Internet of Things (IoT) Discovery Using Deep Neural Networks

Ephraim Lo
North Point Defense
elo@northpointdefense.com

JoHannah Kohl
North Point Defense
jkohl@northpointdefense.com

Abstract

We present a novel approach to Internet of Things (IoT) discovery using Deep Neural Network (DNN) based object detection. Traditional methods of IoT discovery are based on either manual or automated monitoring of predetermined channel frequencies. Our method takes the spectrogram images that a human analyst visually scans for manual spectrum exploration and applies the state-of-the-art You Only Look Once (YOLO) object detection algorithm to detect and localize signal objects in time and frequency. We focus specifically on the class of signals that employ the Long Range (LoRa) modulation scheme, which uses chirp spread spectrum technology to provide high network efficiency and robustness against both in- and out-of-band interference. Our detection system is designed with scalability for real or near real-time processing capabilities and achieves 81.82% mAP in real-time on a fourth generation mobile Intel CPU without GPU support. Lastly, we present preliminary detection results for other IoT signals including Zigbee, Bluetooth, and Wi-Fi.

1. Introduction

The Internet of Things (IoT) is a network of devices that enable connectivity, interactivity, and the exchange of data with a broad set of applications, ranging from consumer level home automation and wearable technologies to large-scale infrastructure development. These IoT devices communicate using intermittent transmissions that need to be robust to interference and can employ proprietary technology with flexible transmission frequencies. With the ubiquity of such devices, it has become increasingly difficult to identify and catalog the wireless spectrum. While commercial hardware solutions exist for Radio Frequency (RF) sniffing, they are often limited to predetermined channel frequencies.

For more generalized discovery, the simplest method requires an experienced analyst to manually scan the RF with a spectrum analyzer or receiver paired with a digital waterfall display of the spectrum. Once a signal of interest has been visually identified, it can be subsequently filtered and routed to follow-on processing (demodulation, payload extraction, validation). However, due to expanding frequency coverage requirements coupled with increasingly transient IoT communications, the likelihood of missed transmissions is significantly increased. Methods such as energy detection and matched filtering are susceptible to these same issues and can yield unacceptably high rates of false positives. Other recent approaches [1–3], leverage Deep Learning (DL) to automatically classify wireless signals, but typically assume a baseband In-phase/Quadrature (IQ) input that has already been isolated from concurrent signal activity.

Our novel approach to IoT discovery takes as an input the spectrogram images that a human would view and applies Deep Neural Network (DNN) based object detection. We re-imagine signals as objects that exist within an "image" of the spectrum and leverage object detection algorithms to localize these signal objects in time and frequency by predicting bounding box coordinates and class probabilities jointly.

We focus our efforts on the class of signals that employ Long Range (LoRa) communication technology, a derivative of chirp spread spectrum [4]. These signals exhibit distinct time-frequency visualizations comprised of up- and down-chirps that naturally fit our object detection paradigm. See Figure 1.

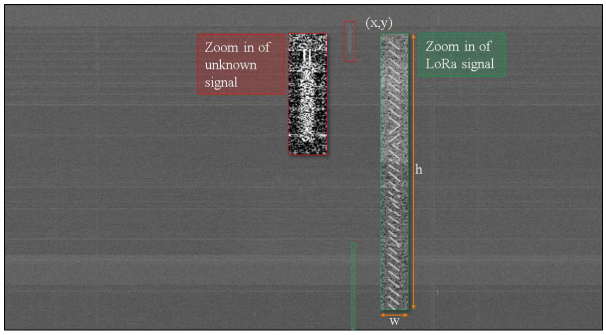

Figure 1. Wideband spectrogram image with illustration of bounding boxes.

Maximizing discovery potential requires using high bandwidth Software-Defined Radio (SDR) solutions. However, these high sampling rates result in the generation of vast quantities of spectrogram images. For example, sampling at 32 MHz would generate over 30 Frames Per Second (FPS) assuming a 4096-FFT split into four 1024x1024 images. Even with the assistance of current GPU hardware for accelerated inference, state-of-the-art object detection systems often struggle to process 20 FPS, while CPU-only inference throughput can easily be an order of magnitude lower. This places a significant burden on the detection system to achieve real or near real-time processing.

Currently, one of the fastest object detection approaches is the You Only Look Once (YOLO) algorithm [5]. Despite being fast, YOLO also has comparable accuracy to other state-of-the-art detectors and can be scaled depending on Size, Weight, and Power (SWaP) constraints. On a fourth generation Intel® CoreTM i7-4700EQ processor at 2.4 GHz with 8GB RAM without discrete GPU support, we were able to achieve real-time processing of 24 MHz bandwidth with 81.82% mAP on a set of real LoRa signals collected over-the-air from just under 2 miles away.

