SoNIC: Classifying Interference in 802.15.4 Sensor Networks

Frederik Hermans
Uppsala University, Sweden
frederik.hermans@it.uu.se

Olof Rensfelt
Uppsala University, Sweden
olof.rensfelt@it.uu.se

Thiemo Voigt
SICS and Uppsala University
thiemo@sics.se

Edith Ngai
Uppsala University, Sweden
edith.ngai@it.uu.se

Lars-Åke Nordén
Uppsala University, Sweden
lln@it.uu.se

Per Gunningberg
Uppsala University, Sweden
per.gunningberg@it.uu.se

ABSTRACT

Sensor networks that operate in the unlicensed 2.4 GHz frequency band suffer cross-technology radio interference from a variety of devices, e.g., Bluetooth headsets, laptops using WiFi, or microwave ovens. Such interference has been shown to significantly degrade network performance. We present SoNIC, a system that enables resource-limited sensor nodes to detect the type of interference they are exposed to and select an appropriate mitigation strategy. The key insight underlying SoNIC is that different interferers disrupt individual 802.15.4 packets in characteristic ways that can be detected by sensor nodes. In contrast to existing approaches to interference detection, SoNIC does not rely on active spectrum sampling or additional hardware, making it lightweight and energy-efficient.

In an office environment with multiple interferers, a sensor node running SoNIC correctly detects the predominant interferer 87% of the time. To show how sensor networks can benefit from SoNIC, we add it to a mobile sink application to improve the application's packet reception ratio under interference.

Categories and Subject Descriptors

B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance.

Keywords

Interference Classification, Wireless Sensor Networks, SoNIC, Decision Tree, Mobile Sink

IPSN'13, April 8–11, 2013, Philadelphia, Pennsylvania, USA.

1. INTRODUCTION

Due to a rapid increase in the number of technologies and devices operating in the license-free 2.4 GHz band, radio interference becomes an increasing problem for low-power wireless sensor networks. It has been shown that interference from other devices reduces sensor network performance, as it causes packet loss, reduces throughput, increases delay, and drains the sensor nodes' limited energy reserves [3, 24, 25].

The problem is exacerbated by the diversity of technologies that share access to the 2.4 GHz band. At any given time, a sensor network may have to compete with WiFi devices, Bluetooth peripherials, microwave ovens, baby monitors, or car alarms. All these technologies differ widely in when they access the spectrum, what frequencies they use, and for how long they use them [2]. It has been shown that when the interference source is known, using a specialized mitigation approach can improve performance [4, 15]. For example, the BuzzBuzz protocol [15] improves the delivery rate in sensor networks exposed to WiFi interference.

We address the problem of detecting and classifying interference within a 802.15.4 sensor network, so that the network can select a suitable mitigation strategy. Existing approaches to interference detection rely on sampling the spectrum continuously and over long durations for patterns in signal strength that are characteristic of certain interferers [1, 4, 14, 26]. Such active sampling is often prohibitively costly for battery-powered sensor nodes, where a radio in receive mode is commonly the most energy-hungry component. Some of the approaches require nodes to tune into different 802.15.4 channels to gauge the spectral footprint of an interferer [1]. This channel hopping makes it complicated for the sensor network to continue operation during interference detection. Furthermore, some approaches depend on additional hardware such as software-defined radios [14] or PC-class computers for processing [1, 26].

We present the Sensor Network Interference Classification (SoNIC) system, which takes a novel path to interference detection. Rather than actively sampling the spectrum, a node using SoNIC detects interferers by considering individual corrupted 802.15.4 packets, i.e., packets that the node has received, but for which the received payload did not match the packet's checksum. Through extensive measurements, we establish that different interferers corrupt individual 802.15.4 packets in distinct patterns, thereby leaving a "fingerprint" on the packet. The interferer's fingerprint

becomes visible in *(i)* how the signal strength varies during packet reception, *(ii)* in the link quality indication (LQI) associated with the packet, and *(iii)* which bytes of the payload are corrupted. SoNIC exploits retransmissions to identify the corrupted parts of a packet. By solely relying on corrupted packets from regular sensor network traffic, SoNIC does not incur additional communication costs.

To demonstrate the feasibility of our approach, we have implemented SoNIC for the TelosB platform. We define a set of simple features that characterize the different fingerprints, and use a supervised learning approach to create a classifier that classifies corrupted packets. We assess the performance of SoNIC in both a controlled radio environment and an uncontrolled environment. Our results show that SoNIC reaches a mean accuracy of 72% when classifying individual corrupted packets. In a challenging office environment with multiple interferers and under mobility, SoNIC correctly detects the predominant interference source 87% of the time by considering a set of corrupted packets. We also show how a mobile sink application can be augmented with SoNIC and two mitigation techniques to increase the application's network performance under interference.

In summary, our three key contributions are:

- We establish that different interferers corrupt individual 802.15.4 packets in unique patterns, and show that these patterns prevail in different radio environments. We show that by using these patterns, a classifier can attribute corrupted packets to an interference type.
- We demonstrate that the classification is feasible on resource-constrained sensor nodes at moderate computational overhead, while maintaining a high classification accuracy. We show that SoNIC correctly detects the predominant interferer in a challenging radio environment with multiple interferers.
- We show how sensor networks can benefit from SoNIC by augmenting a mobile sink application with the system. With SoNIC, the application's packet reception rate under heavy interference is improved from 45% to 61%.

The rest of the paper is organized as follows. We develop the approach of classifying corrupted packets in Sec. 2. In Sec. 3, we describe how the approach can be implemented in a system for resource-constrained sensor nodes. The system's overhead and its performance in a controlled environment are considered in Sec. 4. We evaluate SoNIC in an uncontrolled radio environment in Sec. 5, and show how a mobile sink application benefits from SoNIC, using straightforward WiFi and microwave mitigations strategies in Sec. 6. Related work is described in Sec. 7. After a brief discussion on new and multiple interferers, we conclude the paper in Sec. 9.

2. CLASSIFYING CORRUPTED PACKETS

In this section, we develop the underlying idea of our approach: classifying corrupted 802.15.4 packets according to the source of interference. We begin by describing a set of measurements in which we systematically expose a sensor network to different interferers.

2.1 Characterizing Interference

To study the effect of interference on individual 802.15.4 packets, we set up thirteen TelosB sensor nodes alongside

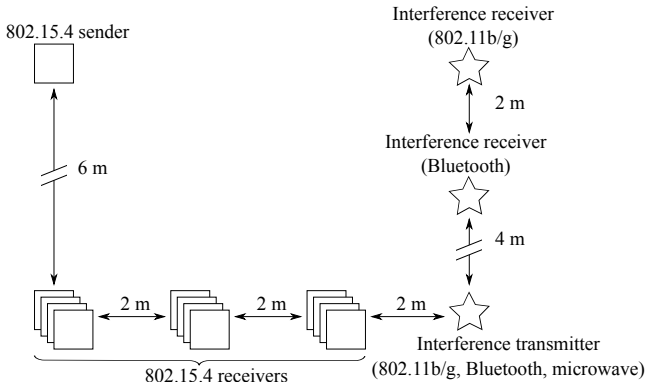


Figure 1: Experimental setup in the anechoic chamber

different interferers in an anechoic chamber (Fig. 1). The TelosB nodes are referred to as 802.15.4 sender and receivers in the figure, respectively. In each experiment, the 802.15.4 sender periodically broadcasts packets while one of the interferers is active. Three groups of four receivers are placed at varying distances from the sender and interferer. The anechoic chamber is shielded from outside radio emissions, so we are confident that corruption in 802.15.4 packets is caused by emissions from the active interferer. We focus on three sources of interference that are prevalent in indoor environments.

