




## Article

# Spectrogram Data Set for Deep Learning Based RF-Frame Detection

Jakob Wicht <sup>1,\*</sup> , Ulf Wetzker <sup>1</sup>  and Dr. Vineeta Jain <sup>1,2</sup> 

<sup>1</sup> Fraunhofer Institute for Integrated Circuits, Division Engineering of Adaptive Systems EAS Dresden, Germany; {firstname}.{lastname}@eas.iis.fraunhofer.de

<sup>2</sup> LNM Institute of Information Technology Jaipur, India

\* Correspondence: jakob.wicht@eas.iis.fraunhofer.de; Tel.: +49-351-45691-374

**Abstract:** Automated spectrum analysis serves as a troubleshooting tool that helps to diagnose faults in wireless networks like difficult signal propagation conditions and coexisting wireless networks. It provides a higher monitoring coverage while requiring less expertise compared to manual spectrum analysis. In this paper, we introduce a data set that can be used to train and evaluate deep learning models, capable of detecting frames from different wireless standards as well as interference between single frames. Since manually labelling a high variety of frames in different environments is too challenging, an artificial data generation pipeline has been developed. The data set consists of 20 000 augmented signal segments, each containing a random number of different Wi-Fi and Bluetooth frames, their spectral image representations and labels that describe the position and type of frame within the spectrogram. The data set contains results of intermediate processing steps that enables the research or teaching community to create new data sets for specific requirements or to provide new interesting examination examples.

**Dataset:** <https://fordatis.fraunhofer.de/handle/fordatis/287>

**Keywords:** spectrogram data set; wireless network monitoring; spectrum analysis; frame detection; object detection; deep learning

## 1. Introduction

Due to the flexibility and mobility offered by wireless networks, they are playing an increasingly important role in industrial applications, consumer electronics, medical monitoring and building automation. However, it is challenging to meet the demanding latency and reliability requirements that are particularly prevalent in an industrial context. Applications that fail to meet these requirements suffer degradation leading to economic losses and system failures. Inadequate signal propagation conditions due to path loss, multipath propagation, non-line-of-sight (NLOS) and interference caused by coexistence problems with other wireless networks are among the most common sources of failures that lead to long service downtimes. Especially in the industrial, scientific and medical (ISM) frequency bands, the available channel capacity gets exhausted quickly, which is why coexistence problems have become a major problem. Techniques for mitigating these problems, as implemented in the UMTS (CDMA), Bluetooth (FHSS) and Wi-Fi6 & 5G (OFDMA) communications standards, can not avoid them completely.

System failures caused by faults within the wireless communication system are difficult to detect and can lead to severe financial losses. Time-efficient troubleshooting and preventive countermeasures can only be guaranteed if troubleshooting equipment is used to constantly monitor the wireless communication networks. Intelligent spectrum analysis provides deep insight into the physical layer, enabling the elimination of coexistence problems between individual communication participants.

Many spectrum analysis approaches already exist in literature. Most of these approaches [4–9] employ machine learning algorithms for automatic frame detection and classification. Some of the proposed approaches use advanced image recognition and object



**Citation:** Wicht, J.; Lastname, F.; Lastname, F. Title. *Preprints* **2022**, *1*, 0. <https://doi.org/>

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

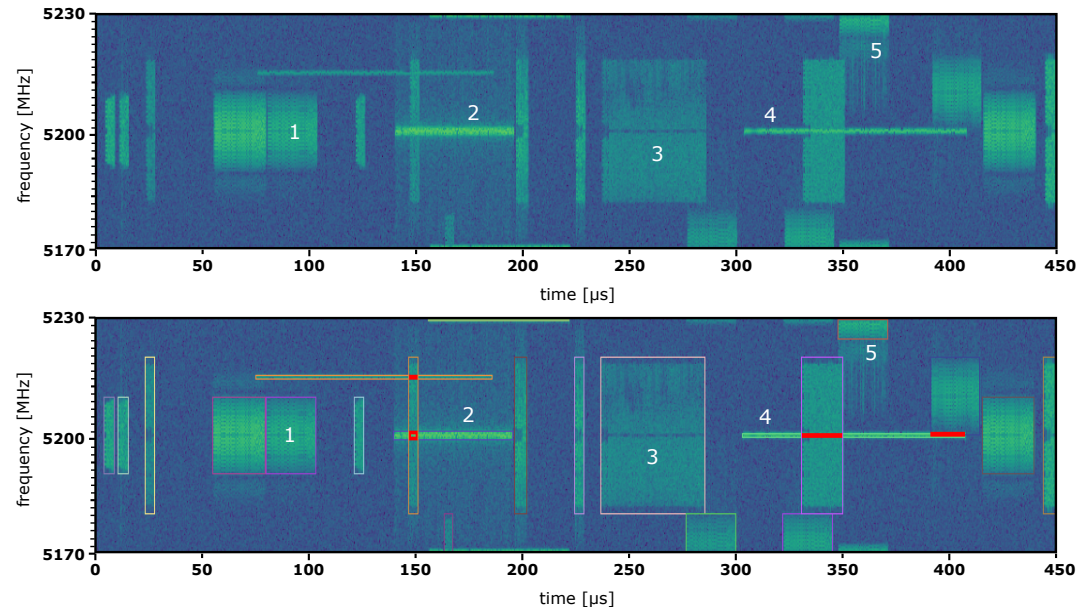
detection algorithms. Our proposed data set has already been evaluated [10] with a state-of-the-art object detection algorithm for detection of coexisting standards and collisions in the network. Although ML-based approaches show significant improvement in accuracy compared to heuristic approaches, generating a substantial labelled data set is challenging. Especially in the case of machine learning based object detection systems, the availability of labelled data is of prime importance. Capturing spectrograms of a real wireless communication in a shielded environment would be an intuitive idea. A radio frequency (RF) channel emulator could provide the required variety of different environments. However, manual labelling is too time consuming. Determining the timestamps and duration of frames sent is not feasible, as real traffic at this level is difficult to measure or reproduce. To generate labelled spectrograms, we have developed a data augmentation pipeline. Using pre-recorded frames of different RF standards, a policy specific to the communication standard randomly arranges these frames and modifies them using a channel model. In this way, the exact position and dimension of each frame is known, while maintaining the resemblance of spectral representations to the real measurements. Moreover, collisions between multiple frames can be generated in this way, which are difficult to detect in a real environment.

