both when mobility is used and not used). However, since all features contribute with a significant weightage in predicting link quality, all 8 features of Table I are used to train the Support Vector Regression model.

	d	RSSI	r_S	r_R	$\lvert V_R\rvert$	F_R	$\lvert V_S\rvert$	F_S
Mobility	1.73	0.46	0.025	0.014	0.16	0.25	0.09	0.25
No mobility	1.08	0.395	0.032	0.041	0.14	0.17	0.035	0.27

TABLE II: Feature ranking using information gain.

C. Model Training

Each sample acquired from the data collection protocol consists of a measured packet reception rate (link quality), y, and a vector of 8 feature values (see previous section), $\mathbf{x}$, that summarize its local wireless environment. For link quality learning, we use an ϵ-SVR approach, that non-linearly maps the input data features $\mathbf{x}$ into a higher dimensional feature space using a *kernel* [26]. Thus, given a set of N training data $G = \{(y_i, \mathbf{x}_i),\ i = 1, 2, \ldots, N\}$, where $\mathbf{x}_i$ and y_i represent respectively the i-th input feature vector and i-th actual target output (link quality), the ϵ-SVR learns the relationship between the features $\mathbf{x}_i$ and y_i for each sample.

We use k-fold *cross-validation* (CV) to avoid the overfitting of the training data while providing good generalization. In k-fold CV, the training set is divided into k subsets of equal size. In sequence, each subset is tested using the trained ϵ-SVR on the other $k-1$ subsets. Since each sample in the entire training set is predicted once, the CV accuracy indicates the percentage of samples which are correctly predicted.

Before starting training the ϵ-SVR model, all feature vectors $\mathbf{x}_i$ in the training set are normalized in $[0, 1]$. To allow the ϵ-SVR model to fit the training data accurately and produce good predictions, the following internal parameters need to be tuned: error penalty parameter C, Gaussian Radial Basis Function kernel parameter σ, and loss function parameter ϵ. A model selection (parameter search) is performed to identify the best parameter combination (C, σ, ϵ) for the given set of data. At this aim, we use the *Grid Search* method of Hsu et al. [6]. In Grid Search, various pairs of ϵ-SVR parameters are evaluated, where for each setting of a parameter pair, a CV accuracy is obtained. Evidence from the literature [6] indicates that using exponentially growing sequences of ϵ-SVR parameters, $C = [2^{-1}, 2^0, 2^1, \ldots, 2^6]$, $\sigma = [2^0, 2^{-1}, 2^{-2}, \ldots, 2^{-10}]$ and

$\epsilon = \left[2^{-10}, 2^{-9}, 2^{-8}, \ldots, 2^{-1}\right]$, with k-fold CV is effective to identify the best ϵ-SVR parameters for a set of training data. The ϵ-SVR parameter pair with the highest CV accuracy was used on the entire training set to build the ϵ-SVR model.

VI. System evaluation

In this section, we evaluate the prediction accuracy of our SVR model by comparing the estimates it provides against the empirical measures obtained through *realistic network simulations* using the *TOSSIM* simulator [22]. The current version of TOSSIM does not support mobility, therefore we added this functionality to the simulator. The mobility model is implemented following the guidelines described in Section IV.

A. Evaluation process

The evaluation process was organized in three steps. First, through network simulation, we collected link quality samples by running the data collection protocol for a small set of 20-nodes networks. In order to assess the impact of mobility on model prediction accuracy, we ran also experiments with a simplified version of the protocol, in which we excluded node mobility. By these two different procedures, we obtained two separate sets of measurements.

In the second step, these sets are used to train two independent ϵ-SVR models. Finally, we validated the accuracy of these models by comparing their predicted link quality estimates against the link quality measurements obtained by simulation in networks of different sizes and under different traffic loads.

B. ϵ-SVR model training

As mentioned in the previous section, two different sets of training samples are acquired, one using mobility and the other without node mobility. The number of samples acquired in both cases is set to t_s = 10,000. We use the LIBSVM library [16] to implement the ϵ-SVR technique. The models are constructed in a two-phase approach. First, we use the Grid Search method discussed in section V-C to identify the best ϵ-SVR parameters for both models using k-fold CV with k = 10. Then, the best parameters are used to build two separate models, as shown in Table III. The CV *Mean Square Error* (MSE) and CV *Squared Correlation Coefficient* (SCC-r^2) [6] are the error metrics we used for evaluating the performance of the ϵ-SVR on the training sets. In supervised learning regression problems, the lower the MSE the better will be the prediction performance during CV training, and the higher the SCC the better will be the learning performance. In addition, the smaller the number of *Support Vectors* (SVs) in a model, the lower will be its complexity, and the faster will be its performance for online testing. Therefore, from Table III is evident that the model using mobility has better learning and computational performance for online testing.