WiFi. We use two 802.11b/g routers to create WiFi interference. One of the routers acts as an access point streaming constant bitrate UDP traffic to the other router, which acts as a client. The resulting traffic resembles a video streaming session. We repeat the experiments for all 802.11b/g bitrates and modulations.

Microwave ovens. We heat a bowl of water in a residential-type microwave oven to create interference typical of these devices.

Bluetooth. To create Bluetooth interference, a Bluetooth dongle sends back-to-back packets to another dongle. Since Bluetooth uses adaptive frequency hopping, we reset the dongles before each experiment.

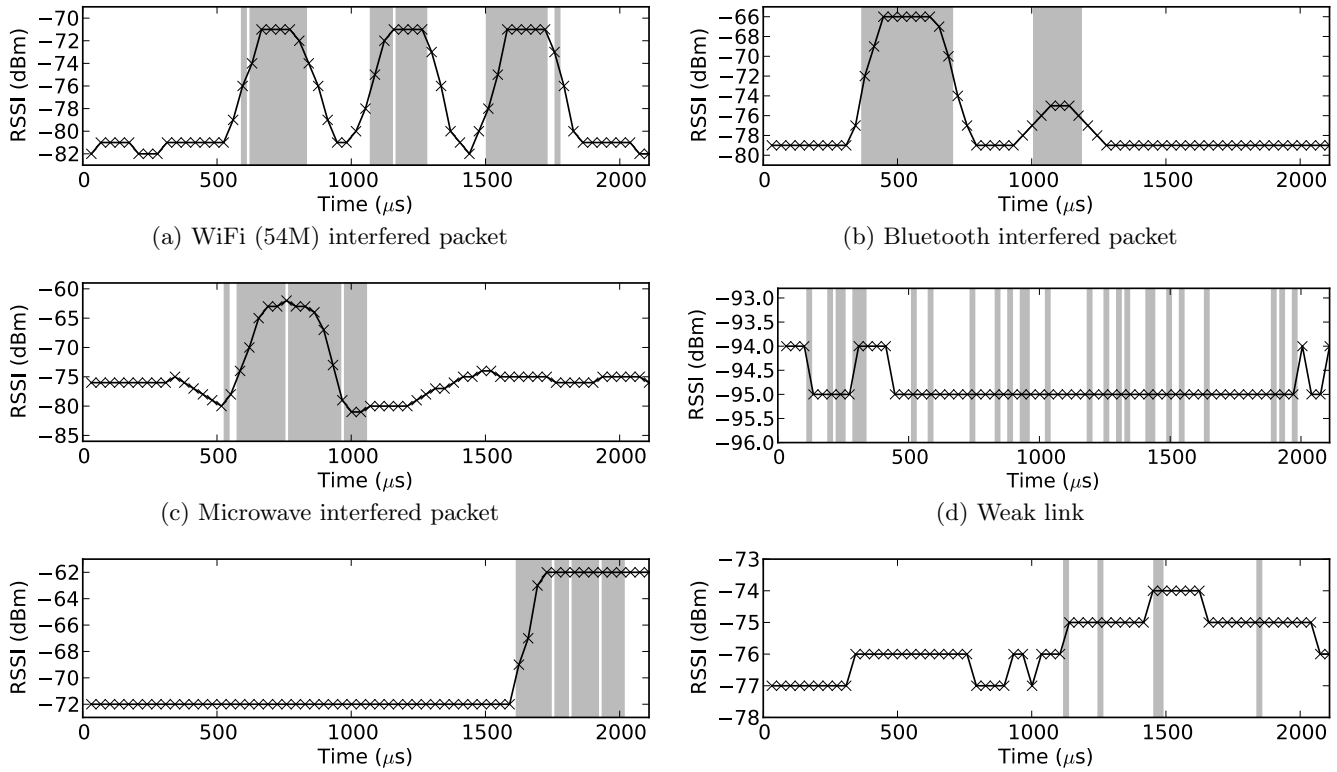
Non-interfered weak links. We also perform an experiment without interference. We set the sender's transmission power so that the signal strength at the receivers is close to the sensitivity threshold, causing corruption and packet loss. These experiments serve as a reference case for packet loss not caused by interference.

All experiments are repeated with hardware from different vendors, i.e., different models of WiFi routers, Bluetooth dongles, and microwave ovens. We also use a range of 802.15.4 channels to vary the frequency offset between interferer and sensor network. The packet length is varied from 16 to 124 bytes.

2.2 Features of Corrupted Packets

The receiving nodes logged about 900,000 *corrupted packets* during our experiments. These are packets that were received, but for which the received payload did not match the packet's checksum. Note that we distinguish corrupted packets from *lost packets*, for which the receiver failed to detect the packet preamble and hence was unaware of the transmission attempt.

For each corrupted packet, the receiver logs the signal



(a) WiFi (54M) interfered packet

(b) Bluetooth interfered packet

(c) Microwave interfered packet

(d) Weak link

(e) WiFi (6M) interfered packet, partially overlapped

(f) WiFi (5.5M) interfered packet, little corruption

Figure 2: Corruption and signal strength for some 802.15.4 packets in the presence of different interferers. SoNIC classifies corrupted packet based on the variations in signal strength and payload corruption.

strength during packet reception, the LQI associated with the packet, and the received (corrupt) payload. LQI is a measure of link quality that an 802.15.4-compatible transceiver provides for each received packet. In case of the CC2420 that is used on TelosB and many other sensor node platforms, LQI can be understood as the inverse of the chip error rate of the packet's preamble. To obtain the signal strength during packet reception, we modify the radio driver in Contiki. Our modified driver starts sampling RSSI whenever an interrupt signals an incoming packet, and it keeps sampling at a rate of one sample/byte until the last byte of the packet is received. Finally, since we know the correct packet payload from our log data, we determine which symbols of the received payload suffered transmission errors[1].

Fig. 2 shows examples of corrupted 802.15.4 packets that were received in the presence of different interferers. Each subfigure represents one 64-byte long 802.15.4 packet. The shaded areas indicate incorrectly decoded symbols in the packet payload, whereas the non-shaded area indicate correctly received symbols. The line shows the signal strength at the receiver during packet reception. A valid packet would have a mostly straight RSSI line with occasional quantization noise. The top three subfigures each represent one packet that was corrupted by WiFi, Bluetooth, and microwave interference respectively. The corruption coincides with distinct peaks in signal strength, caused by emissions from the interferer. Note that the packets are quite distinct in the amount and location of corruption and in the shape of the RSSI series. We exploit these kinds of regularities to classify corrupted packets. For comparison, Fig. 2d shows a packet that was not interfered, but that was received over a weak link with a signal strength close to the sensitivity threshold. Note the low RSSI compared to the other figures. This packet was also corrupted, but the corrupted symbols are spread much more evenly over the whole packet, and the low variation in signal strength is an effect of the transceiver's quantization of RSSI values rather than external emissions.

We attempt to capture the regularities that are exemplified in the packets shown in Fig. 2a–2d to create a classifier. However, we do not want and cannot aim to achieve perfect classification accuracy. For example, the packet in Fig. 2e was partially overlapped by a WiFi transmission at 6 MBit/s. Due to the short overlap, little information is available for classifying such packets. Similarly, the packet in Fig. 2f has only very few corrupted symbols and shows little variation in signal strength. Thus, we aim to achieve a reasonable classification accuracy per packet, so that we can infer the presence of an interferer from a set of corrupted packets.

We now describe a set of simple features that can be calculated for each corrupted packet. Based on these features, we train a classification algorithm to distinguish packets that

[1]In 802.15.4, one byte is represented by two four-bit symbols, which are transmitted as 32-bit chip sequences [11]. Decoding of an incoming chip sequence to a symbol is commonly performed in hardware. Thus, the finest granularity at which we can detect transmission errors is one symbol.