The proposed approach can be used to fully automate the generation of a large data set of spectrograms. We even provide python scripts to generate data set using user-specific parameters such as image resolution or color map. The scripts can also be used for conversion of the labels into a different format.

## 2. Data set description

This section gives an overview of the published data set and describes how it is structured. It consists of 20 000 spectrograms, which contain a total of about 362 780 Wireless Fidelity (Wi-Fi) frames, 21 340 frames of each Bluetooth® Low Energy (BLE) 1 MHz, BLE 2 MHz as well as Bluetooth® Classic (BT) and 77 600 collisions between different frames.

Figure 1 contains an example of the spectrograms that form the training data. The horizontal axis represents the time in  $\mu\text{s}$  and the vertical axis represents the frequency in MHz. The brightness encodes the reception strength in dBm and has been colored using the *viridis* scale. The spectrogram depicted corresponds to a sampling rate of 60 MS per second and includes complementary code keying (CCK) (e.g., frame 1) and orthogonal frequency-division multiplexing (OFDM) (frame 3) modulated 802.11 frames as well as 2 MHz BLE (frame 2) and BT (frame 4) frames. Frame 5 is a 20 MHz Wi-Fi frame that is only partially within the acquisition bandwidth of the spectrogram. The bottom figure illustrates the same spectrogram, with the individual frames marked by colored bounding boxes. The thick red lines illustrate the areas where two or more frames overlap, referred to as collisions.



**Figure 1.** Example of a simulated spectrogram. It has been generated using the software based method and has 60 MHz bandwidth.

The data set not only contains the fully labelled simulated spectrogram frames, but also files from two intermediate processing steps. First, there are separate frames recorded at a high Signal-to-noise ratio (SNR) that serve as input to the augmentation pipeline. Second, there are chunks of the time-domain signal in the form of complex values, available as raw binary data from which final spectrogram images can be generated. The data set is stored in the following directory tree:

```
spectrogram_training_data_20220711/
├── single_packet_samples/
│   ├── config_packet_capture.toml
│   ├── wlan/
│   │   ├── frame138649214288305070-0-0__sampRate_
│   │   │   1.25E+08_len_15_enc_1_signalLvl_0_bw_40E+06_
│   │   │   freqOffset_0.00E+00_std_WAC.packet
│   │   ├── frame138649214288305070-0-0__sampRate_
│   │   │   1.25E+08_len_15_enc_1_signalLvl_0_bw_40E+06_
│   │   │   freqOffset_0.00E+00_std_WAC.png
│   │   └── ...
│   ├── BLE_1MHz/...
│   ├── BLE_2MHz/...
│   └── BT_classic/...
├── merged_packets/
│   ├── bw_25e6/
│   │   ├── frame_138769090412766230_bw_25E+6
│   │   ├── labels_138769090412766230_bw_25E+6.csv
│   │   └── ...
│   ├── bw_45e6/...
│   ├── bw_60e6/...
│   └── bw_125e6/...
└── results/
    ├── result_frame_138769289703249260_bw_125E+6
    ├── result_frame_138769289703249260_bw_125E+6.txt
    ├── result_frame_138769289703249260_bw_125E+6.png
    ├── result_frame_138769289703249260_bw_125E+6_marked.png
    └── ...
```

The `./single_packet_samples/` directory contains all the frames being used by our automated pipeline to assemble them into the augmented spectrograms. These images can also be reused for creation of a new or modified pipeline with a different focus. Within the filenames, the properties of each frame have been encoded and therefore represent part of the labels. The binary files `*.packet` contain the complex-valued time signal samples of a single frame, consisting of pairs of 32-bit floating point numbers. The configuration file `./single_packet_samples/config_packet_capture.toml` contains parameters used for generating and capturing the single frames of the respective RF standard, described in more detail in section 3.1.

The `./merged_packets` directory contains sample sections populated with a selection of single frames to which a simulated channel model has been applied. These single-packet samples have the same format as the time-domain signal files described earlier. All corresponding labels are available as csv files with the same identification number. These labels contain detailed information about all frames within the signal section, see table 1.

**Table 1.** Description of the label file columns.

| Column                | Description                                                                                                               |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------|
| id                    | index given to a frame when added; When describing a collision it refers the involved frames indices, connected by a '-'. |
| sample_position_start | number of samples after which the frame is added to the section                                                           |
| sample_position_end   | number of samples after which the added frame ends                                                                        |
| pdu_length            | payload of the frame in number of bytes (empty when collision)                                                            |
| level                 | signal level of the frame in dB, having an arbitrary reference                                                            |
| bandwidth             | bandwidth of the frame signal                                                                                             |
| freq_offset           | frequency shift from centre frequency of the sample range                                                                 |
| class                 | String giving the class name [WLAN, BT_classic, BLE_1MHz, BLE_2MHz, collision]                                            |
| rf_std                | String giving the specific Wi-Fi standard (empty if not Wi-Fi)                                                            |
| WLAN_mcs              | modulation coding scheme of a Wi-Fi frame (empty if not Wi-Fi)                                                            |
| BT_packet_type        | modulation coding scheme of a BT or BLE frame (empty if not BT or BLE)                                                    |
| noise_lvl             | mean magnitude of noise level of the complete section or a string "usrp_txrx_loop" in case of hardware induced noise      |
| sample_rate           | sample rate of the complete section                                                                                       |
| samples_total         | number of samples within the complete section                                                                             |
| doppler_speed_kmh     | speed of objects that would produce the emulated doppler effect on that frame                                             |
| k_factor              | channel model parameter of the individual frame (ratio of line-of-sight signal power over the scattered signal power)     |
| multipath_components  | channel model parameter of the individual frame (number of reflections)                                                   |
| PDP_delays            | channel model parameter of the individual frame (delay (in samples) for arriving reflected Ray)                           |
| PDP_delay_max_dev     | channel model parameter of the individual frame (maximum deviation of delay per reflection)                               |
| PDP_delay_std_dev     | channel model parameter of the individual frame (step size gaussian standard deviation per reflection)                    |
| PDP_mag               | channel model parameter of the individual frame (magnitude of each arriving reflected Ray)                                |
| identifier            | unique identifier used within the filename                                                                                |

