A Lightweight Approach to Online Detection and Classification of Interference in 802.15.4-based Sensor Networks

Frederik Hermans
IT Department
Uppsala University
frederik.hermans@it.uu.se

Olof Rensfelt
IT Department
Uppsala University
olofr@it.uu.se

Lars-Åke Larzon
IT Department
Uppsala University
lln@it.uu.se

Per Gunningberg
IT Department
Uppsala University
perg@it.uu.se

ABSTRACT

With a rapidly increasing number of devices sharing access to the 2.4 GHz ISM band, interference becomes a serious problem for 802.15.4-based, low-power sensor networks. Consequently, interference mitigation strategies are becoming commonplace. In this paper, we consider the step that precedes interference mitigation: interference detection. We have performed extensive measurements to characterize how different types of interferers affect individual 802.15.4 packets. From these measurements, we define a set of features which we use to train a neural network to classify the source of interference of a corrupted packet. Our approach is sufficiently lightweight for online use in a resource-constrained sensor network. It does not require additional hardware, nor does it use active spectrum sensing or probing packets. Instead, all information about interferers is gathered from inspecting corrupted packets that are received during the sensor network's regular operation. Even without considering a history of earlier packets, our approach reaches a mean classification accuracy of 79.8%, with per interferer accuracies of 64.9% for WiFi, 82.6% for Bluetooth, 72.1% for microwave ovens, and 99.6% for packets that are corrupted due to insufficient signal strength.

Categories and Subject Descriptors

C.2.1 [**Computer Communication-Networks**]: Network Architecture And Design—*wireless networks*; C.2.3 [**Computer Communication-Networks**]: Network Operations—*network monitoring*

General Terms

Measurement, Design

Keywords

Interference, interference detection, interference classification, sensor networks

1. INTRODUCTION

The unlicensed 2.4 GHz ISM band is heavily populated by wireless devices as diverse as WiFi laptops, Bluetooth headsets, baby monitors and microwave ovens [6]. Consequently, sensor networks operating in this frequency band increasingly suffer from cross-technology interference. This is a severe problem because even unsuccessful communication attempts take a toll on the sensor nodes' scarce energy budgets. Mitigation strategies aim to warrant timely and energy-efficient communication even in the presence of interferers. Knowing the type of interference a network is exposed to can increase the effectiveness of mitigation, resulting in higher packet delivery ratio and ultimately lower power consumption [13, 8].

To facilitate useful mitigation decisions, we consider the problem of interference detection and classification in 802.15.4-based sensor networks. This is a challenging task, because 802.15.4 devices generally cannot decode transmissions from interferers using other radio technologies. Existing approaches actively sample the spectrum for activity from interferers [1, 16] or use dedicated hardware such as software-defined radios [12]. However, spectrum sampling is a very energy-consuming task for sensor networks, and adding dedicated hardware increases both price and complexity of sensor nodes.

In this paper, we describe an approach that enables resource-constrained sensor nodes to classify individual corrupted 802.15.4 packets according to the cause of corruption. Using data from our extensive measurements on how different interferers affect 802.15.4 communication, we show that each interferer has characteristic patterns that emerge from observing the (1) Link Quality Indicator (LQI) of an interfered 802.15.4 packet, (2) the signal strength during packet reception, and (3) information about what parts of the packet are corrupted. We define a set of *features* on these three observations that extract the essential information. Our features are sufficiently lightweight so that a sensor node can compute them for a given corrupted packet. A neural network maps features to an interference class, i.e., it allows to determine the type of interferer from the data collected about an individual corrupted packet. We implement a fixed-point neural network on the TelosB platform to demonstrate the feasibility.

A key strength of our approach is its resource efficiency: It does not require active spectrum sensing, additional hardware, or probing packets. Instead, it gathers information about interference only during the regular operation of the sensor network. Assuming that the network uses either forward error correction or retransmissions, our approach does not incur any communication overhead. The main energy cost of our approach comes from turning on the Micro Controller Unit (MCU) during packet reception, whereas usually it

would be woken up only when packet reception is completed.

The contributions of this paper are:

- We identify interference patterns that distinguish interference sources, and show how from information that can be gathered during a sensor network's regular operation, we can define features that allow to capture the essence of these patterns.
- The features and the classification method we use are chosen to be sufficiently lightweight so they can be computed on a resource-constrained sensor node.
- We evaluate the classification accuracy of our approach with respect to four different causes of packet loss: interference from WiFi networks, Bluetooth networks, and microwave ovens, as well as packet loss due to insufficient sender TX power. The mean classification accuracy is 79.8%.

While in this paper we focus on three types of interferers (WiFi, Bluetooth, microwave ovens) and packet loss due to insufficient signal strength, our approach is not confined to detecting these. To classify an additional interferer type requires collecting appropriate training data, re-training the classifier, and updating the classification parameters on the sensor nodes. It may also be necessary to define additional features.

The paper is organized as follows: A brief summary of relevant radio technologies is given in Sec. 2. We describe our interference measurements in Sec. 3. Sec. 4 describes the core ideas of our approach: how features can be extracted from data about corrupted packets, and how these features can be used to classify packets according to the cause of corruption; we also present implementation and overhead considerations. In Sec. 5, we evaluate the classification accuracy. Related work is described in Sec. 6 and we present conclusions and an outlook on future work in Sec. 7.

2. TECHNICAL BACKGROUND

We briefly summarize the technical aspects of 802.15.4, 802.11b/g, Bluetooth, and microwave ovens that are relevant to our goal of interference classification.

802.15.4

The 802.15.4 standard defines a physical layer and a MAC layer for low-power, low-rate wireless networks [10]. We consider 802.15.4 at 2.4 GHz in this paper, because most interference is faced in this popular ISM band. At 2.4 GHz, 16 channels of 2 MHz width are defined with an inter-channel spacing of 3 MHz. A maximum transmission power of 0 dBm is common. The standard implements direct sequence spread spectrum by mapping each four-bit symbol to be transmitted to a pseudo-random 32-chip sequence. Offset quadrature phase-shift keying is used for modulation. The data rate is 250 kbps, the symbol period is 16 µs.