Feature	Purpose
LQI > 90	Detect sudden changes in channel conditions caused by interferers
range(RSSI)>2 dB	Distinguish interference from weak links
Mean error burst spacing	Capture temporal behavior of interferers, esp. for Bluetooth, WiFi
Error burst spanning	
Mean normalized RSSI	Characterize RSSI series, esp. for microwave
1 - mode($RSSI_{normed}$)	

Table 1: SoNIC uses six features that in combination enable classification of corrupted packets.

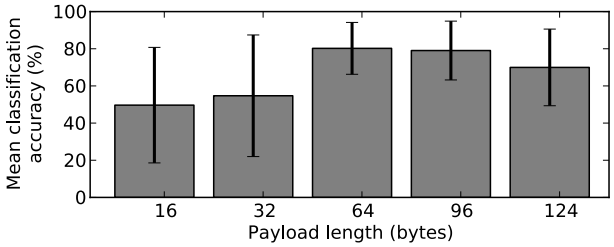


Figure 3: A mean classification accuracy larger than 70% is reached for packets of 64 bytes and more, whereas short packets are not classified accurately mostly due to insufficient overlap with the interferer's emissions.

have been corrupted by WiFi, Bluetooth, microwave ovens or weak links. The features are summarized in Tab. 1.

LQI threshold. If a packet has a high LQI but is corrupted, this is an indication that channel conditions were good when reception started, but then deteriorated. We observe such sudden changes in channel conditions when an interferer starts emitting during packet reception. In contrast, packets that are corrupted due to a weak link usually have a low LQI because channel conditions are equally poor over the whole packet reception time. To help distinguish interference from weak links, we define a binary feature that indicates whether the LQI of a corrupted packet is greater than 90. This threshold value is chosen empirically.

Error bursts. We define an error burst in a packet to be a sequence of corrupted symbols that may contain subsequences of at most four consecutive correct symbols. For example, we consider the packet in Fig. 2a to contain three error bursts.

MAC and PHY layers commonly impose strict timing constraints on transmissions. For example, Bluetooth devices stay on one frequency for a duration of 625 μs, and WiFi mandates strict inter-frame spacings. These constraints become visible in the error bursts. Rather than trying to detect specific timings, we define two general features to characterize the temporal behavior of different interferers. One is the mean number of symbols between error bursts in a packet, which depends on the interferer's inter-frame spacings; the other feature is the number of symbols between the first burst's start and the last burst's end, which depends on the interferer's transmission duration and frame spacing.

RSSI-based features. We define another binary feature to indicate whether the range of RSSI values associated with a packet is greater than 2 dB. Packets that are corrupted due to interference often show distinct peaks in RSSI, whereas packets received on a weak link contain little variation in signal strength. Before considering further features, note that RSSI depends on transmission power and the distances between receiver, sender, and interferer. To avoid these dependencies while preserving the series' shape, we normalize each series. We also reduce quantization noise by smoothing the series with a moving average filter.

We define the arithmetic mean of the normalized RSSI series to be a feature. We observe distinct distributions for the mean for packets corrupted by different interferers. This is because the RSSI series for packets corrupted by a given interferer often have similar, distinct shapes, and the mean is one way to characterize the shape, albeit a very coarse one. Next, we define a feature to capture a pattern that we commonly observe in packets interfered by microwave ovens: the signal strength slightly drops, then peaks, and drops again (Fig. 2c). We attribute this pattern to the operation of the CC2420's gain control, as observed earlier [2]. We define the feature as $1 - mode$, where $mode$ is the most common value of the normalized series. A series which exhibits the pattern has a low value for the feature.

We have selected the above described feature set from the features we considered during our research, because the set is small, and the features are simple to compute and give a reasonable classification accuracy, as we will show next.

2.3 Feasibility of Classification

We use a supervised learning approach to create and evaluate a classifier. The classifier assigns a corrupted packet to one of four classes: interfered by WiFi, microwave oven, Bluetooth, or corrupted due to a weak link. We create a training and a testing set from the packets collected in the experiments. To ensure that the classifier is not biased to specific nodes, the testing set contains packets corrupted by one set of nodes, and the training set contains packets corrupted by another set from different vendors. We also balance the testing set with respect to the classes and the experiment parameters (e.g., frequency offset, packet sizes, WiFi rates). The testing set contains packets of 16, 32, 64, 96 and 124 bytes. We restrict the training set to packet sizes of 64 and 96 bytes, so we can test whether the classifier can classify packets of length that it has not been trained for.

Based on our datasets, we create a support vector machine (SVM) classifier on a PC using libsvm[2]. SVMs are considered to be the best "out-of-the-box" classifiers [17], and thus provide a good benchmark for testing the usefulness of our features for interference classification.

Fig. 3 shows the mean classification accuracy averaged over all classes for different packet sizes, with the error bars indicating variance between classes. While packets of 64, 96, and 124 bytes are correctly classified with an accuracy of 80%, 79% and 70% respectively, short packets have much lower classification accuracy. This is because short packets overlap only partially with the interferer's emission, and thus often do not carry enough information for meaningful classification. The accuracy does not significantly improve when we include short packets in the training set. We conclude that our classification approach requires packets of 64 bytes or more.

The classification accuracy for packets of 64 bytes and more is shown in Tab. 2. The row labels denote the (true) interferer, whereas the column labels denote the classification

[2] `www.csie.ntu.edu.tw/~cjlin/libsvm/`

Interferer	Classification result: Bluetooth	microwave	weak link	WiFi
Bluetooth	**87.8%**	11.5%	0.4%	0.3%
microwave	29.5%	**60.2%**	0.8%	9.5%
weak link	0.1%	3.2%	**96.1%**	0.7%
WiFi	16.4%	20.9%	1.2%	**61.5%**

Table 2: Classification results for 64 bytes and longer packets. The diagonal shows the classification accuracy.

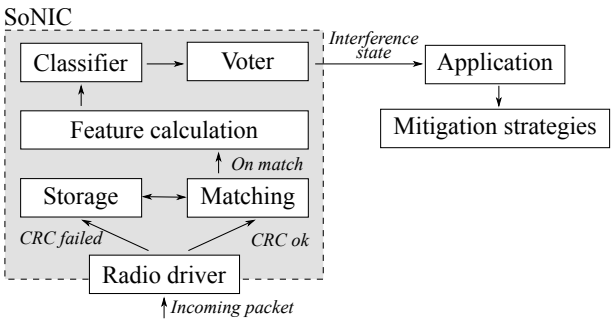


Figure 4: Overview of the SoNIC system

result. The diagonal indicates the percentage of correctly classified packets. Our approach performs best for Bluetooth interference and weak links. At around 60%, accuracy is slightly lower for WiFi- and microwave-interfered packets, but it is still vastly in excess of random chance (25%). Misclassifications occur for packets similar to those shown in Fig. 2e, 2f. We verified that accuracy is similar over all sets of parameter combinations, e.g., WiFi rates and channel offsets. No parameter combination stands out as being extremely hard to classify. The above results demonstrate that our features can effectively characterize the source of interference for a corrupted packet.

3. THE SONIC SYSTEM

We implement the classification approach that we laid out in the previous section in SoNIC, a system that enables a resource-limited sensor node to detect interference sources. Our implementation is based on Contiki [6]. An overview of SoNIC is shown in Fig. 4. The system is comprised of a modified *radio driver* that samples RSSI during packet reception and then hands off the received packet to the *storage module* (if the packet is corrupt) or to the *matching module* (if the packet is valid). The matching module identifies corrupted symbols in the stored packets, and then feeds them to the *feature calculation* module. This module calculates the features described in the previous section and inputs them to the *classifier*. Classification results are collected by the *voter*, which infers an interference state that it indicates to the application. We briefly consider the respective modules of SoNIC.