The `./results/` directory contains the final spectrogram images. They are stored in png file format together with the corresponding label files in Yolov4 format. In addition, there is a duplicate of each spectrogram containing the bounding box visualizations for all object classes. These files were generated using a Yolov4 object detector specifically trained for this purpose. However, all images containing bounding boxes are only used for evaluation of the generated results and must not be used for training.

The next sections provide a detailed description about how the published data set was created.

3. Data set generation

Figure 2 provides an overview of the developed data generation and enhancement pipeline, which is described in detail in the following sections.



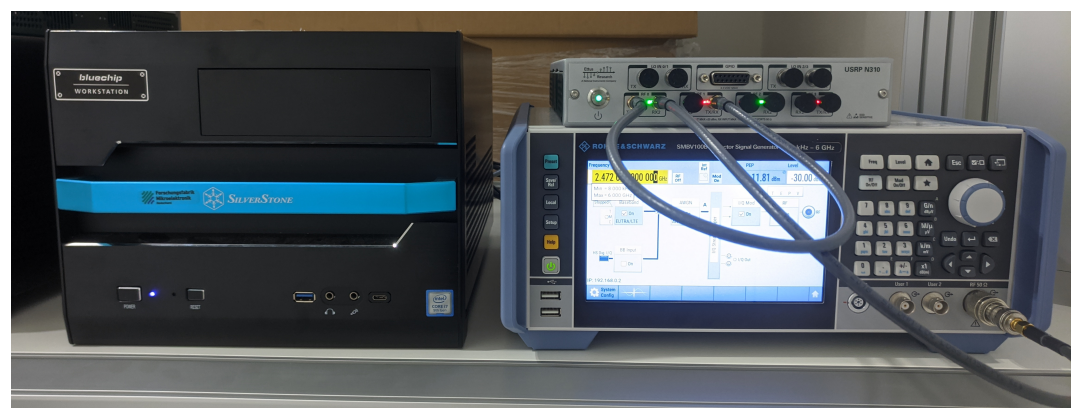
**Figure 2.** General structure of our automated pipeline

At the first stage, parameterizable frames of different RF standards are generated by a Rhode & Schwarz<sup>®</sup> SMBV100B vector signal generator [1] and recorded by a Software Defined Radio (SDR), details explained in section 3.1. Individual frames are then extracted from the recorded signal and stored separately. The second stage is called *data augmentation*, details explained in section 3.2. Here, the signals of the individual frames are randomly arranged, according to a policy corresponding to the RF standards, to form larger signal sections of 4.5 ms. During composition, a separate channel model is applied to each frame. Since the positioning of the frames is random, a number of frame collisions occur (shown in red in Figure 2), and the overlapping regions are labelled separately. This approach allows to simulate different RF environments whose appearance closely resembles real scenarios and significantly increases the variability within the data set. Subsequently, the composite signal sections are converted into actual spectrogram images, which are stored together with accurate labels, details explained in section 3.3. By following the method described above, we have generated 20 000 spectrograms representing the given data set.

The following section, 3.1, describes the setup for generating and recording the various 802.11, BLE and BT frames, which serve as input to the actual augmentation pipeline.

### 3.1. Single frame acquisition

A Rhode & Schwarz<sup>®</sup> SMBV100B vector signal generator [1] is used to generate raw single frame signals. The parameters of these frames are defined in a configuration file (`./single_packet_samples/config_packet_capture.toml`) and read by a Python program, which automates the process of signal generation and capture. To record the generated signals, a Universal Software Radio Peripheral (USRP) N310 SDR from Ettus Research<sup>®</sup> [2] is used, which is connected to the vector signal generator via a coaxial cable to ensure sufficient SNR. All generated signals are recorded at a fixed sampling rate of 125 MS/s. From the measurements in form of a continuous sampling stream, individual frames are isolated using a simple threshold-based method and stored in separate files. Figure 3 shows the setup of the hardware components. In Figure 3, the output of the signal generator is connected to an input of the USRP via a coaxial cable. Additionally, an output of the USRP is directly connected to an input to loop back the signal of the simulated spectrogram. See section 3.3.2, spectrogram generation using the USRP pipeline for more details.



**Figure 3.** Hardware setup used to generate and capture single frame signals. Host PC with an 10 Gbit/s network card (left), SMBV100B vector signal generator (right) and the USRP N310 on top.

Table 2 displays the variety of parameters of different RF-standards used for creation of data set. They can also be found in the data set directory tree in `./single_packet_samples/config_packet_capture.toml`.

**Table 2.** RF-standard parameters used for creation of data set.

| Communication Standards                            | Frame Parameters                                                                                                              | Total number of frames per standard                    |
|----------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|
| IEEE 802.11 b/g<br>IEEE 802.11 n<br>IEEE 802.11 ac | Payload length<br>Modulation Coding Scheme (MCS)<br>Frame bandwidth<br>Packet type (Data, beacon, trigger or sounding frames) | 480 (147 x 20 MHz +<br>144 x 40 MHz +<br>189 x 80 MHz) |
| Bluetooth® Low Energy (BLE)                        | Payload length<br>Channel type (ADV, DATA)<br>Packet type (DATA, AIND)<br>Packet format (L1M, L2M, LCOD)                      | 29 (8 x 2 MHz +<br>21 x 1 MHz)                         |
| Bluetooth® Classic (BT)                            | Payload length<br>Packet type (DHx: 1Mbps; ADHx: 2Mbps, AEDHx: 3Mbps)<br>Channel type (ADV, DATA)                             | 29                                                     |

After the frame records of all defined RF standards are available for the defined parameter combinations, they are stored in `./single_packet_samples/{RF-Standard}`. The following section 3.2 describes the data augmentation pipeline for creating simulated RF-environments. It has been implemented entirely in software, using single frames as an input to generate the actual training data set.