2. Related Work

Traditional algorithmic approaches to signal detection in RF systems typically fall into one of two categories: generalized methods that cannot take advantage of contextual emission information and specialized methods that only work on the signals they are designed for. However, an alternate approach has emerged with the growth of DL technology that achieves the robustness of specialized approaches, but with high generalization capability.

2.1. One-dimensional case

In [1], autoencoders were applied to raw IQ samples for the problem of modulation classification and in [2], Convolutional Neural Networks (CNNs) were leveraged to detect weak signals in strong background noise. To simulate realistic data, Cheng [2] adopted the Longley-Rice channel model with additive white Gaussian noise to simulate topographical factors at various receiver and transmitter locations. However, a major limitation of such approaches is the assumption that the time series input is free from both in- and out-of-band interfering signals. Bitar [3] addressed this by collecting thousands of over-the-air samples at varying Signal-to-Noise Ratios (SNR) with both homogeneous cases of each signal type, as well as heterogeneous co-interfering cases. They demonstrated that CNN models can outperform traditional machine learning techniques in identifying Wi-Fi, Zigbee, and Bluetooth signals in the presence of co-channel interference. However, Bitar [3] ignored realistic transmission channel effects and oversimplified the interference scenarios by collecting in a semi-anechoic chamber with a limited number of transmitting devices.

2.2. Two-dimensional case

DL has been applied to audio spectrograms for tasks ranging from birdsong classification in [6, 7] to CNN-based audio enhancement in [8]. Furthermore, Pham [9] utilized CNNs to exploit the texture and structure of patterns in these spectrograms to detect audio events such as car horns, dog barks, and gunshots. Lastly, Boddapati, et al. [10] designed an environmental sound classifier using object recognition DNNs with various image representations for temporal localization of audio events.

In the RF domain, Lees et al. [11] demonstrated that DNNs greatly outperformed classical signal detection methods in detecting SPN-43 air traffic control radar using spectrograms. Their DNN detector was trained and tested on over 14,000 spectrograms containing realistic channel conditions and out-of-band emissions collected from the 3.5 GHz band. By only using real-world data, Lees et al. [11] had difficulty balancing their training, validation, and test sets with sufficient representation of each radar signal class. O'Shea [12] employed YOLO to detect and localize signals with varying single carrier modulation types from spectrogram images for the DARPA Battle of the ModRecs competition. They built a GNU Radio simulator to generate training datasets with labeled, wideband radio data and reported promising qualitative performance, but did not specify training parameters or quantitative mAP results.

3. Object detection

Recent advances in object detection systems have transitioned from highly inefficient sliding window approaches to a single network evaluation, from image pixels directly to bounding box locations and associated class predictions. Current state-of-the-art detection systems include R-CNN, Faster R-CNN, Single Shot Multibox Detector (SSD), and YOLO [5, 13–17]. While two stage detectors (R-CNN, Faster R-CNN) reach the highest accuracy rates, one stage detectors (SSD, YOLO) have only slightly lower accuracy and run significantly faster. Although it is difficult to definitively compare algorithms due to inconsistent test conditions, including hardware setups and evolving code bases, research consistently places YOLO among the leading algorithms where inference speed is concerned. This advantage is our main motivation for selecting the YOLO algorithm as the basis for our detection system.

3.1. You only look once

YOLO was first introduced in 2016 [5]. Since then, the authors have released two subsequent versions that built upon their original algorithm. YOLOv1 made significant localization errors and had relatively low recall, especially

in comparison to region proposal methods such as R-CNN. YOLOv2 mitigated those deficiencies while maintaining its speed by adjusting the architecture and introducing anchor boxes [13]. YOLOv3 more significantly modified the baseline network by doubling the number of convolutional layers and linking predictions from three separate scales to improve small object detection with a minor speed penalty [14]. Additionally, there are "tiny" versions of each network that reduce the number of layers, sacrificing accuracy for speed. We determined that this detection performance hit was unacceptably large, and so we only considered the larger networks. Analysis of inference throughput shows that YOLOv2 is the fastest of the three standard architectures and as such, it was selected for our research efforts. For the rest of this paper, we will refer to it as simply YOLO and its smaller variant as Tiny-YOLO.

3.2. Network architecture

YOLO is implemented in a custom neural network backend called Darknet. This network has 19 convolutional layers and 5 maxpooling layers. For a detailed network architecture description, see Table 1. Since the model only uses convolutional and pooling layers, it can be resized arbitrarily during training. By periodically randomizing the network size, the network learns to predict well across a variety of input dimensions. This provides the additional benefit that with only a single training run, we can scale the network size as necessary to meet processing requirements. Reducing network size increases inference speed, albeit at the cost of reduced accuracy [13].