3.1 Radio Driver

We modify the CC2420 radio driver in Contiki to sample RSSI during packet reception. Sampling starts when an interrupt indicates an incoming packet and ends when the last byte of the packet has been received. No specific functionality is needed to obtain corrupted packets. The radio chip merely indicates that a packet is corrupted (i.e., failed the CRC), rather than discarding it, and thus the corrupted packet is readily available to the node. If a corrupted packet is received, its payload, the sampled RSSI values, and LQI are saved in the storage module. A correct packet is passed up the stack and then handed over to the matching module.

3.2 Storing and Matching Packets

To identify corrupted symbols, SoNIC exploits retransmissions, which are used in many sensor network protocols such as the Collection Tree Protocol [7]. SoNIC stores corrupted packets in a FIFO buffer. When a valid packet is received, the matching module compares the payload of the valid packet against all payloads stored in the buffer. Each corrupted packet that is sufficiently similar to the received, valid packet is passed to the feature calculation module.

Note that if a corrupted packet is accidentally matched to the wrong packet, i.e., a valid packet that is not a retransmission of the corrupted packet, error bursts cannot be correctly identified. To minimize accidental matching, we use a matching algorithm that requires two payloads to share large contiguous regions to be considered a match.

3.3 Feature Calculation

The feature calculation takes as input a valid and a corrupted payload of a packet, and the RSSI values and LQI of the corrupted packet. It uses these inputs to calculate the features summarized in Tab. 1. Testing whether the LQI exceeds 90 and whether the range of RSSI values exceeds 2 dB is straight-forward. Error bursts are identified by a symbol-wise comparison of the valid to the corrupt payload. After the comparison, the mean burst spacing and the spanning of the bursts can be easily calculated. To avoid costly floating-point operations, we use fixed-point arithmetics to normalize the RSSI values and calculate their mean. By using fixed-point arithmetics, the normalization and mean calculation is reduced to a series of computationally cheap integer operations. To determine the most common RSSI value (referred to as the mode in Tab. 1), we sort the RSSI values and count element occurrences.

3.4 Classification

The features are fed into a classification algorithm. The SVM classifier that we have used to asses the feasibility of our features makes extensive use of floating-point operations. We therefore deem it unsuitable for implementing SoNIC on a severely limited platform. Instead, we use a decision tree classifier [5]. Classifying with a decision tree requires traversing the tree and comparing feature values against values stored at the tree's nodes. Thus, classification consists of a series of computationally cheap tests for inequality. We use the C4.5 algorithm [5] on a PC to create a decision tree from the training set collected in the anechoic chamber. The resulting tree consists of 365 inner nodes and 366 leaves.

3.5 Voting

From our investigation of feasibility in Sec. 2.3, we expect the classifier to occasionally misclassify a packet. To tolerate such errors, we use a voting mechanism. The voter considers recently received, corrupted packets taken from a configurable time window. The most common class of packets in the window is indicated as the interference state to the application. In our experiments, we use a window length of

30 s, which we found to give a good trade-off between fast interference detection and high confidence in the voting result.

During our initial experiments in an uncontrolled environment, it emerged that packets with very few corrupted symbols that were classified as active interference (as opposed to a weak link) negatively affected voting performance. We believe this to be an effect of overtraining in the anechoic chamber for packets with little corruption. Indeed, if only very few symbols are corrupted, there is little basis for an informed classification. Therefore, the voter discards packets with less than eight corrupted symbols that are classified as interfered. In the case of the testing set, 13.8% of the packets would be discarded after classification.

3.6 Using and Extending SoNIC

SoNIC reports the interference state to the application. It intentionally does not mandate what mitigation strategy to use, or when to activate and deactivate it. The reason is that the cost of interference, as well as the cost of mitigation, can only be estimated with the requirements of the specific application in mind.

SoNIC's classifier distinguishes between WiFi, Bluetooth and microwave oven interference, and packets that are corrupted due to low TX power. To add detection capabilities for a new interference type, suitable features must be defined and the classifier needs to be retrained.

4. MICROBENCHMARKS

We begin our evaluation by considering the feasibility of classification on resource-constrained sensor nodes. To this end, we first assess the cost of feature calculation and classification, and then evaluate the classification accuracy of the decision tree classifier.

4.1 Memory and Computational Overhead

SoNIC's memory requirements are dominated by the need to store the decision tree in the sensor node's RAM, which requires 1.8 KB. If an application has very tight memory requirements, the C4.5 algorithm can be configured to produce a smaller tree at the cost of classification accuracy. In its default configuration, SoNIC uses 1 KB to store corrupted packets in the FIFO buffer for later matching. This buffer can be adjusted if an application uses large packets. Furthermore, another static buffer of 128 bytes is used to store valid packets, so they can be matched after they have been processed by the network stack. In summary, SoNIC's static memory consumption amounts to ca. 3 KB. We consider this an acceptable memory overhead, and thus have not tried to further optimize memory consumption.

The computational overhead of SoNIC is comprised of feature calculation and classification. We select 1000 packets at random from the testing set and measure the time it takes to calculate features and classify them on a TelosB node. The mean feature calculation time of 26.5 ms (σ = 7.0 ms) is dominated by normalizing the RSSI values, which accounts for about 60% of the total calculation time, because it requires repeated 32-bit integer divisions. The variance is due to differences in packet length. One classification takes 1.2 ms on average (σ = 0.5 ms). Note that newer sensor nodes feature a faster CPU that executes typical operations more than 10 times faster than a TelosB, even when the CPU does not run at full speed [13]. On a modern Cortex-M3-based platform, the time required for feature calculation and classification would hence be lower than the transmission time of a medium-sized 802.15.4 packet. Together with the energy needed to keep the MCU on during packet reception to sample the RSSI, the feature calculation is the only additional energy cost for interference classification with SoNIC.

		Classification result			
		Bluetooth	microwave	weak link	WiFi
Interferer	Bluetooth	**68.1%**	23.9%	0.3%	7.6%
	microwave	25.3%	**63.9%**	0.7%	10.1%
	weak link	0.1%	0.8%	**98.5%**	0.6%
	WiFi	14.3%	24.1%	1.1%	**60.4%**

Table 3: The decision tree achieves a mean classification accuracy of 72.8%, almost on par with more powerful support vector machines (76.4%).

4.2 Accuracy of Decision Tree Classifier

SoNIC uses a decision tree on the sensor nodes due to its low computational complexity. In contrast to SVMs, a decision tree does not perform feature space transformation and separates classes linearly[3]. To test whether the decision tree significantly degrades classification accuracy, we evaluate it on a TelosB sensor node using the data set from the anechoic chamber. The results are summarized in Tab. 3. The mean classification accuracy is 72.8%, which is close to the accuracy of the SVM (76.4%). The decision tree performs slightly better for packets corrupted by microwave, WiFi, or weak links, but performs worse for packets corrupted by Bluetooth (87.8% vs. 68.1%). Most of the misclassified Bluetooth-interfered packets are attributed to microwave. We assume that while the SVM finds a good non-linear separation for the classes Bluetooth and microwave, the C4.5 algorithm that we use to create the decision tree avoids this separation because it aims to build a tree with few nodes. Although a better classification accuracy may be achieved by configuring C4.5 to create a larger tree, we will show that the accuracy we observe in Tab. 3 is sufficient to detect Bluetooth interference.

We conclude that classifying corrupted packets according to their source of interference is feasible even on resource-constrained sensor nodes when using a lightweight classification approach.