### 3.2. Generation of simulated signal environments

To create spectrograms with a realistic appearance, we combine the signal of single frames, that has been recorded as described in section 3.1, into a mutual signal segment. To maximize the diversity of the data set, different channel models are applied to each frame, defined by a set of random parameters. Table 3 lists all parameters that affect the spectrogram segments. The described parameters specify the distribution of the frames within the dataset to be created, representing the ratio of the individual standards in the entire dataset or the minimum and maximum values of a uniform distribution of values. For instance, the value of *ratio of a spectrograms without a frame* parameter highlights that there are 3 % of spectrograms in the data set that only contain background noise.

**Table 3.** Parameters used for generating the signal sections and resulting spectrograms as defined in `./config_training_data.toml`.

| Parameter                                     | Value                  |
|-----------------------------------------------|------------------------|
| sample rates                                  | [25, 45, 60, 125] MS/s |
| number of spectrograms per sample rate        | 5000                   |
| ratio of RF-standards [Wi-Fi]                 | 0.85                   |
| ratio of RF-standards [BT]                    | 0.05                   |
| ratio of RF-standards [BLE (1 MHz)]           | 0.05                   |
| ratio of RF-standards [BLE (2 MHz)]           | 0.05                   |
| time section per spectrogram                  | 4.5 ms                 |
| number of frames per spectrogram [min, max]   | [18, 25]               |
| maximum number of frame collisions            | 4                      |
| ratio of spectrograms without a frame         | 0.03                   |
| ratio of spectrograms with a single frame     | 0.1                    |
| ratio of spectrograms generated using an USRP | 0.2                    |
| amplitude range of added noise [min, max]     | [0.0055, 0.0065]       |
| resolution of the spectrogram images [x, y]   | [1024, 192]            |

The signal segments are generated with different sample rates. In this way, all sampling rates supported by the SDR can be covered either by training a single model or by separate models. The sampling rates are defined in a tuple, where the number of generated signal segments is equal for all rates. To generate the present data, we chose the sampling rates supported by the USRP N310, which correspond to integer divisions of its master

clock. Since all frames are captured at the maximum sampling rate of 125 MS/s, they are downsampled to the target rate when added to a section.

Besides having the ratio of the included RF-Standard defined, the chance of selecting a frame with a higher bandwidth than the target sample rate of the spectrogram section is hard-coded to be 1 %. The ratio of the remaining frame parameters like payload and modulation scheme distribution is implicitly defined by single frame generation parameters, see Table 2. The imbalance in RF standards in favor of Wi-Fi over BT and BLE was intentional, since the effects of the different channel models on wideband signals are greater, and long signals send with a higher bandwidth also allow for much more complex collision patterns. This leads to a higher diversity compared to narrowband standards, which has to be considered when creating the data set.

The section length of 4.5 ms per spectrogram has been iteratively found suitable for detecting even the shortest frames when the image is compressed to the target resolution while maintaining the real-time capability of the detection algorithm [10]. The number of frames between 18 and 25 to be added into a section is considered high for that length. Since the position of a frame is picked randomly, the diversity of inter frame spacings and frame collision pattern is high. As shown in the flowchart in Figure 4, a new frame and position is selected once the maximum number of collisions is reached. Figure 4 illustrates the algorithm for combining individual frames into artificial spectrograms and how the random parameters are used.

**Figure 4.** Methodology used for spectrogram augmentation

The range of additive white Gaussian noise (AWGN) to be added was chosen experimentally so that all frames are of varying strength and are not lost in background noise due to too low SNR. This background noise is added before the spectrograms are created using the software-based method.

Compared to Table 3, which contains the parameters for the main spectrogram configuration, Table 4 defines the channel model parameters per single frame before it is added to a spectrogram segment.

**Table 4.** Random channel model applied to a single frame, before being added to a spectrogram.

| Parameter                                  | Value                               |
|--------------------------------------------|-------------------------------------|
| gain per frame [min, max]                  | [-20, 6]                            |
| frequency offsets                          | -65 MHz...65 MHz, step size = 5 MHz |
| ratio of frames with frequency offset      | 0.2                                 |
| channel model [max k factor]               | 10                                  |
| channel model [is ricean]                  | true                                |
| channel model [max doppler speed]          | 20 km/h                             |
| channel model [max multi-path components]  | 6                                   |
| channel model [delay standard deviation]   | 0.0                                 |
| ratio of frames with multi-path components | 0.5                                 |

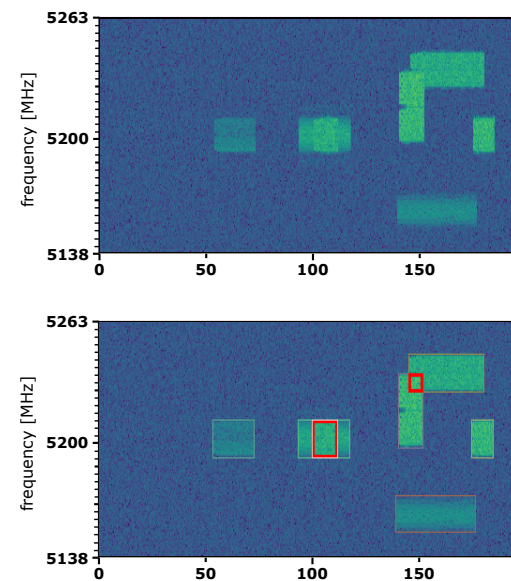
The first parameter in the list describes the limits of gain values that are picked from a uniform distribution and applied on a frame when being added to the spectrogram section. In this way, different distances between the transmitting nodes and the spectrum analyzer's observation position are simulated. To achieve the same mean spectral power density for all standards, the signal is amplified according to its bandwidth and normalized to 1 MHz.

$$gain = random\_gain + 10 \log_{10} \frac{signal\_bandwidth}{1 \text{ MHz}} dB$$

The next parameter is the list of frequency offsets, which defines the center frequency shift of a frame from the horizontal center of the spectrogram. Only 20 % of the frames are shifted from the center frequency to avoid too many frames being only partially within the

sampling bandwidth. In order to apply a frequency shift to the time signal of a frame, it is upconverted by multiplying it with a sinusoidal signal having a frequency corresponding to the shift. It is then downconverted back to the target sampling rate, where aliasing effects are avoided by the upconversion and downconversion.