The format of an 802.15.4 PHY packet is shown in Fig. 1. Each packet begins with a preamble, which consists of four zero bytes, followed by a one-byte start frame delimiter (SFD) field with a fixed value. The frame length field contains the number of the packet's payload bytes, which may be up to 127. The length includes the two-byte frame check sequence (FCS) field which trails the packet payload. The FCS field contains a checksum which

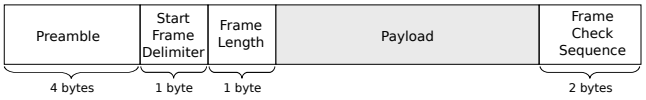


Figure 1: Format of an 802.15.4 packet

is calculated over the length field and the payload bytes. A receiver synchronizes to incoming zero-bytes; after receiving four zero-bytes, it scans for an SFD. Only after correctly receiving the SFD, it reads the following payload field and then reads the specified number of payload bytes. If no SFD is received after four zero-bytes, the receiver synchronizes to incoming zero-bytes again. A receiver can detect transmission errors by comparing the FCS against the checksum calculated for the received packet.

802.11b/g

The amendments b and g describe the two most prevalent physical layer implementations of the 802.11 standard for wireless local area networks [11]. Fourteen channels of 22 MHz width are defined, of which eleven are available worldwide. Nominal transmission power is between 15 dBm and 20 dBm, and bit rates range from 1 MBit/s to 54 MBit/s using a variety of modulations. The MAC layer is a variant of CSMA/CA and mandates the minimum time between two frames.

Bluetooth

Bluetooth is a standard for short-range radio communication within personal area networks [9], and is often used for connecting peripherals. Bluetooth divides the 2.4 GHz band into 79 bands of 1 MHz each and employs frequency hopping over these bands. Devices in a network synchronize on a hopping sequence, and stay on each frequency for a slot duration of 625 µs. The most common class of Bluetooth devices has a maximum transmission power of about 4 dBm.

Microwave oven

Residential microwave ovens are kitchen appliances that are used to heat food using non-ionizing microwave radiation in the 2.4 GHz band. Usually the whole band is affected by the emissions, with ovens exhibiting a frequency sweeping behavior. Minor variations in the affected frequencies can be observed for different models. The ovens' emissions alternate between on and off phases every $\frac{1}{2f}$ s during the heating period, where f is the mains frequency. The equivalent isotropically radiated power (EIRP) has been reported to range between 16 dBm and 33 dBm [5].

3. INTERFERENCE MEASUREMENTS

We conducted a series of experiments in which an 802.15.4-based sensor network was exposed to radio interference. In each experiment, we activated one interference source and collected data on the corrupted 802.15.4 packets. The purpose of these experiments is twofold. First, we collected corrupted packets to gain a better understanding of the effects that different interferers have on individual 802.15.4 packets. Second, we use subsets of the data for training and evaluating our classification approach.

The experiments were carried out in an anechoic chamber, which is shielded from outside radio transmissions. Such a controlled environment gives us high confidence that a corrupted 802.15.4 packet recorded during an experiment was indeed corrupted by the source

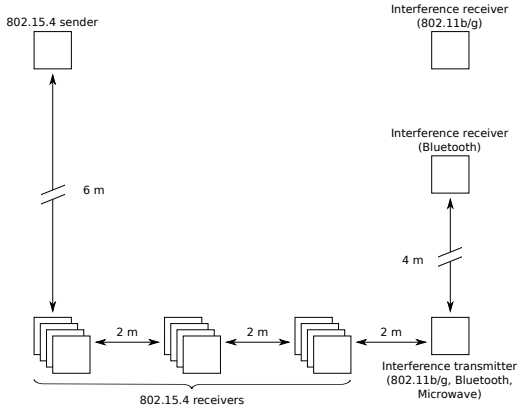


Figure 2: Experiment layout

Figure 3: The microwave interferer and four of the sensor nodes (on a tripod) in the anechoic chamber

of interference that we have activated. In a less controlled environment, e.g., our university building, it is virtually impossible to prevent radio devices that are outside of our control from affecting the experiments. The anechoic chamber is also constructed to minimize multipath propagation. This is desirable, because multipath propagation is strongly dependent on the concrete physical layout of an environment, and we do not want to capture environment-specific effects in our measurements.

3.1 Experiment Setup

The experiment setup is shown in Fig. 2. We used TelosB sensor nodes [4], which have a 802.15.4-compliant CC2420 transceiver [20]. One node acted as a sender and twelve nodes were receivers. The receivers were divided into groups of four nodes and receivers within one group were placed close to each other. With this setup, receivers within one group all have similar distance to the sender and interferer, which allows us to separate effects of distance from actual interference.

We placed different interferers in the anechoic chamber during the experiments. The position of the interfering transmitter, and the receiving counterpart (in case of WiFi and Bluetooth) is also shown in Fig. 2. We describe the interferers and their parameters in more detail in the next subsection.

During each experiment, the 802.15.4 sender periodically broadcasted packets. We added a jitter to the transmission period to avoid accidental synchronization with an interferer. The sender's transmission power was fixed to -15 dBm, unless mentioned otherwise. We confirmed that when no interferer was active, the packet reception rate was close to 100% at a transmission power of -15 dBm. As we want to focus exclusively on the effects of interference, no MAC layer was used in the sensor network, and the sender did not perform carrier sensing prior to sending. We varied the length of the payload of 802.15.4 packets between 16, 32, 64, 96 and 124 bytes, excluding the FCS field. The receiving nodes logged all packets they received, regardless of whether they passed the CRC check or not. Notice that nodes can only log packets for which they can successfully decode the preamble and the SFD field. For data collection and experiment control we use Sensei-UU, our nomadic sensor network testbed [17].