5. EVALUATION

So far we considered interference in an anechoic chamber, an idealized environment that is shielded from external radio emissions and that minimizes multipath propagation. We now address classifying interference in a more representative environment: an office corridor with heavy multipath effects and multiple interferers.

5.1 Experimental Settings

The layout of our testbed is shown in Fig. 5. There are six stationary sensor nodes and one mobile sensor node, all TelosB. We use a testbed infrastructure that includes robots for sensor mobility and provides repeatable movement patterns [21]. The mobile node's path follows the corridor and

[3]Strictly speaking, an SVM also only uses linear hyperplanes for separation of classes, but using a transformation allows for non-linear separations in feature space.

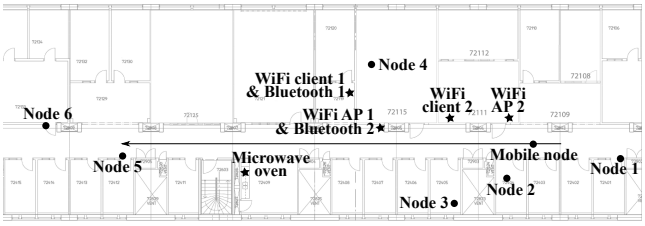


Figure 5: Experiment layout in the office corridor. We omit the location of interferers that we do not control, such as Bluetooth keyboards and university WiFi.

is 32 m long. By using a mobile sensor node, we test SoNIC over a variety of distances between interferers, senders and receivers. Thereby we can evaluate whether our approach is coupled to the specific setup in the anechoic chamber.

The testbed includes two WiFi access points and clients, as well as two Bluetooth dongles, and a microwave oven (stars in Fig. 5). We use these devices to create interference during our experiments. However, there are a number of radio devices in the corridor that we cannot control. The university's WiFi is present on all 802.11g channels. Bluetooth mice and keyboards, which periodically beacon their presence even when not actively used, can be found in many offices. Furthermore, we confirmed with a spectrum analyzer that emissions from microwave ovens located on adjacent floors reach our floor. Thus, the sensor nodes are exposed to varying and uncontrolled interferers during all experiments. We deliberately do not aim to control these interferers because we want to understand how robust our approach is in the presence of such background interference.

5.2 Link Conditions under Interference

We first consider the effect of interference in the corridor. Our aim is to quantify the performance impact and to demonstrate that a considerable fraction of packets is received as corrupted packets.

Methodology. We focus on the link from node 1 to the mobile node. Node 1 sends 64-byte long packets with a period of 125 ms to the mobile node, which travels away from the sender at a speed of 0.12 $\frac{\text{m}}{\text{s}}$ along the arrow shown in Fig. 5. That means that one packet is sent every 1.5 cm. We first perform two experiments without controlled interference, in which the sender's TX power is set to 0 dBm (maximum), and -7 dBm respectively (weak link case). We then perform three experiments under interference from WiFi AP 1 and client 1, the Bluetooth devices, and the microwave oven respectively. The sender uses maximum TX power in these experiments. Our metrics are packet error ratio (PER = 1−PRR) and packet corruption ratio (PCR), which we define as the ratio of received corrupted packets to sent packets. This means that PER includes both lost and corrupted packets, whereas PCR only includes corrupted packets. Results are averaged over ten runs for each experiment.

Results. When none of our interferers is active and the sender uses maximum TX power, PER is stable and varies between 0% and 15%, as we expect for an indoor environment. PER and PCR for the weak link (low TX power) are shown in Fig. 6a as a function of location of the receiving mobile node. The gray shaded areas denote one standard deviation calculated over the experiment repetitions. We attribute the variance primarily to background RF activity. There is a strong increase in PER at 10–24 m, which we attribute to multipath effects in that area (which features many reflective obstacles) causing a notable drop in signal-to-noise ratio (SNR). After 24 m, SNR recovers and PER drops, probably due to the corridor acting as a wave guide.

PER and PCR under interference from different sources are shown in Figs. 6b–6d. The peaks in packet loss roughly correspond to the interferers' locations, which is an effect of the near-far problem [20]. The interferers incur different amount of packet loss due their different channel utilization. WiFi causes the highest packet loss, since its traffic saturates the channel. By principle of operation, the microwave has a "channel utilization" of 50% [2], which corresponds closely to the peak PER observed in Fig. 6c. The microwave has the strongest signal, and hence affects packets at the start and end of the track. Bluetooth interference incurs the least packet loss, because it only briefly overlaps with the sensor network channel due its frequency hopping.

SoNIC requires corrupted packets for classification. Thus, an important observation from Figs. 6a–6d is that regardless of the cause of packet loss, the ratio of corrupted packets is between 10%-20% when the link suffers significant packet loss, i.e., when PER is above 20%.

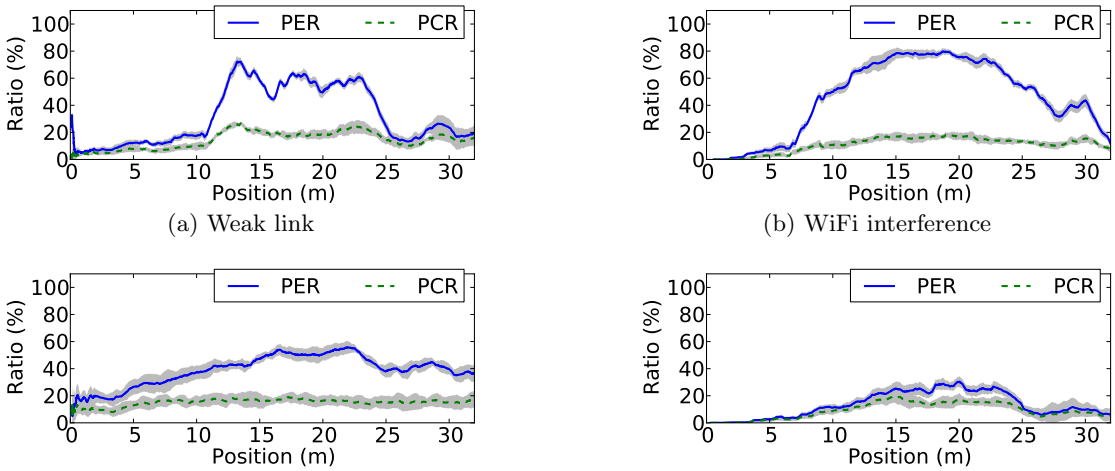
5.3 Classification Results

Next, we consider the classification results from the measurements we just described. Our aim is to evaluate how well the classifier, which has been trained on data from the anechoic chamber, can classify corrupted packets in a realistic radio environment with background RF activity. Note that because the experiments were performed in a live radio environment that we cannot fully control, we do not have the ground truth on whether a given corrupted packet was indeed corrupted by the interferer that we have activated during the experiment. Therefore, we estimate the classification accuracy by statistical means instead. We consider all classifications done during the last 30 s, i.e., all classifications in the voting window. This decision is based on the insight that the dominant interferer causes a significant amount of packet corruption and that a sliding window approach will reduce overreaction to sporadic misclassifications, but that it will on the other hand slow down the reaction time.

Results. We consider the distribution of classes in the voting window at each location of the receiver, i.e., we determine the fraction of packets in the window that are classified to each of the four classes. For example, if the window contains six Bluetooth-, one WiFi-, one microwave-, and two weak-link-classifications, the fractions are 0.6, 0.1, 0.1, and 0.2, respectively.

Fig. 7 shows the distribution of classes at each receiver location, averaged over ten runs. Areas of heavy packet loss (PER > 20%) are marked in gray. Each subfigure shows results for experiments under one type of interference. Each line represents the fraction of packets that have been classified to a given interferer. The most important observation from these figures is that under packet loss, most packets are classified to the interferer that was activated during the experiment. That is, on average SoNIC correctly attributes most of the packets to the dominant interference source.