The remaining parameters are used to tune the multipath propagation model individually for each frame. A simpler model consisting only of a reflection path and random Doppler shift due to moving objects is applied to 50 % of the frames. Figure 5 shows a spectrogram containing only the frames generated using these models. In particular, for the frames labelled 1, 2, and 3, clear frequency-selective fading effects of different simulated Doppler velocities is observed, but no frequency spreading is seen.



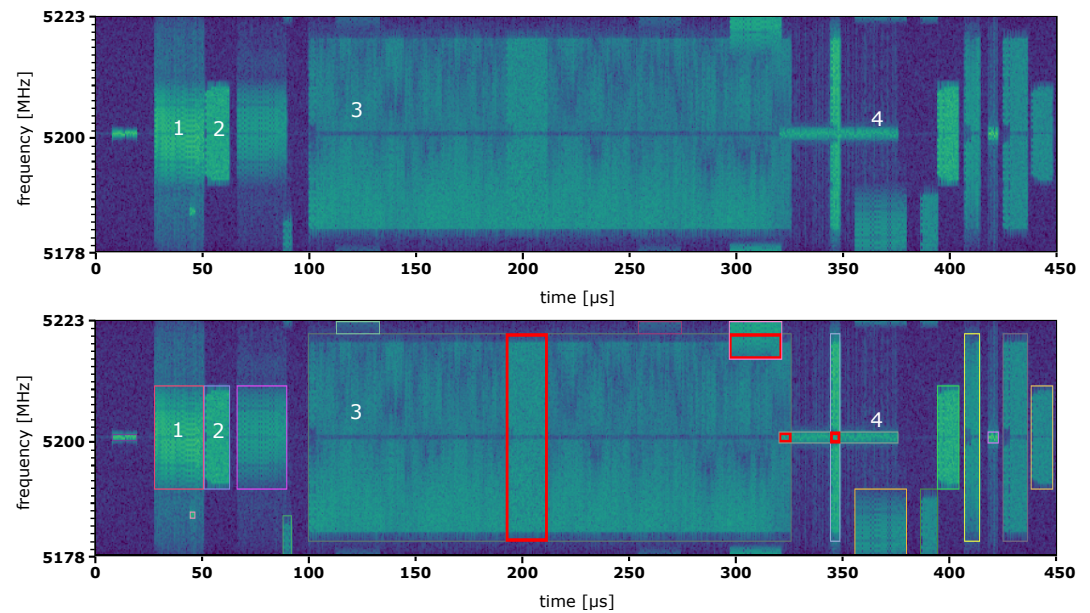
doppler/result<sub>frame138687118374386790,w125E+6.pdf</sub>

**Figure 5.** Example spectrogram with 125 MHz bandwidth. Only the doppler effect has been applied to all the frames.

A more complex frequency selective fading model is applied [11,12] to the other half of the frames. It simulates an environment with reflective objects and surfaces and is called Rician fading. A random combination of the K-factor, the number of multipath components (MPC), and a set of random sub-parameters for each reflected ray (time delay, size, and standard and maximum deviation of power delay profile (PDP)) describes a user-defined RF channel for each frame. The K-factor defines the ratio of the determined direct line-of-sight signal power to the scattered signal power. For low K-factors, the line-of-sight component becomes subdominant, approaching Rayleigh fading.

Different combinations of model parameters can have different effects on the spectral signal appearance, as shown in Figure 6.





**Figure 6.** Example of a spectrogram with 45 MHz bandwidth. The Doppler effect and more complex multipath propagation has been simulated.

Some significant channel parameters and their spectral representation for different frames types are marked in Figure 6. The corresponding channel parameters are listed in Table 5.

Frame 1 (20 MHz 802.11 b/g, CCK modulation) is severely affected by the effect of Doppler spreading and propagation along different paths, resulting in spectral broadening. The channel parameters of Frame 2 (20 MHz 802.11 n, Modulation Coding Scheme (MCS)=3) have a higher Doppler shift than Frame 1. However, Frame 2 has only a reflection component with no delay variation and has effectively been modelled as a determined line of sight transmission. Frame 3 (40 MHz 802.11 ac, MCS=1) is an example of a strong influence by MPC. A clear fading is evident, showing a wide variation over time. Frame 4 is a 2 MHz BLE data frame with only one reflection that exhibits a delay deviation over time, which together with the high K-factor results in a slight frequency spread.