3.2 Interferers

We consider three different technologies that commonly cause interference to 802.15.4-based sensor networks: IEEE 802.11b/g networks, Bluetooth networks, and household microwave ovens. For each interferer, we ran experiments with different models from different vendors to avoid model-specific effects. Two different WiFi chipsets, three different microwave ovens, and three different Bluetooth devices were used.

802.11b/g

The devices operated in infrastructure mode. The access point (referred to as interference transmitter in Fig. 2) sent constant bitrate UDP traffic saturating the WiFi link to the receiver, which acted as a stationary client. We chose this scenario to resemble a video streaming session. The following parameters were varied between experiment runs: WiFi transmission power (4 dBm to 22 dBm), WiFi rate (1 MBit/s up to 54 MBit/s), 802.15.4 channel. The WiFi channel was fixed to 5, so by varying the 802.15.4 channel we controlled the frequency distance between WiFi and 802.15.4.

Bluetooth

Bluetooth traffic was created by one Bluetooth dongle (interference transmitter) sending back-to-back L2CAP packets to another Bluetooth dongle (interference receiver). Since Bluetooth performs adaptive frequency hopping, the dongles were reset at the beginning of each experiment, such that the dongles were oblivious with regard to the frequency on which the 802.15.4 network would be active. Between experiment runs, we varied the 802.15.4 channel and 802.15.4 transmission power.

Microwave oven

Using a spectrum analyzer, we confirmed that the ovens were mostly active around 802.15.4 channels 18 and 23. A bowl of water was heated in the microwave during each experiment run. The effect was set to maximum power in all runs, because changing power only resulted in varying length of idle periods between the oven's heating periods. Between experiment runs, we varied the 802.15.4 channel.

Insufficient signal strength

We also ran experiments in which the 802.15.4 sender sent packets with a transmission power of -25 dBm without any interferer being active. At this transmission power, the 802.15.4 receivers experienced significant packet loss due to an insufficient SNR. This

experiment serves as a reference case for packet loss that was not caused by interference.

4. INTERFERENCE DETECTION AND CLASSIFICATION

We describe a classifier that assigns each incorrectly received packet to an interference class. Each interference class represents a source of interference: WiFi, Bluetooth, or microwave. We further define an additional class to represent packets that have been received incorrectly in the absence of an interference source due to insufficient signal strength at the receiver.

In this section, we first consider what data can be feasibly gathered in a sensor network for such a classification task; we then consider how this data can be condensed into numerical features. We discuss suitable classification algorithms, and finally consider implementation, energy cost and overhead.

4.1 Extracting Information about Interference from Corrupted 802.15.4 Packets

As described in Sec. 2, an 802.15.4 sensor node can detect transmission errors by matching the payload against the CRC at the end of the packet. We consider what information a node could extract from a corrupted packet to help the goal of determining the cause of packet corruption. Only information that can be gathered with moderate energy costs and without additional hardware is regarded.

LQI

The 802.15.4 standard requires transceivers to provide a Link Quality Indication (LQI) measurement for each received packet. Transceivers such as the CC2420 determine LQI in terms of the chip error rate for the first eight symbols following the SFD field. Thus, LQI describes how well the first eight received chip sequences match any of the known chip sequences. LQI is popularly used by link quality estimators. Obtaining the LQI for a received packet incurs no additional energy cost.

RSSI during packet reception

The standard also requires transceivers to be capable of providing an estimate of the received signal power within the bandwidth of the currently selected channel, called Received Signal Strength Indication (RSSI). The CC2420 measures RSSI over a 128 µs window (8 symbol periods). For each packet, it provides an RSSI measured over the packet's first eight symbols by default. This RSSI measurement is also commonly used in link quality estimators as a metric of the channel's quality. Further RSSI measurements can be requested from the transceiver explicitly.

Rather than considering only one RSSI measurement taken at the beginning of reception, a more fine-grained view of how signal strength changes while a packet is being received potentially allows us to learn about the presence of high-power interferers. We therefore modify the CC2420 radio driver in the Contiki OS source code to continuously measure RSSI while a packet is being received. Our modified driver produces a series of RSSI values for each received packet, containing about one sample per payload byte. Examples of RSSI during packet reception are shown in Fig. 4.

By measuring the signal strength only during packet reception, we ensure that we exclusively capture changes in channel conditions that could have an impact on the sensor network communication. In contrast, if we sampled the channel while there is no ongoing sensor network communication, we may capture emissions from a non-interfering radio devices, e.g., devices that back off during ongoing 802.15.4 transmission.

Incorrectly received symbols

We assume that a sensor node can determine what parts of a received packet are corrupted, i.e., which symbols of the packet's payload have been decoded incorrectly. This assumption can be warranted if either forward error correction (FEC) or a retransmission protocol is used. If the sensor network applies FEC and a sensor node can successfully reconstruct a damaged packet, it can immediately determine which parts of the packet were corrupted. If no FEC is used, a sensor node may still be able to determine the corrupted parts of a packet if a retransmission protocol is used. In that case, the sensor node can keep recently received corrupted packets in a buffer, and probabilistically match successfully received packets against the corrupted packets in the history. In this way, it is possible for the node to identify which parts of the packets stored in the history are damaged. Specifying a method to perform this matching is outside of the scope of this paper and we leave it to future work, but from our experimental data, we are confident of the feasibility of such an approach.