In the case of low TX power and no controlled interference (see Fig. 7a), most packets are attributed to a weak link during heavy packet loss (11 m–25 m, 28 m–30 m). The accuracy is slightly poorer than the one in the anechoic



(a) Weak link (b) WiFi interference

(c) Microwave interference (d) Bluetooth interference

Figure 6: Packet error ratio and ratio of corrupted packets in the office corridor. A substantial fraction of packets is corrupted, which is necessary for classification.

chamber, which we attribute to uncontrolled radio devices. Most corrupted packets between 0 m and 11 m are attributed to WiFi and Bluetooth interference. This is expected, because the corridor contains a number of WiFi and Bluetooth devices that we do not control.

Fig. 7b shows results for experiments under WiFi interference. As the mobile receiver gets closer to the interferers, the number of packets classified as WiFi increases, until it peaks at the interferers' location at around 15 m. This dependency on distance can be explained as follows. We observed that the amount of corrupted symbols per packet strongly depends on distance in the case of WiFi interference. As the receiver gets closer, it can detect the error bursts in the packets more clearly, and thus the error-burst features take values closer to the values observed in the anechoic chamber. Furthermore, note that a number of packets at the edge of the area of heavy packet loss are attributed to a weak link. This is an effect of the voter discarding packets with very little corruption, as described in Sec. 3.5. Nonetheless, most packets are correctly classified as WiFi when packet loss exceeds 20%.

The classification results under microwave oven interference are shown in Fig. 7c. In contrast to the anechoic chamber, a considerable number of corrupted packets are classified as WiFi interference. When inspecting individual experiment runs, we found that in four consecutive runs out of the ten runs under microwave interference, a large number of packets was classified as WiFi interference. These four runs strongly affect the average results shown in the figure. The 802.15.4 channel we used in these experiments overlapped with 802.11g channel 11, on which we observed most WiFi traffic from the university WiFi. Thus, it is conceivable that runs were indeed exposed to WiFi interference. However, we do not remove the runs from the data, since we ultimately cannot gauge the impact of the university's WiFi on these four runs. We also point out that a certain number of misclassifications are expected from the accuracy observed in the anechoic chamber. Nonetheless, most of the packets are classified as being interfered by a microwave oven.

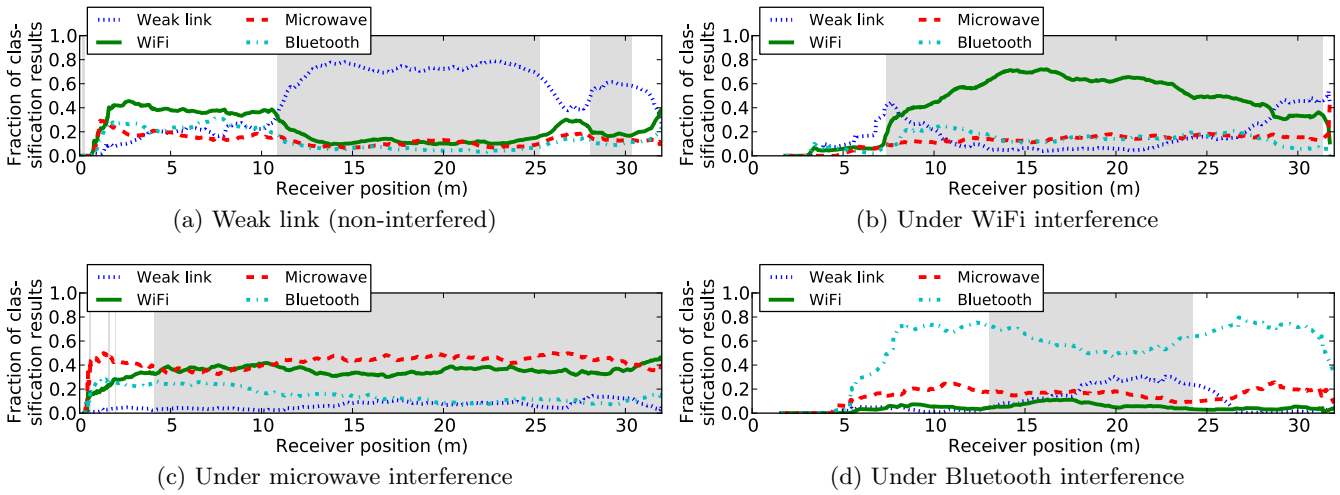
Finally, Fig. 7d shows the average distribution of classification results in the presence of Bluetooth interference. The majority of corrupted packets is classified as interfered by Bluetooth. A number of instances is classified as corrupted by microwave, which is in line with the anechoic chamber classification.

The above results show that the classifier, which has been trained on data from the anechoic chamber, is able to classify the dominant interference source in our experiments. We conclude that our features characterize interferers not only in the anechoic chamber, but also in the uncontrolled radio environment under background RF activity.

5.4 Detection Results

The results from the previous section are averaged over multiple runs. The averaging hides variations in performance between runs. Therefore, we now consider the amount of time that the voter indicated each interference state, while under heavy packet loss. This metric exposes the performance during each run.

The results are shown in Tab. 4, normalized by the total time under heavy packet loss. The table's rows show the true interferer, and the columns show the detected interference state. SoNIC returns the state *unknown* in case there is an insufficient number (< 5) of classified packets for voting. SoNIC correctly detects that the receiver is experiencing interference almost all of the time, and rarely returns the state *unknown*. Furthermore, the time that each interference state is detected corresponds to the results from the previous section. Both weak links and Bluetooth interference are detected with high accuracy. WiFi interference is correctly declared 82% of the time, but 16.2% of the time, SoNIC attributes the packet loss to a weak link. This is in agreement with Fig. 7b, which shows that at the edges of the interference zone, WiFi interference is misclassified as a weak link. Note that if we defined heavy packet loss as PER above 50% (instead of 20%), SoNIC correctly detects WiFi interference 95% of the time. Finally, microwave interference is correctly declared 74.1%. As mentioned before, four consecutive runs of the microwave experiments contained a large number of WiFi classifications, so there may indeed have been uncontrolled WiFi activity during these runs. In



(a) Weak link (non-interfered)

(b) Under WiFi interference

(c) Under microwave interference

(d) Under Bluetooth interference

Figure 7: Average distributions of classification results in the voting window. Under heavy packet loss (gray areas, PER > 20%), SoNIC correctly attributes most corrupted packets to the dominant interferer.

		Percentage of time per detected interference state				
		unknown	weak link	WiFi	microwave	Bluetooth
Interferer	weak link	0.3%	**97.6%**	0.5%	0.3%	0.6%
	WiFi	0.1%	16.2%	**82.0%**	0.5%	1.1%
	microwave	0.6% (0.7%)	0% (0%)	23.2% (17.4%)	**74.1% (79.5%)**	2.0% (2.3%)
	Bluetooth	0%	3.7%	0%	0%	**96.3%**

Table 4: The duration that SoNIC detects an interference state is shown in columns, the dominant interferer is shown in rows. Overall, the correct interferer is detected 87.5% of the time.

parentheses, we state the amount of time per interference state when discarding these four runs, which increases detection to 79.5%.

We conclude that when experiencing packet error ratios of 20% and more, SoNIC correctly detects the interference state 87.5% of the time on average.

6. AUGMENTING A MOBILE SINK WITH SONIC

To show how a sensor network can benefit from SoNIC, we add it to a mobile sink application to detect and mitigate interference. The mobile sink polls its stationary one-hop neighbors for data in a round-robin fashion. When a stationary node receives a request, it responds with a stream of 20 packets of 64 bytes. Retransmissions are used to handle packet loss. We set the number of retransmissions to 30, the same value used in the Collection Tree Protocol [7] implementation in TinyOS.