**Table 5.** RF-standard parameters used for creation of data set.

| frame id | fading model parameter                                                                                                                                                                                                                                                           |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1        | doppler_speed = 17 km/h<br>K-factor = 5<br>multipath components = 6<br>PDP delays = [1 2 3 4 5 6] samples<br>PDP delay_max_dev = [0.1 0.2 0.3 0.4 0.5 0.6]<br>PDP delay_std_dev = [0.0121 0.0072 0.0101 0.0095 0.0053 0.0061]<br>PDP magnitude = [0.97 0.79 0.79 0.67 0.54 0.38] |
| 2        | doppler_speed = 19 km/h<br>K-factor = 7<br>multipath components = 1<br>PDP delays = [1]<br>PDP delay_max_dev = [0]<br>PDP delay_std_dev = [0]<br>PDP magnitude = [1]                                                                                                             |
| 3        | doppler_speed = 7 km/h<br>K-factor = 6<br>multipath components = 5<br>PDP delays = [1 2 3 4 5]<br>PDP delay_max_dev = [0.1 0.2 0.3 0.4 0.5]<br>PDP delay_std_dev = [0.0096 0.0195 0.0067 0.0049 0.0033]<br>PDP magnitude = [0.87 0.78 0.75 0.59 0.5]                             |
| 4        | doppler_speed = 5 km/h<br>K-factor = 10<br>multipath components = 1<br>PDP delays = [1]<br>PDP delay_max_dev = [0.1]<br>PDP delay_std_dev = [0.0037]<br>PDP magnitude = [0.9]                                                                                                    |

After the augmentation of the signal sections, they are still present in form of complex numbers in binary format. The last step consists of generating spectrogram images.

### 3.3. Spectrogram generation

The prior composition of frames into segments is done on time domain signals. To generate a time-spectral representation of these sections that can be used by image processing algorithms, two different methods are employed. The first spectrogram calculation is completely software based. It provides full control over the added noise and its Fast Fourier Transformation (FFT) output shows higher magnitude resolution compared to the hardware based generation. The second method consists of the same pipeline that is used within the actual measurement and processing stage. It utilizes a USRP N310 for sending, recapturing and a hardware-based calculation of the FFT from the time domain signal inside the on-board Field Programmable Gate Array (FPGA). While this approach allows for results that are very close to a real measurement system, the software-based generation of spectrograms provides better reusability due to its generic approach. For this reason, 80 % of the spectrograms were obtained using the software-based method.

Regardless of the method used, the resulting spectrogram images must be reduced to a temporal resolution which guarantees real-time frame detection inference. Here, it must be taken into account that as little information as possible is lost.

The following subsections describe the individual components of the two spectrogram generation methods. Both of them take the time-domain signal of 4.5 ms sections as input including combined frames with individual channel models, as described in Section 3.2.

### 3.3.1. Software based spectrogram generation

All sample segments that should be processed by this method have a noise level defined in their respective labels file and the AWGN has been already added in order to simulate the thermal noise of a receiving amplifier.

The Python library matplotlib [3] is used to generate the spectrograms, scale them to the target resolution and save them as lossless PNG images. The 'Hanning' window is used for interpolation and 'viridis' is used for the colormap. A three channel colormap showed better results than grey-scale with our detection algorithm. Additional Python programs are provided allowing a convenient regeneration of images for the use of different color maps or resolutions.

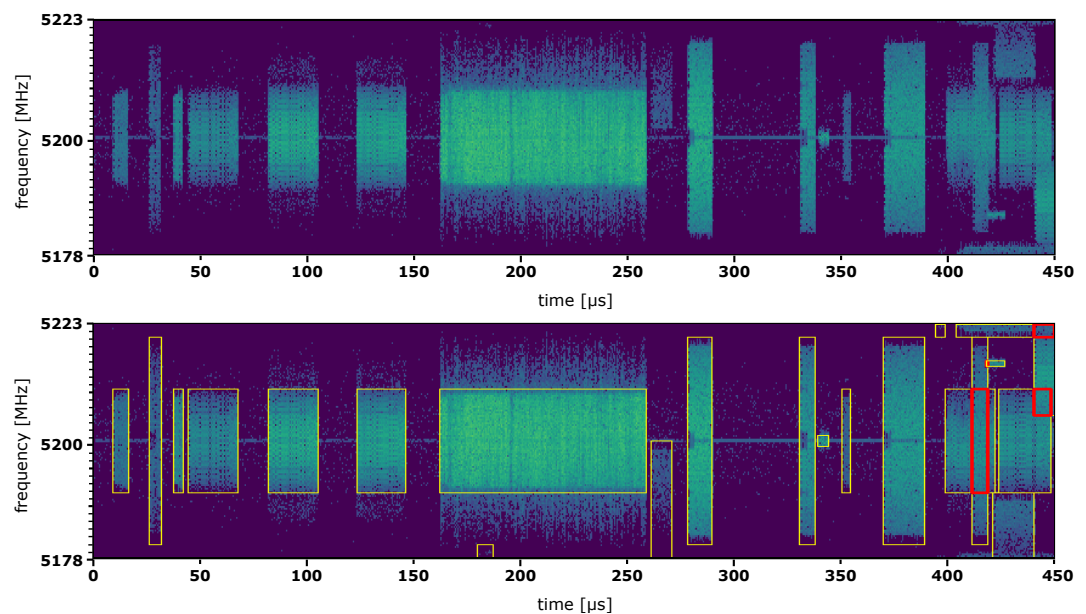
### 3.3.2. SDR loopback spectrogram generation

Figure 7 shows the transmit-receive loopback pipeline used to generate spectrograms with the characteristic properties of a radio communication.