4.2 Features

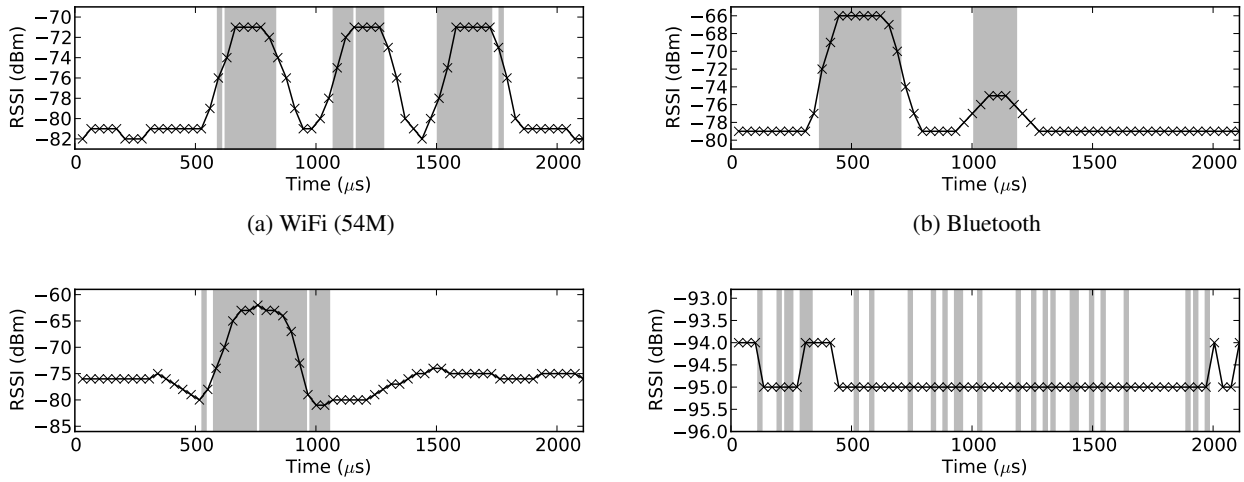
The data we can gather for each corrupted packet is not suitable as direct input to a classification algorithm. Not only is it too large for a classifier run on a resource-constrained sensor node, but the RSSI values are also dependent on the concrete transmission powers and distances of the given scenario. Thus, we define some features that represent the data more concisely and that abstract from transmission power and distance. These features will then be used as inputs to the classifier.

Our approach to feature selection is empirical; we aim to capture patterns that we observe in corrupted packets that we collected in our experiments. Note that the packets shown in Fig. 4 are exemplary, and we display them here to explain the reasoning behind our features—in practice, not all interfered packets show such well-defined patterns. (Readers may skip forward to Fig. 8 for an impression of hard-to-classify packets.)

None of the following features by itself will give a perfect distinction between interferers. Rather, it is the combination of multiple features that facilitates classification. Furthermore, we have evaluated a number of features in the course of this work that turned out to be of little use in classification. For the sake of brevity, we omit their description and concentrate on those features that we use in the evaluation in Sec. 5.

LQI threshold

We define a binary feature that indicates whether the *LQI of a corrupted packet is higher than 90*. As described previously, LQI can be considered to represent the chip error rate over the first two bytes of a packet. If a packet is received with a high LQI, but the packet fails the CRC check, we take this to as an indicator that channel conditions where good when reception started, but then deteriorated. Such a sudden change in channel conditions can be observed when an interferer starts emission during packet reception. In contrast, packets that cannot be decoded correctly due to insufficient signal strength usually have a low LQI value, because the channel conditions are poor over the whole packet reception time. The



(a) WiFi (54M)

(b) Bluetooth

(c) Microwave

(d) Insufficient signal strength

Figure 4: Corruption of some 802.15.4 packets in the presence of different interferers. Dark gray areas show which parts of a packet have been corrupted. The RSSI during packet reception is also shown. Note that the Y axis' range differs between subfigures.

threshold value 90 was chosen empirically from our measurement data.

RSSI-related features

We define a binary feature to indicate whether the *range of RSSI values is greater than 2 dB*. For packets that are corrupted due to insufficient signal strength, the RSSI values often contain little variation, whereas distinct peaks in signal strength can usually be seen for interfered packets.

Before considering further RSSI-related features, we assume that the series of RSSI measurements is normalized by the maximum value in the series. As argued before, absolute RSSI measurements are dependent on distances and transmission powers of sensor nodes and interferers. Normalization allows us to abstract from the concrete distances and transmission powers, while preserving the series' shape. Before normalizing, we smooth the series using a moving average to reduce high-frequency noise which is due to the CC2420's internal quantization of the signal strength. We found a window size of eight to give reasonable filtering while preserving the shapes of interest.

We define the features *mean normalized RSSI* to be the average value of the smoothed, normalized RSSI readings, and similarly, we use the *standard deviation of normalized RSSI*. Our reasoning is that the RSSI series of packets corrupted by a given interferer often have similar, distinct shapes, and these features are a simple means of characterizing the shape of a series of RSSI values. Though obviously two different shapes can have the same mean and standard deviation, we observe distinct distributions for the mean and standard deviation of RSSI for different interferers. This is demonstrated in Fig. 5a, which shows the distribution of standard deviation for different interferers.

Finally, we define a feature in terms of the *difference between the maximum normalized RSSI value and the most common RSSI value*. This feature captures a pattern that can be seen in Fig. 4c and that we commonly observe in packets interfered by microwave ovens: the signal strength slightly drops, then peaks, and drops again. For such packets, this difference is smaller than the difference for packets without this pattern.