Mitigation. The mobile sink implements two exemplary mitigation strategies. When WiFi interference is detected, the mobile sink switches communication to another 802.15.4 channel, separated 30 MHz from the interfered channel. In this way, it avoids a frequency overlap with the WiFi channel. To mitigate microwave interference, the nodes time their transmission so they do not coincide with the microwave emissions. Microwave emissions are very regular in time, following a 10 ms on, 10 ms off pattern [2]. When the mobile sink detects microwave oven interference, it requests its peers to schedule transmissions 20 ms apart. If a peer does not receive an ACK for a transmission, it randomizes its transmission time to avoid synchronization with the microwave oven. We selected these two mitigation approaches due to their simplicity. Note that SoNIC can readily be used with other approaches as well, e.g. [4, 15].

PRR under interference, without SoNIC	45.3% (σ = 0.94%)
PRR under interference, with SoNIC	61.7% (σ = 6.16%)
PRR without interference	70.0% (σ = 0.00%)
Mean time to detect WiFi interference	18.14 s (σ = 23.9 s)
Mean time to detect microwave interference	2.57 s (σ = 1.41 s)

Table 5: Summary of the results for mobile sink experiments. The PRR is improved by choosing a suitable mitigation strategy.

Mitigation decisions. The mobile sink uses the following approach to making a mitigation decision. If the average PER of the links to its neighbors exceeds 40%, the mobile sink enters a back-off phase of ten seconds. If PER still exceeds 40% after the back-off phase and either WiFi or microwave interference is detected, the sink activates the respective mitigation strategy. Using a back-off ensures that after the network conditions have changed (as indicated by the increase in PER), a sufficient amount of time is spent in the new conditions to collect corrupted packets.

Scenario. The mobile sink travels along the path denoted by the arrow in Fig. 5 and polls data from the stationary sensor nodes 1–6. For the duration of the whole experiment, the WiFi AP 2 (located near the beginning of the track)

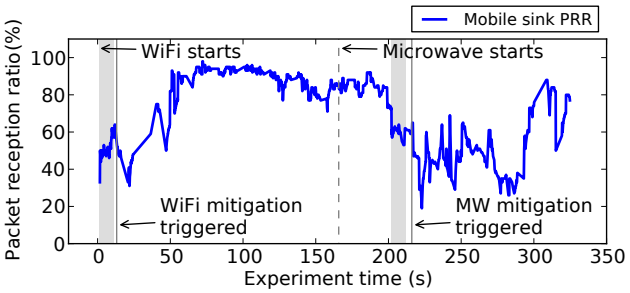


Figure 8: Annotated run of the mobile sink experiment. After PRR has dropped, SoNIC quickly indicates the detected interferer to the application

streams constant bitrate UDP traffic on channel 6, which overlaps with the 802.15.4 channel that the sensor nodes are using for communication. When the mobile sink has traveled half of the path, the microwave is activated for the rest of the run's duration. We repeat the experiment six times.

Results. The key results are summarized in Tab. 5. Using SoNIC significantly improves the number of correctly received packets under interference. In all runs, the mobile sink correctly detected and mitigated WiFi interference at the beginning of the track. It correctly detected and mitigated the microwave interference in five of the six runs. In the remaining run, the mobile sink erroneously detected WiFi interference instead. This performance matches our observations from the previous sections.

Fig. 8 demonstrates the operation of the mobile sink using SoNIC during one run. The figure shows the sink's PRR, and the annotations indicate when the respective interferers have been activated and when the sink has detected them and made a mitigation decision. The gray areas show when the mobile sink backs off after a drop in PRR, before making a mitigation decision. In the beginning of the experiment, the mobile sink experiences poor performance due to WiFi interference. It correctly detects WiFi interference after the 10 s back-off period and activates the appropriate mitigation, i.e., it changes to channel 23, a channel that is heavily affected by microwave interference [2]. PRR gradually increases as the sink informs the other nodes of the new channel. At 165 s, the microwave is activated. The sink's PRR does not drop instantaneously, because it happens to be communicating to a nearby node (and hence is not as strongly affected by the interference), and because the PRR estimation introduces a slight lag. However, once the drop in PRR is registered, the mobile sink backs off and then correctly detects microwave interference and activates the mitigation strategy. PRR does not recover to non-interference levels; this is partly because the microwave mitigation is not as effective as the channel switching, but also caused by the fact that most sensor nodes are located near the beginning of the track, and hence the mobile sink experiences poor connectivity to those nodes regardless of interference.

On average, it takes the mobile sink 18.1 s after the back-off period to trigger a mitigation decision in the case of WiFi interference. This average is inflated by two runs, in which it took 59.0 s and 43.8 s respectively before the mitigation decision was made. There were two reasons: *(i)* the mobile sink was communicating with nearby nodes with strong links, and thus the PER did not exceed 40% after the back-off had elapsed. *(ii)* when querying nodes on links that were strongly affected by the interference, it took a while to collect enough corrupted packets for classification. Taking a decision to mitigate microwave interference took 2.5 s on average after the back-off time. Since microwave ovens are usually operated for short durations in the order of a few minutes, such quick detection is especially desirable.

7. RELATED WORK

The problem of interference in wireless sensor networks is well-known. For example, Sikora and Groza have studied the impact of microwave ovens, Bluetooth and WiFi on packet loss in sensor networks [25]. Petrova et al. have quantified the impact of 802.11 devices on sensor networks [19]. Sha et al. have studied the spectrum usage in home area networks and realized that wireless conditions in homes and hence interference are more complex than in office environments [23]. Also in the context of body area networks, interference is recognized as a problem [8, 10, 12]. Various techniques have been proposed to classify the source of interference for wireless sensor networks. Zacharias et al. have monitored a single IEEE 802.15.4 channel to classify the source of inference using RSSI readings [26]. However, the main part of their data analysis was done offline in Matlab, which is different from the real-time analysis performed by individual sensors in our SoNIC system. Similarly, Bloessl et al. have presented an interferer classification framework, which measures RSSI in a predefined spectrum and displays the signal strengths and their corresponding frequencies in real-time [1]. Different from the above work, SoNIC provides a comprehensive approach to explore a wide range of features for interference classification, including LQI, RSSI and error bursts.

Many authors have proposed solutions for mitigating interference. Some of the solutions are agnostic to the source of interference. For example, Boano et al.'s approach to make MAC protocols robust against interference does not consider the interference source [3]. There are further multi-channel protocols that avoid interference by switching channels [22, 27]. Noda et al. have devised a channel metric to quantify interference that is per design agnostic to the interference source [18]. In contrast to these solutions, we explicitly classify the source of interference and adapt our mitigation strategy to the source of interference.

Alternatively, there are solutions to mitigate interference by observing pattern of specific interference sources [2]. For example, Hauer et al. have suggested how to mitigate the effects of RF interference through RSSI-based error recovery [9]. Liang et al. have specifically targeted WiFi interference and applied forward error detection to combat the impact of this interference source [15]. Chowdhury and Akyildiz have presented an approach that based on the spectral characteristics distinguishes between WiFi and microwave interference [4]. In contrast to us they use a distributed approach that has been evaluated in simulation only. While we aim at classifying the source of interference, Musaloiu-E. and Terzis's goal has been to detect WiFi interference by means of periodic RSSI samples [16]. Both Chowdhury and Musaloiu-E.'s approaches are based on active channel sampling which requires sensor nodes to turn on the radio and hence contributes to higher power consumption by idle listening. On the contrary, SoNIC does not rely on active

spectrum sampling over long timespans or additional hardware, which makes it lightweight and well-suited for low-power sensor nodes.