Training	Model with mobility	Model without mobility
CV-MSE	0.00607	0.01007
CV-SCC (r^2)	0.7223	0.5950
No. of SVs	6480	7029
Param.	$c = 0.5, \sigma = 0.5, \epsilon = 0.031$	$c = 2, \sigma = 0.13, \epsilon = 0.063$

TABLE III: Training results for the ϵ-SVR model.

C. Evaluation of the link quality prediction performance

To test the trained ϵ-SVR models online, we ran network simulations using a number of topologies of different sizes, from 10 up to 100 nodes, and we measured the link qualities. Figure 2 shows the diversity of the considered network topologies in terms of node degree. For each network size, we compare the actual and the predicted values for 20,000 samples using both models. By comparing actual and predicted values we compute the absolute difference for each testing sample.

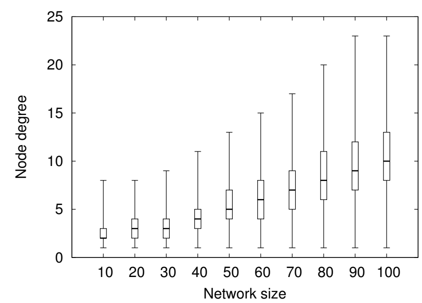


Fig. 2: Mean node degree diversity of tested topologies.

Figure 3 shows, for each of the considered network sizes, the distribution of the absolute deviation between actual and predicted values. In the large majority of the cases, the deviation between observed and predicted values is very small, with a median around 0.1 for the case using mobility and 0.15 not using mobility. In both cases, the performance does not degrade significantly with the increasing of network size. The

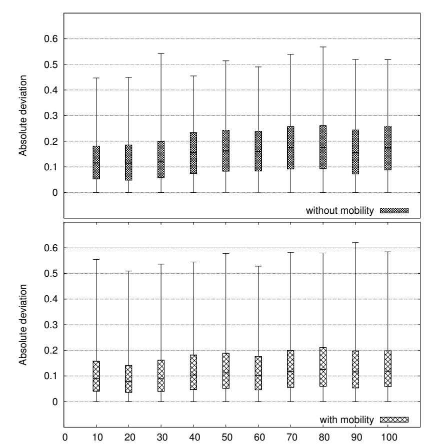


Fig. 3: Evaluation results for model accuracy.

results show the overall efficacy of the MAPPLE approach, and its ability to build a prediction model with very good generalization capabilities: using networks of only 20 nodes to collect data, we could reliably predict the link quality for networks up to 5 times larger. The impact of mobility on the accuracy of the model is significant, especially for larger networks, where the approach without mobility starts to progressively degrade its performance. Overall, the results show that the designed combination of probing, mobility, and supervised learning is effective for the considered problem.

VII. Conclusion

In this paper we presented MAPPLE, a mobility-assisted probing-based scheme using supervised machine learning, namely Support Vector Regression, to estimate link quality in mobile wireless networks. MAPPLE consists of two steps. In the initial step (mobility-assisted proactive probing), which is performed when user applications are not active, the nodes artificially generate data traffic (message probes) and perform movements in order to collect measurements representative of different network conditions. In the second step, the gathered data are used to train an ϵ-SVR model to predict link qualities. The model could be then deployed to the nodes and used for the estimation of link qualities during online network operations (e.g., by a routing protocol). Results obtained by extensive network simulations show the very good prediction accuracy of the proposed system. Moreover, they show that the prediction model is able to generalize to networks that are much larger and with very different layouts than those used for learning. This can found applications in many mobile and dynamic scenarios. Future work will include testing in real networks, the use of incremental learning techniques to let the system working online, concurrently with user applications, and the investigation of energy-aware strategies for mobility.

Acknowledgment

This research has been partially funded by the Swiss National Science Foundation (SNSF) Sinergia project SWARMIX, project number CRSI22_133059.

References

[1] I. Akyildiz. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, Mar. 2002.

[2] U. Ashraf, S. Abdellatif, and G. Juanole. An interference and link-quality aware routing metric for wireless mesh networks. In *Proc. of the IEEE Vehicular Technology Conference*, pages 1–5, Sept. 2008.

[3] N. Baccour, A. Koubâa, H. Youssef, M. Ben Jamâa, D. do Rosário, M. Alves, and L. Becker. F-LQE: A fuzzy link quality estimator for wireless sensor networks. In *Wireless Sensor Networks*, volume 5970 of *Lecture Notes in Computer Science*, pages 240–255. 2010.

[4] N. Baccour, A. Koubâa, M. Zuniga, H. Youssef, C. Boano, and M. Alves. Radio link quality estimation in wireless sensor networks: a survey. *ACM Transactions on Sensor Networks (to appear)*, 2011.

[5] J. Boyan and M. Littman. Packet routing in dinamically changing networks: A reinforcement learning approach. In *Advances in Neural Information Processing Systems 6 (NIPS6)*, pages 671–678, 1994.

[6] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, 2011.

[7] M. Corson. A distributed channel probing scheme for wireless networks. In *Proc. of IEEE INFOCOM 2001*, pages 403–411, 2001.