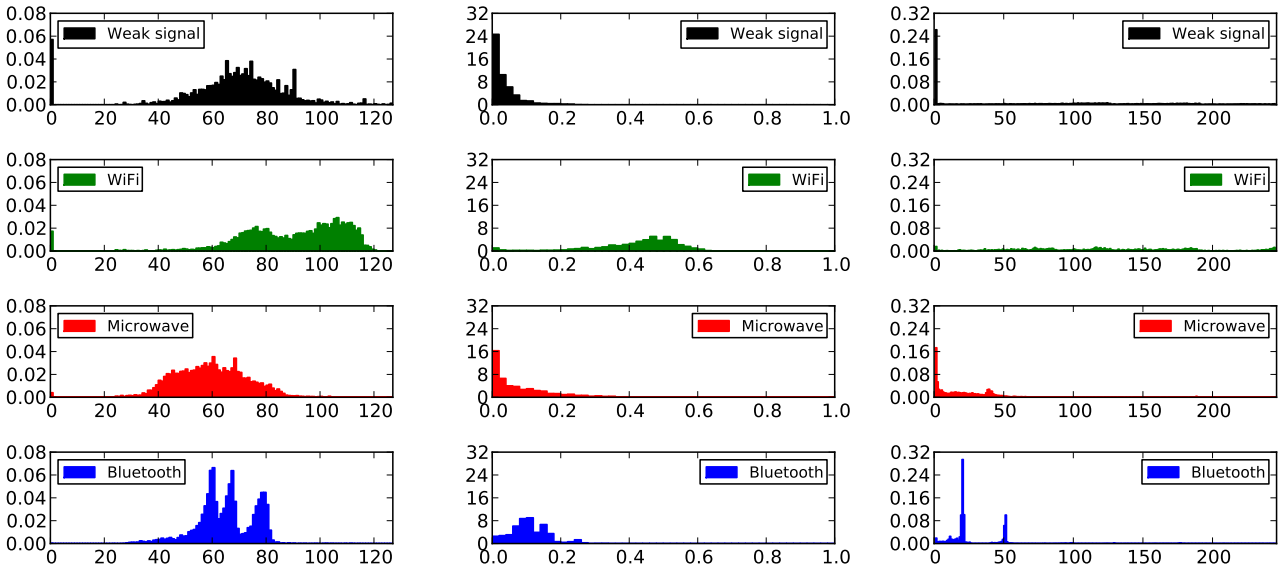
Number of corrupted symbols

Knowing the true payload of a packet, we can count the number of symbols that have been decoded incorrectly. We normalize this number by the total number of symbols in the payload. This feature helps to distinguish different types of interferers. Radio technologies such as Bluetooth or 802.11g specify bit rates, minimum and maximum packet lengths, as well as inter-packet delays for medium access control. These specifications put a constraint on the amount of damage that can be done to a 802.15.4 packet. This can be seen in Fig. 5b, which shows the histogram of corrupted symbols per packet for different interferers. The distributions of WiFi and Bluetooth have distinct means, whereas the distributions for microwave-interfered packets and packets with insufficient signal strength are more similar.

Error bursts

Comparing the received payload to the correct payload, we can define *error bursts* for each corrupted packet. An error burst is a sequence of corrupted symbols that may contain subsequences of at most four consecutive correct symbols. Allowing error bursts to contain correct symbols allows us to detect contiguous areas of interference. For example, we consider the packet in Fig. 4a to contain three error bursts, and the packet in Fig. 4c to contain one error burst.

For WiFi, we often observe regular bursts of specific lengths, exemplified Fig. 4a. We thus define the feature *mean burst length* for each packet. Bluetooth-interfered packets often contain two error bursts of specific length, whereas the pattern for packets that are corrupted due to insufficient signal strength is much more random. To capture these variations, we define the feature *standard deviation of burst length*. For Bluetooth, we often observe that the time between the start of the first burst and the end of the second burst roughly matches the Bluetooth slot length. Hence we define the



(a) Standard deviation of normalized RSSI

(b) Normalized number of corrupted symbols

(c) Bursts spanned

Figure 5: Histograms of selected features of corrupted 802.15.4 packets. One histogram is shown for each group of packets corrupted by one type of interferer.

feature *bursts spanned*, which counts the number of symbols between the first burst's start and the last burst's end. The distribution of this feature is shown in Fig. 5c.

The MAC protocols of WiFi and Bluetooth impose constraints on the time between two packets. Thus, considering the spacing between two error bursts, i.e., the number of symbols correctly received between two error bursts, can give away useful information about the interferer. We define the feature *burst spacing mean* to capture these variations. If a packet contains only one error burst, this feature is undefined. An overview of all features can be found in Tab. 1.

LQI-based	$\text{LQI} < 90$
RSSI series-based	$\max(\text{RSSI}) - \min(\text{RSSI}) > 2\text{dBm}$
	$\text{avg}(\text{RSSI}_{\text{normed}})$
	$\text{stddev}(\text{RSSI}_{\text{normed}})$
	$\max(\text{RSSI}_{\text{normed}}) - \text{mode}(\text{RSSI}_{\text{normed}})$
Error burst-based	Number of corrupted symbols/payload length
	$\text{avg}(\text{burst lengths})$
	$\text{stddev}(\text{burst lengths})$
	End of last burst − start of first burst
	$\text{avg}(\text{burst spacings})$

Table 1: Overview of features used for classification

4.3 Classification Algorithms

We use a supervised learning approach to train a classifier to assign each corrupted packet to a class representing either WiFi, Bluetooth or microwave interference, or corruption due to insufficient signal strength. In supervised learning, a classifier is trained on a set of examples for which the correct class (i.e., the interference source in our case) is given. A corrupted packet is represented by the features described in the previous section. The learning phase, which is computationally more costly than classifying individual packets after training is completed, is carried out on a regular PC, whereas the actual classification is performed online in the sensor network. We consider two different classification algorithms: Support Vector Machines (SVMs, [3]) and feed-forward neural networks [18].

An SVM transforms the feature vectors to a high-dimensional space and, during the learning phase, constructs (potentially non-linear) hyperplanes to separate the vectors from the learning set according to their classes. In the classification phase, the SVM determines the class of the input vector by considering on which side of the hyperplanes the vector lies. Unfortunately, it turned out that online classification in a sensor network with SVMs is not feasible in our case due to the limited amount of RAM available on the sensor nodes. We nevertheless present results for the SVM classification in the evaluation as a reference case, since they are often considered the best "out-of-the-box" classification algorithm [15]. Furthermore, SVMs find a global, unique solution, whereas neural networks may get stuck in local minima during the learning phase.