8. DISCUSSION

SoNIC's classifier distinguishes between WiFi, Bluetooth and microwave oven interference, and packets that are corrupted due to low TX power. To add detection capabilities for a new interference type, suitable features must be defined and the classifier needs to be retrained.

In the experiments in Sec. 5 and Sec. 6 there are a number of interfering devices that we cannot control, in particular the university's WiFi. We have, however, not performed any explicit experiments with multiple interferers. SoNIC is currently designed to identify the main interferer. As described in Sec. 3.5, the voter chooses the most common class of packets in the window as the interfering state and passes this state to the application. To address multiple interferers of different kinds, we would change the voting algorithm to, for example, estimate the likelihood of the presence of a specific interferer.

An open question is if we need to provide additional features or change the features to classify corrupted packets in the presence of multiple interferers. Communication protocols usually have mechanisms to avoid simultaneous transmissions from two or more devices and hence packets are seldomly corrupted by concurrent transmissions. Therefore, packets are not corrupted differently in the presence of multiple interferers of the same type which suggests that our current features are sufficient to handle this case.

Unfortunately, protocols are usually not designed with coexistence in mind and obviously microwave ovens do not consider communication protocols. Hence, there is a higher chance that packets form different interferers are transmitted simultaneously and corrupt packets in ways different from what our features are designed for. We leave the investigation of this problem for future work.

9. CONCLUSION

Sensor networks that use 802.15.4 at 2.4 GHz face cross-technology interference from many other technologies operating in the same frequency band. Previous research has shown that interference mitigation in sensor networks can be more effective if the type of interference is known. This paper addressed the problem of classifying and detecting interference in a sensor network. We introduced a novel approach to interference classification that considers individual, corrupted 802.15.4 packets, rather than using costly continuous spectrum sampling. The evaluation has shown that our implementation of the approach is sufficiently lightweight for use on resource-constrained sensor nodes, and that it correctly detects the predominant interferer in an uncontrolled office environment.

10. ACKNOWLEDGEMENTS

This work has been funded by the VINNOVA through the Uppsala VINN Excellence Center for Wireless Sensor Networks (WISENET). It has been partially supported by the European Commission with contract INFSO-ICT-317826 (RELYonIT). Thanks to our shepherd Chenyang Lu and the reviewers for their comments.

11. REFERENCES

[1] B. Bloessl, S. Joerer, F. Mauroner, and F. Dressler. Low-Cost Interferer Detection and Classification using TelosB Sensor Motes. In *Poster Session, MobiCom 2012*, Istanbul, Turkey, August 2012.

[2] C. A. Boano, T. Voigt, C. Noda, K. Römer, and M. Zúñiga. JamLab: Augmenting Sensornet Testbeds with Realistic and Controlled Interference Generation. In *International symposium on Information processing in sensor networks (IPSN)*, Chicago, IL, USA, Apr. 2011.

[3] C. A. Boano, T. Voigt, N. Tsiftes, L. Mottola, K. Römer, and M. Zuniga. Making Sensornet MAC Protocols Robust Against Interference. In *European Conference on Wireless Sensor Networks (EWSN)*, Coimbra, Portugal, Feb. 2010.

[4] K. Chowdhury and I. Akyildiz. Interferer Classification, Channel Selection and Transmission Adaptation for Wireless Sensor Networks. In *International Conference on Communication (ICC)*, June 2009.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition, Nov. 2001.

[6] A. Dunkels, Björn Grönvall, and T. Voigt. Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors. In *1st Workshop on Embedded Networked Sensors (EmNetS)*, Tampa, Florida, USA, Nov. 2004.

[7] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. In *International Conference on Embedded Networked Sensor Systems (SenSys)*, Nov. 2009.

[8] 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 *European Conference on Wireless Sensor Networks (EWSN)*, Cork, Ireland, Feb. 2009.

[9] J.-H. Hauer, A. Willig, and A. Wolisz. Mitigating the Effects of RF Interference through RSSI-Based Error Recovery. In *European Conference on Wireless Sensor Networks (EWSN)*, Coimbra, Portugal, Feb. 2010.

[10] J. Hou, B. Chang, D. Cho, and M. Gerla. Minimizing 802.11 Interference on ZigBee Medical Sensors. In *International Conference on Body Area Networks (BodyNets)*, Apr. 2009.

[11] 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.

[12] J. Ko, T. Gao, and A. Terzis. Empirical Study of a Medical Sensor Application in an Urban Emergency Department. In *International Conference on Body Area Networks (BodyNets)*, Apr. 2009.

[13] J. Ko, K. Klues, C. Richer, W. Hofer, B. Kusy, M. Bruenig, T. Schmid, Q. Wang, P. Dutta, and A. Terzis. Low Power or High Performance? A Tradeoff Whose Time Has Come (and Nearly Gone). In *European Conference on Wireless Sensor Networks (EWSN)*, Trento, Italy, Feb. 2012.

[14] K. Lakshminarayanan, S. Sapra, S. Seshan, and P. Steenkiste. RFDump: An Architecture for

Monitoring the Wireless Ether. In *ACM CoNEXT 2009*, Dec. 2009.

[15] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis. Surviving Wi-Fi Interference in Low Power ZigBee Networks. In *International Conference on Embedded Networked Sensor Systems (SenSys)*, Zürich, Switzerland, Nov. 2010.

[16] 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.

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

[18] C. Noda, S. Prabh, M. Alves, T. Voigt, and C. Boano. Quantifying the Channel Quality for Interference-aware Wireless Sensor Networks. *ACM SIGBED Review*, 8(4):43–48, 2011.

[19] M. Petrova, L. Wu, P. Mahonen, and J. Riihijarvi. Interference Measurements on Performance Degradation between Colocated IEEE 802.11g/n and IEEE 802.15.4 Networks. In *International Conference on Networking (ICN), 2007*, Sainte-Luce, Martinique, Apr. 2007.

[20] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 2001.

[21] O. Rensfelt, F. Hermans, P. Gunningberg, L. Larzon, and E. Björnemo. Repeatable Experiments with Mobile Nodes in a Relocatable WSN Testbed. *The Computer Journal*, 54(12):1973–1986, 2011.

[22] M. Sha, G. Hackmann, and C. Lu. Arch: Practical Channel Hopping for Reliable Home-area Sensor Networks. In *17th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, Chicago, IL, USA, Apr. 2011.

[23] M. Sha, G. Hackmann, and C. Lu. Multi-channel Reliability and Spectrum Usage in Real Homes: Empirical Studies for Home-area Sensor Networks. In *IEEE 19th International Workshop on Quality of Service (IWQoS)*, San Jose, CA, USA, June 2011.

[24] S. Y. Shin, H. S. Park, and W. H. Kwon. Mutual interference analysis of IEEE 802.15.4 and IEEE 802.11b. *Computer Networks*, 51(12):3338 – 3353, 2007.

[25] A. Sikora and V. Groza. Coexistence of IEEE 802.15.4 with other Systems in the 2.4 GHz-ISM-Band. In *IEEE Instrumentation and Measurement Technology Conference (IMTC)*, 2005.

[26] S. Zacharias, T. Newe, S. O'Keeffe, and E. Lewis. Identifying Sources of Interference in RSSI Traces of a Single IEEE 802.15.4 Channel. In *International Conference on Wireless and Mobile Communications*, Venice, Italy, June 2012.

[27] G. Zhou, Y. Wu, T. Yan, T. He, C. Huang, J. Stankovic, and T. Abdelzaher. A Multifrequency MAC Specially Designed for Wireless Sensor Network Applications. *ACM Transactions on Embedded Computing Systems (TECS)*, 9(4):39, 2010.