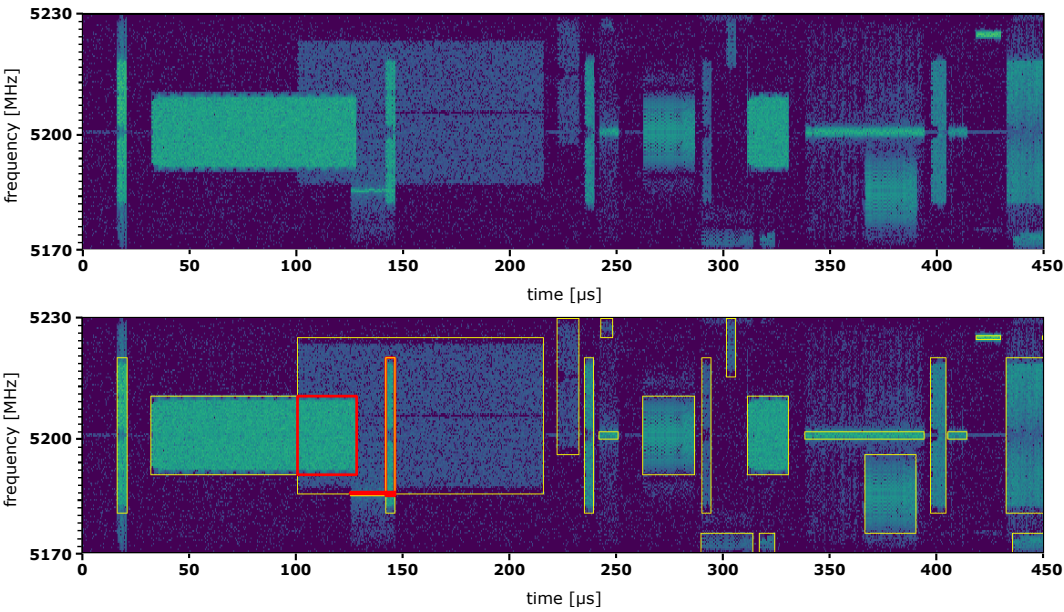
**Figure 7.** Pipeline that is not only used during the actual measurement and inference stage but also during training data generation for having samples with appearance close to actual measurements.

All function blocks within the USRP are implemented in hardware and controlled by the host PC. The transmit and receive pipelines must be synchronized precisely. The resolution of the magnitude of the hardware-FFT is lower compared to the software based implementation by matplotlib. In order to preserve low intensity parts of the spectrum, an additional gain block shifts the range before calculating the squared power density. With this method, the noise floor is created by the actual hardware amplifier and not added artificially, making it more in line with reality.

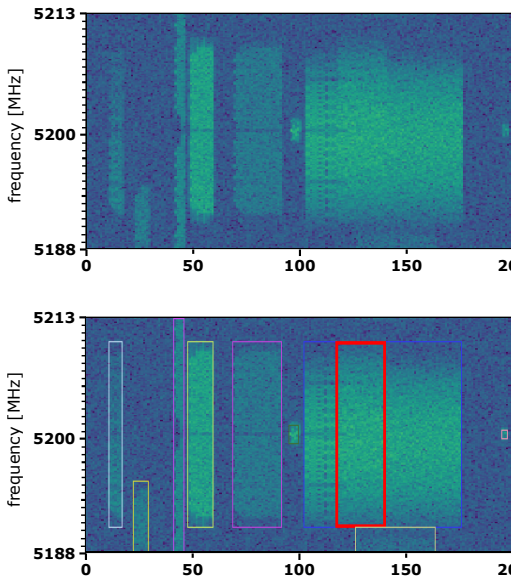
The following Figures 8 - 11 are further examples of spectrograms to give an impression about the appearance of the different parameters like sample rate and method of spectrogram generation. The difference in appearance when using the hardware based method is visible in Figures 8 and 9 in direct comparison to Figures 10 and 11 or previously shown spectrograms.



**Figure 8.** Example of a simulated spectrogram, created using the USRP loopback at 45 MS/s



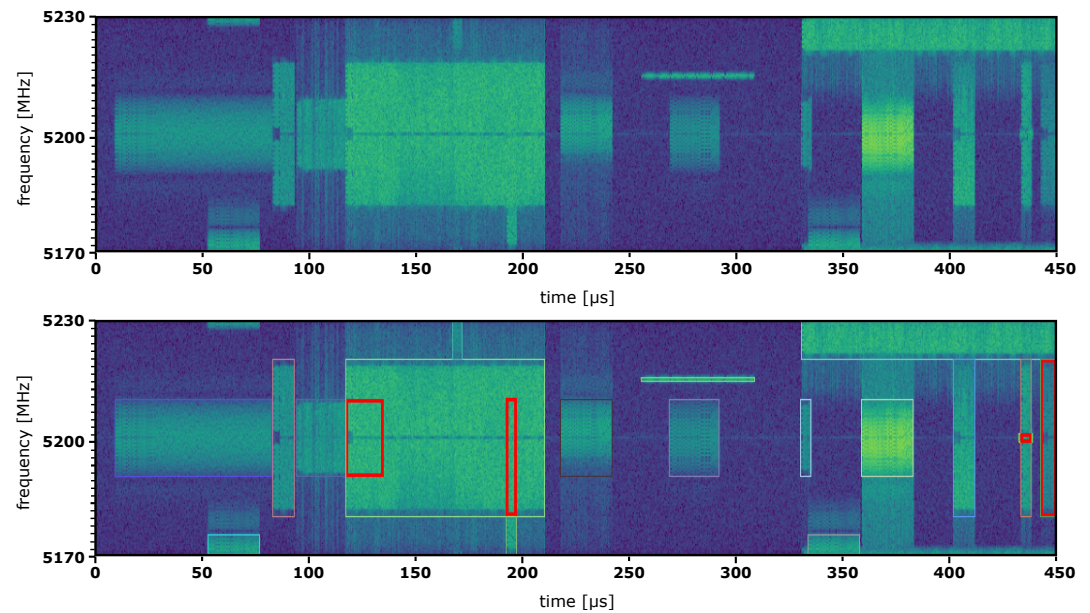
**Figure 9.** Example of a simulated spectrogram, created using the USRP loopback at 60 MS/s



doppler/result<sub>frame<sub>1</sub>38687116768133381<sub>b</sub>w<sub>2</sub>5E+6.pdf</sub>

**Figure 10.** Example of a simulated spectrogram, created using software based method with a bandwidth of 25 MHz





**Figure 11.** Example of a simulated spectrogram, created using software based method with a bandwidth of 60 MHz

#### 4. Discussion

In this paper, we present an extensive data set of spectrograms that can be used as training data for detecting frames of several RF standards. The data set is fully and extensively labelled, making it suitable for training machine learning models using supervised learning techniques, as well as for verifying the detection accuracy of computer vision algorithms. The data set was created using a novel method that records real radio signals of individual frames transmitted under controlled conditions and creates spectrograms using a specially designed data augmentation pipeline. In contrast to the direct recording of a transmission, the augmentation pipeline offers the possibility to freely parameterize the channel models to be used. This procedure greatly improves the versatility of the environmental conditions represented within the data set. Frames can be selectively placed in the spectrum, allowing the recreation of specific scenarios and extensive labelling of each image.