Feed-forward neural networks are a class of classification algorithms inspired by biological neural networks. They are represented by directed acyclic graphs, where each node represents a computational unit and weighted edges describe input/output relationships between the nodes. Each node represents either an input value (i.e., a component of a feature vector), an output value, or an activation function, which usually is a sigmoid function that takes as input the weighted outputs from incoming edges. During the learning phase, an optimal set of edge weights is found. In the clas-

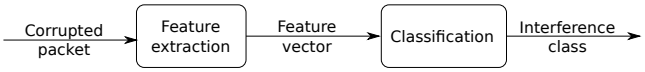


Figure 6: Steps in classification on a sensor node

sification phase, an input feature vector is propagated through the network and the output nodes indicate the classification result. After the costly learning phase, a neural network can be represented by a matrix and the classification cost is dominated by matrix multiplications. Thus, with careful implementation, the use of a feed-forward neural network is feasible even on a resource-constrained platform like TelosB.

4.4 Implementation and Overhead Considerations

We briefly consider some implementation and overhead aspects. Our approach requires two steps shown in Fig. 6: extraction of features from the data gathered for a corrupted packet and evaluation of the feed-forward neural network.

Measuring RSSI during packet reception incurs an additional energy cost, because the sensor node's MCU needs to be awake while the packet is being received. This cost is proportional to the length of the received packet. But since power consumption is usually dominated by radio operation, we believe the additional cost from the MCU being awake to be tolerable. The buffer for RSSI samples is 128 bytes of RAM.

The cost of determining the damaged parts of a packet depends on whether FEC or retransmissions are used. With FEC, there is no additional overhead. If retransmissions are used, RAM needs to be allocated to store a history of recent packets. The size of the history depends on the retransmission protocol and the network's traffic pattern, but we estimate the history to be on the order ten packets, corresponding to 1270 bytes of RAM, given the maximum payload length of 127 bytes imposed by 802.15.4.

Most operations that are necessary for feature extraction are trivial to perform, such as additions or finding the maximum value in a buffer. We note the following regarding smoothing, calculating mean, standard deviation and the most common value: smoothing can be efficiently implemented using an exponential weighted moving average. Calculating the mean and standard deviation of a series each require one iteration over the series of values. Finally, the most common value of a series can be easily found by sorting the series and then iterating over it. Since our series are small (< 128 bytes), the overhead is moderate.

The extensive use of floating point operations in the evaluation of a neural network poses a challenge on constrained platforms that do not support floating point operations. Thus, to make online classification feasible, we have implemented a fixed point neural network that operates on 32-bit integer values and does not require any floating point operations. The C implementation takes approximately 60 lines of code. The memory consumption is less than 1 kilobyte of RAM to represent the weights of the neural network and less than 100 bytes of RAM for temporary variables. As described in the previous section, the weights of the neural network are determined during the neural network's learning phase, which is performed on a PC.

5. EVALUATION

We now investigate classification accuracy to assess how well our features are capable to capture interference patterns, and whether the fixed point implementation of the neural network performs significantly worse than a floating point neural network or SVM.

5.1 Datasets

We constructed two datasets of interfered packets from the measurements described in Sec. 3: a training set for training the classifiers and a testing set for evaluation. The sets are balanced, i.e., each set contains an equal amount of corrupted packets for each interference class and an equal amount of packets corrupted due to insufficient signal strength. The sets are also balanced regarding the set of parameters for each interferer. E.g., each set has an equal amount of packets interfered by the various WiFi rates. For the interfered packets (as opposed to packets with low signal strength), we exclude all such packets from our sets that were received incorrectly by only one of the twelve receivers, since such packets may be corrupted due to transient errors in that receiver rather than due to interference.

For each interference class, we used devices from different vendors for the testing set and the training set. E.g., the training set contains WiFi packets corrupted by devices using an Atheros AR9100 chipset, whereas the testing set contains WiFi packets corrupted by different devices using a Broadcom BCM5354 chipset. This is to ensure that the features we train on are sufficiently general and not model-specific. Furthermore, it also ensures that the training sets and testing sets are disjoint.

The training set consists of packets of 64 and 96 bytes length. To ensure that our features are not specific to packet length, the testing set additionally includes packets of 124 bytes. We exclude shorter packets as they have shown to carry insufficient information for meaningful classification.

5.2 Classification Accuracy

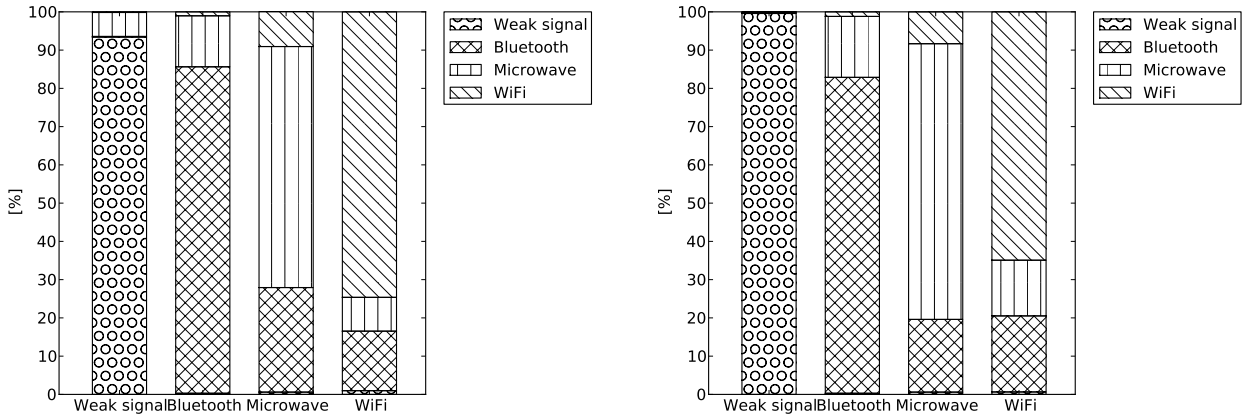
To evaluate how well the features we have selected can distinguish between different interferers and to establish a comparison case for the implementation for resource-constrained devices, we have trained an SVM, a fixed point neural network, and a floating point neural network to classify interferers. The result from the SVM serves as a base case for the neural networks' performance.