[8] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proc. of MobiCom '03*, pages 134–146. ACM Press, 2003.

[9] G. A. Di Caro and M. Dorigo. AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research (JAIR)*, 9:317–365, 1998.

[10] G. A. Di Caro and E. Feo Flushing. Optimal relay node placement for throughput enhancement in wireless sensor networks. In *Proc. of the 50th FITCE International Congress*, Palermo, Italy, 2011.

[11] R. Draves, J. Padhye, and B. Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proc. of of MobiCom '04*, pages 114–128. ACM Press, 2004.

[12] K. Farkas, T. Hossmann, F. Legendre, B. Plattner, and S. Das. Link quality prediction in mesh networks. *Computer Communications*, 31(8):1497–1512, May 2008.

[13] Y. Gao, D. Chiu, and J. Lui. Determining the end-to-end throughput capacity in multi-hop networks: methodology and applications. *ACM SIGMETRICS Performance Evaluation Reviews*, 34:39–50, June 2006.

[14] P. Gupta and P. Kumar. The capacity of wireless networks. *IEEE Trans. on Information Theory*, 46(2):388–404, 2000.

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[16] C. Hsu, C. Chang, and C. Lin. A practical guide to support vector classification. Technical report, National Taiwan Univ., Taiwan, 2003.

[17] N. Hu and P. Steenkiste. Estimating available bandwidth using packet pair probing. Technical Report 02-166, Carnegie Mellon Univ., 2002.

[18] A. Iyer, C. Rosenberg, and A. Karnik. What is the right model for wireless channel interference? *IEEE Trans. on Wir. Comm.*, 8(5):2662–2671, May 2009.

[19] X. Ji and H. Zha. Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling. In *Proc. of IEEE INFOCOM*, 2004.

[20] G. Judd, X. Wang, and P. Steenkiste. Efficient channel-aware rate adaptation in dynamic environments. In *Proc. of MobiSys '08*, pages 118–131. ACM Press, 2008.

[21] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *Proc. of IPSN'06*, pages 2–10. ACM, 2006.

[22] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: Accurate and scalable simulation of entire TinyOS applications. In *Proc. of SenSys'03*, pages 126–137. ACM, 2003.

[23] T. Liu and A. E. Cerpa. Foresee (4C): Wireless link prediction using link features. In *Proc. of IPSN'11*, pages 294–305, 2011.

[24] T. Melodia, D. Pompili, and I. Akyildiz. A communication architecture for mobile wireless sensor and actor networks. In *Proc. of IEEE SECON'06*, pages 109–118. IEEE, 2006.

[25] J. Nagi, K. Yap, F. Nagi, S. Tiong, and S. Ahmed. A computational intelligence scheme for the prediction of the daily peak load. *Applied Soft Computing*, 11(8):4773–4788, 2011.

[26] J. Nagi, K. Yap, S. Tiong, S. Ahmed, and M. Mohamad. Nontechnical loss detection for metered customers in power utility using support vector machines. *IEEE Trans. on Power Delivery*, 25(2):1162–1171, 2010.

[27] A. Nasipuri, J. Zhuang, and S. Das. A multichannel CSMA MAC protocol for multihop wireless networks. In *Proc. of IEEE WCNC'99*, volume 3, pages 1402–1406, 2002.

[28] J. Padhye, S. Agarwal, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill. Estimation of link interference in static multi-hop wireless networks. In *Proc. of ACM SIGCOMM IMC '05*, page 1. ACM Press, 2005.

[29] C. Renner, S. Ernst, and C. Weyer. Prediction accuracy of link-quality estimators. In *Proc. of EWSN'11*, 2011.

[30] J. Roberts, T. Stirling, J. Zufferey, and D. Floreano. 2.5D infrared range and bearing system for collective robotics. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3659–3664, 2009.

[31] S. H. Shah, K. Chen, and K. Nahrstedt. Available bandwidth estimation in IEEE 802.11-based wireless networks. In *Proc. of 1st ISMA/CAIDA Workshop on Bandwidth Estimation*, 2003.

[32] T. Sun, M. Sanadidi, and M. Gerla. Ad hoc probe: path capacity probing in wireless ad hoc networks. In *Proc. of WICON'05*, pages 156–163, 2005.

[33] Y. Wang, M. Martonosi, and L.-S. Peh. Predicting link quality using supervised learning in wireless sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 11(3):71, July 2007.

[34] R. Wattenhofer and a. Zollinger. XTC: A practical topology control algorithm for ad-hoc networks. *Proc. of the 18th International Parallel and Distributed Processing Symposium*, pages 216–223, 2004.

[35] Y. Yang, J. Wang, and R. Kravets. Designing routing metrics for mesh networks. In *Proc. of WiMesh*, 2005.

[36] Y. Yang, J. Wang, and R. Kravets. Interference-aware load balancing for multihop wireless networks. Technical Report 361702, University of Illinois at Urbana-Champaign, 2005.

[37] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proc. of ACM SenSys '03*, pages 1–13, 2003.