This data set was initially developed to train an object detection system that can recognize single frames from different RF standards and collisions between frames. Considering this objective, these specially tailored spectrograms differ from real measurements. The main deviation is inaccurate representation of temporal dependencies between frames within a spectrogram. Communication between the individual participants of a network is governed by certain definitions of a communication standard, which can be recognized in the spectrum in a condensed form. This includes the duration and time interval between individual frames as well as switching to a different channel or to a different bandwidth. For example, frames sent in rapid succession from one network node are subject to almost the same channel characteristics and thus there are no sudden changes in the reception power at the receiving node.

These restrictions on the use of the data set can be addressed in the future. Functional extensions are straightforward since we not only provide a single data set, but also the underlying raw frames and the essential Python programs to generate the spectrograms. Simulator coupling or a more simple rule-based approach could be used to model temporal dependencies of a communication realistically. Subsequent developments include the generation of labelled spectrogram videos to enable the training of specific models for real-time spectral analysis, and an extension of the frames database to include sources of interference such as microwaves or radar systems that can be used for generating spectrograms.



In its current form, this data set offers a wide range of possibilities for testing and developing algorithms from the field of computer vision and in the context of research and teaching. It serves as a valuable and easy-to-use reference data set.

**Author Contributions:** J.W. is responsible for conceptualization, software, data set preparation, validation, visualization, and writing; U.W. contributed to conceptualization, software, and writing; V.J. contributed to writing

**Funding:** This work was funded by the Federal Ministry of Education and Research of the Federal Republic of Germany (BMBF) within the PENTA project “SunRISE” (<https://www.project-sunrise.eu/>) under the Project Number 16ES0974 and in cooperation with the Center for Analytics – Data – Applications (ADA-Center) which is supported by the Bavarian Ministry of Economic Affairs, Regional Development and Energy within the framework of “BAYERN DIGITAL II” (20-3410-2-9-8).

**Data Availability Statement:** The data set described in this document is archived under <https://fordatis.fraunhofer.de/handle/fordatis/287> or <http://dx.doi.org/10.24406/fordatis/216>. Corresponding helper scripts are available under [https://gitlab.cc-asp.fraunhofer.de/ifk\\_public/sunrise/public-mdpi-dataset-helper-scripts/-/tree/dataset\\_20220711](https://gitlab.cc-asp.fraunhofer.de/ifk_public/sunrise/public-mdpi-dataset-helper-scripts/-/tree/dataset_20220711)

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

## References

1. Rhode & Schwarz R&S® SMBV100B, [https://www.rohde-schwarz.com/uk/products/test-and-measurement/vector-signal-generators/rs-smbv100b-vector-signal-generator\\_63493-519808.html](https://www.rohde-schwarz.com/uk/products/test-and-measurement/vector-signal-generators/rs-smbv100b-vector-signal-generator_63493-519808.html), July 29th, 2022
2. USRP N310. Ettus Research, a National Instruments Brand. <https://kb.ettus.com/N300/N310>, July 29th, 2021
3. Thomas A Caswell, Michael Droettboom, John Hunter, Antony Lee, Eric Firing, David Stansby, Jody Klymak, Elliott Sales de Andrade, Jens Hedegaard Nielsen, Nelle Varoquaux, Tim Hoffmann, Benjamin Root, Phil Elson, Ryan May, Darren Dale, Jae-Joon Lee, Jouni K. Seppänen, Damon McDougall, Andrew Straw, ... Elan Ernest. (2019). matplotlib/matplotlib: REL: v3.1.1 (v3.1.1). Zenodo. <https://doi.org/10.5281/zenodo.3264781>
4. Fehske, A.; J. Gaedder; Jeffrey H. Reed, A new approach to signal classification using spectral correlation and neural networks. *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks*, **2005**.
5. Yu Jianyuan; Mohammad Alhassoun; R. Michael Buehrer, Interference classification using deep neural networks. *IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*. **IEEE**, **2020**.
6. Toma Andrea; Tassadaq Nawaz; Yue Gao; Lucio Marcenaro; Carlo S. Regazzoni, Interference mitigation in wideband radios using spectrum correlation and neural network. *IET Communications* vol. 13, no. 10, pp. 1336–1347, **2019**.
7. O'Shea; Timothy J.; Johnathan Corgan; T. Charles Clancy, Convolutional radio modulation recognition networks. *International conference on engineering applications of neural networks*. Springer, Cham, pp. 213–226, **2016**.
8. O'Shea; Timothy James; Tamoghna Roy; T. Charles Clancy, Over-the-air deep learning based radio signal classification. *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp 168–179, **2018**.
9. Mennes Ruben et al, Deep learning-based spectrum prediction collision avoidance for hybrid wireless environments. *IEEE Access* vol. 7 pp: 45818–45830, **2019**.
10. Jakob Wicht; Ulf Wetzker; Andreas Frotzcher, Deep Learning Based Real-Time Spectrum Analysis for Wireless Networks. *26th European Wireless Conference*. VDE, pp 1–6, **2021**.
11. A. Alimohammad, S. F. Fard, B. F. Cockburn and C. Schlegel, "An Accurate and Compact Rayleigh and Rician Fading Channel Simulator," *VTC Spring 2008 - IEEE Vehicular Technology Conference*, pp. 409-413, doi: 10.1109/VETECS.2008.97. **2008**
12. F. Ren and Y. R. Zheng, "A low-complexity hardware implementation of discrete-time frequency-selective Rayleigh fading channels," *2009 IEEE International Symposium on Circuits and Systems*, pp. 1759-1762, doi: 10.1109/ISCAS.2009.5118116. **2009**