Support vector machine

The classification accuracy for the SVM is shown in Fig. 7a. The mean classification accuracy is 79.1%. Packets corrupted by insufficient signal strength are correctly classified to 93.4%. Most of the misclassified weak-signal packets are attributed to microwave interference. The accuracy for Bluetooth and microwave interference is 85.3% and 63.0% respectively. Packets interfered by Bluetooth and microwave share some similarity in the corruption patterns, which shows in the fact that around 27% of microwave interference is incorrectly classified as Bluetooth interference. Regarding WiFi interference, 74.6% of the corrupted packets are correctly classified. Again, WiFi is most likely to be misclassified as Bluetooth interference (15%).

Fixed point neural network implementation

The classification accuracy for the fixed point neural network is shown in Fig. 7b. The mean classification accuracy is 79.6%. The



(a) Classification accuracy for the support vector machine

(b) Classification accuracy for the fixed point neural network

Figure 7: Classification accuracy

classification of non-interfered packets is above 99%, which is significantly better than the SVM. This is an important figure, since mistaking corrupted packets due to insufficient signal strength for actual interference may trigger costly interference mitigation strategies. The fixed point neural network correctly classifies Bluetooth interference in 82.6% of the cases—marginally worse than the SVM. It performs considerably better at classifying microwave interference (72.1% accuracy), but worse at WiFi (64.9%). WiFi is most commonly misclassified as Bluetooth (20.0%) or microwave interference (14.6%).

In total, the performance of the fixed point neural network is similar to the SVM's performance. This is an indication of its feasibility for the classification task at hand, despite the fact that it may theoretically reach non-optimal accuracy due to local minima in the error function.

Floating point neural network

A more resource-demanding floating point neural network implementation performed only marginally better than our fixed point implementation. Just 0.2% of the packets have been classified differently by the floating point neural network compared to the fixed point neural network. Thus, the fixed point implementation does not suffer from a worse accuracy.

5.3 Misclassified Packets

By inspecting the misclassified packet we can understand the shortcomings of our approach. In many cases microwave interference is being classified as Bluetooth interference and vice versa. From manual inspection, we realize that our features do not suffice to distinguish some of them; for an example, see Fig. 8a.

Another set of misclassified packets is due to the fact that in our approach, data is only collected during packet reception. If the transmission of an 802.15.4 packet is only partly interfered, we may have insufficient data to correctly classify the packets. An example of an overlapping transmission is shown in Fig. 8b, in which a 802.15.4 packet is overlapped by a WiFi transmission towards the end.

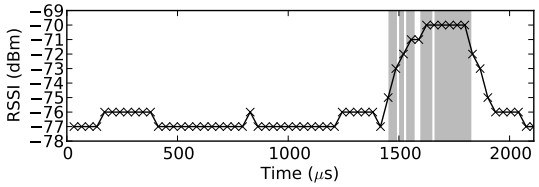
We also observe that some of the interfered packets have only little variation in signal strength and only a few symbols of the payload are corrupted (Fig. 8d, 8c). These packets cannot be classified with our approach, as too little information is available. We did not identify certain sets of parameters (e.g., WiFi bit rate and 802.15.4 channel) for which classification was significantly worse than for other parameter values.

As mentioned earlier, we exclude packets of 16 and 32 byte length from our evaluation. An early evaluation had shown the classification accuracy for packets of 16 bytes and 32 bytes length to be very low (50% and 58% respectively). This is not surprising, as for such packets we get only little data to extract features from.

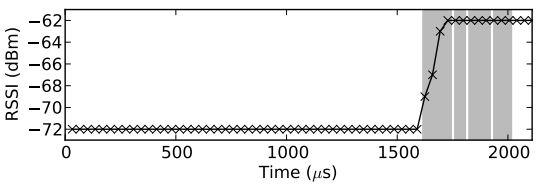
6. RELATED WORK

Relevant research in the WiFi domain includes Airshark [16], a system that uses standard 802.11 cards to sample the spectrum. The sampled data is analyzed using cyclostationary process methods to detect transmission patterns, which are then used to classify interferers. Another example is RFdump [12], which uses a software-defined radio to detect which devices are accessing the medium. RFdump aims to provide a tcpdump-like tool for the wireless communication. Gollakota et al. describe how antenna diversity in 802.11n can be exploited to reconstruct interfered signals [6]. All these approaches have in common that they require advanced signal processing capabilities that are usually not available in sensor networks. Cisco has developed a spectrum analyzer for network analysis that is capable of classifying radio devices [2].

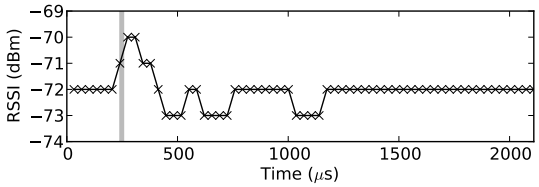
In sensor networks, interference detection can help mitigation, which in turn increases the network lifetime by reducing unsuccessful communication attempts. Chowdhury et al. describe an approach to interference classification by actively scanning channels for characteristic spectrum usage [1]. In contrast to our work, this approach comes at a higher energy cost, because the radio needs to be turned on even when no sensor network communication is ongoing. Their work is also concerned with interference mitigation. Hauer et al. describe how detection of WiFi interference can be used for interference mitigation [8]. Similar to our approach, they also consider RSSI during packet reception for identifying interference. Their



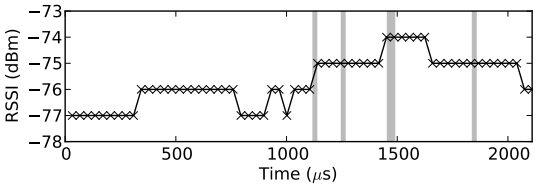
(a) Microwave interference misclassified as Bluetooth interference



(b) WiFi (6M) interference misclassified as microwave interference



(c) Bluetooth interference misclassified as microwave interference



(d) WiFi (5.5M) interference misclassified as microwave interference

Figure 8: Examples of misclassified packets

work is concerned with selectively retransmitting parts of a packet that are suspected to be interfered, without having certain knowledge about what caused the corruption.

A large body of works considers the effect of interference on sensor networks in terms of high-level metrics such as packet reception rate. We selectively refer to [7, 14, 19] for an overview.

7. CONCLUSION AND FUTURE WORK

In this paper, we have investigated the feasibility of a lightweight interference detection and classification approach that only uses data that can be gathered during a sensor network's regular operation. The reason to only use such data is to keep energy consumption as low as possible. We described a set of features that allows us to define the characteristic patterns that we observe for different sources of interference. We also demonstrated that a fixed-point neural network reaches a mean classification accuracy of 79.8% for packets of 64 bytes and more.

For this paper, we have gathered training and testing data in an anechoic chamber, which is a highly controlled environment. While we introduce a certain variance into our data by using different hardware models for the interferers, we would expect a less controlled radio environment to contribute further variance due to multipath propagation. Thus, we also plan to perform experiments in the offices of our university to test the robustness of our features. However, we face the practical issue of establishing ground truth in such an environment—we could not with certainty say that a packet is interfered by the interferer that we have activated, because an RF device outside our control (e.g., the university WiFi) may also affect our experiments.

Interference detection and classification is an important tool for debugging network problems and mitigation strategies, but it is not an end in itself. Therefore, we plan to integrate our approach into an existing interference mitigation strategy. If our approach helps to make significantly better mitigation decisions in an uncontrolled environment, we may avoid the aforementioned issue of establishing ground truth for measuring the performance of our classification approach.

The aim of this work was to assess the feasibility of interference classification with the limited information that can be gathered from corrupted packets. We have focused on classifying the source of interference for individual packets. An interesting track of future work we plan to follow is to incorporate information about previously received, corrupted packets into the classification process. We believe this may yield a significant increase in accuracy. However, incorporating previous information requires careful consideration, especially regarding situations in which multiple interferers are present, or in which the source of interference rapidly changes due to high mobility.

8. REFERENCES

[1] K. Chowdhury and I. Akyildiz. Interferer classification, channel selection and transmission adaptation for wireless sensor networks. In *Proc. of ICC '09.*, pages 1 –5, June 2009.

[2] Cisco Inc. *Cisco Spectrum Expert Wi-Fi*, 2012.

[3] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[4] Crossbow Inc. TelosB datasheet. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Datasheet.pdf`.

[5] P. E. Gawthrop, F. H. Sanders, K. B. Nebbia, and J. J. Sell. Radio spectrum measurements of individual microwave ovens – volume 1 & 2. NTIA Report TR-94-303-1, NTIA Report TR-94-303-2.

[6] S. Gollakota, F. Adib, D. Katabi, and S. Seshan. Clearing the RF smog: making 802.11n robust to cross-technology interference. In *Proc. of SIGCOMM '11*, pages 170–181, 2011.

[7] J.-H. Hauer, V. Handziski, and A. Wolisz. Experimental Study of the Impact of WLAN Interference on IEEE 802.15.4 Body Area Networks. In *Procs. of EWSN '09*, 2009.

[8] J.-H. Hauer, A. Willig, and A. Wolisz. Mitigating the Effects of RF Interference through RSSI-Based Error Recovery. In *Wireless Sensor Networks*, volume 5970 of *Lecture Notes in Computer Science*, pages 224–239. Springer, 2010.

[9] IEEE Computer Society. *Local and metropolitan area networks, specific requirements, part 15.1: Wireless medium*

access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs), 2005.

[10] IEEE Computer Society. *802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)*, 2006.

[11] IEEE Computer Society. *Local and metropolitan area networks, specific requirements, part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2007.

[12] K. Lakshminarayanan, S. Sapra, S. Seshan, and P. Steenkiste. RFDump: An Architecture for Monitoring the Wireless Ether. In *Procs. of CoNEXT '09*, Dec. 2009.

[13] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis. Surviving Wi-Fi Interference in Low Power ZigBee Networks. In *Procs of SenSys '10*, pages 309–322, 2010.

[14] R. Musaloiu-E. and A. Terzis. Minimising the effect of WiFi interference in 802.15.4 wireless sensor networks. *Int. Journal of Sensor Networks*, 3:43–54, 2008.

[15] A. Ng. Support Vector Machines – CS229 Lecture Notes, 2011.

[16] S. Rayanchu, A. Patro, and S. Banerjee. Airshark: detecting non-wifi rf devices using commodity wifi hardware. In *Procs. of IMC '11*, pages 137–154, 2011.

[17] O. Rensfelt, F. Hermans, L.-A. Larzon, and P. Gunningberg. Sensei-UU: a relocatable sensor network testbed. In *Procs. of WiNTECH '10*, pages 63–70, September 2010.

[18] R. Rojas. *Theorie der neuronalen Netze*. Springer, 1993.

[19] A. Sikora. Compatiblity of IEEE 802.15.4 (ZigBee) with IEEE 802.11 (WLAN), Bluetooth and Microwave Ovens in 2.4 GHz ISM-Band. Technical report, University of Cooperative Education Loerrach, 2004.

[20] Texas Instruments Inc. CC2420 - 2.4 GHz IEEE 802.15.4, ZigBee-ready RF Transceiver. `http://www.ti.com/lit/gpn/cc2420`.