3.3. Training

The input image is divided into an SxS grid, where each grid cell is responsible for detecting objects that are centered within that cell. These cells predict B bounding boxes consisting of x, y, w, h along with associated confidence scores using anchor boxes. The (x, y) coordinates represent the center of the box relative to its cell boundaries. The (w, h) represent the width and height of the box. Finally, the confidence scores reflect both the conditional class probability and how well the predicted box fits the object and are defined as:

$$\begin{aligned} C_i &= Pr(Class_i|Object) * Pr(Object) * IOU_{pred}^{truth} \\ &= Pr(Class_i) * IOU_{pred}^{truth}, \end{aligned} \tag{1}$$

where IOU is the intersection over union between the predicted bounding box and ground truth.

Because each grid can predict multiple bounding boxes, some large objects or objects that span numerous adjacent cells may result in excessive detections. Non-maximum suppression (NMS) can be used to remove these superfluous predictions. The anchor priors can be handpicked or generated by running k-means clustering on the training set ground truth to improve training stability and convergence.

Type	Filters	Size	Output
Convolutional	32	3 x 3	288 x 288
Maxpool		2 x 2/2	144 x 144
Convolutional	64	3 x 3	144 x 144
Maxpool		2 x 2/2	72 x 72
Convolutional	128	3 x 3	72 x 72
Convolutional	64	1 x 1	72 x 72
Convolutional	128	3 x 3	72 x 72
Maxpool		2 x 2/2	36 x 36
Convolutional	256	3 x 3	36 x 36
Convolutional	128	1 x 1	36 x 36
Convolutional	256	3 x 3	36 x 36
Maxpool		2 x 2/2	18 x 18
Convolutional	512	3 x 3	18 x 18
Convolutional	256	1 x 1	18 x 18
Convolutional	512	3 x 3	18 x 18
Convolutional	256	1 x 1	18 x 18
Convolutional	512	3 x 3	18 x 18
Maxpool		2 x 2/2	9 x 9
Convolutional	1024	3 x 3	9 x 9
Convolutional	512	1 x 1	9 x 9
Convolutional	1024	3 x 3	9 x 9
Convolutional	512	1 x 1	9 x 9
Convolutional	1024	3 x 3	9 x 9
Convolutional	1000	1 x 1	9 x 9
Avgpool			1000
Softmax			

Table 1. Darknet-19 architecture given an input image size of 288x288.

During training, a multi-part loss function is optimized that has adjustable weights to tailor the resultant model to meet detection objectives and is defined as:

$$\begin{aligned} &\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \alpha_{ij}^{obj} \left[(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2 \right] \\ &+\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \alpha_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{x_i}} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h_i}} \right)^2 \right] \\ &+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \alpha_{ij}^{obj} (C_i - \hat{C_i})^2 \\ &+\lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \alpha_{ij}^{noobj} (C_i - \hat{C_i})^2 \\ &+ \sum_{i=0}^{S^2} \alpha_{i}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p_i}(c))^2, \end{aligned} \tag{2}$$

where α_i^{obj} denotes if an object appears in cell i and α_{ij}^{obj} specifies that the jth bounding box, which has the highest

IOU with the ground truth, is responsible for that object prediction. By adjusting λ_{coord} and λ_{noobj}, the network scales the penalty of predicting a bounding box with inaccurate height and width and for predicting an object in a cell when there isn't any [5, 13].

4. Internet of things signal types

LoRa is a proprietary physical (PHY) layer implementation that uses spread spectrum modulation, providing high resistance to natural interference, making it ideal for use in challenging urban and suburban environments [4]. Additionally, as it defines only the PHY layer, it is compatible with many network architectures such as LoRaWAN [18], a low power, wide area networking protocol. Because LoRaWAN targets critical IoT requirements including connectivity, end-to-end security, and adaptability, it has emerged as one of the fastest growing IoT communication technologies.

In LoRa modulation, the spreading is achieved by modulating symbols onto up- and down-chirp signals with different spreading factors. The symbol rate R_S can be defined as:

$$R_S = \frac{BW}{2^{SF}} \; symbols/sec, \tag{3}$$

where BW is the modulation bandwidth in Hz and SF refers to the spreading factor, ranging from 6 to 12. We can then define the desired modulation bit rate R_b and chip rate R_C as:

$$R_b = SF \cdot R_S \; bits/sec \tag{4}$$

$$R_C = R_S \cdot 2^{SF} = BW \; chips/sec. \tag{5}$$

A LoRa packet has the following basic structure [19]:

- Preamble: series of up-chirps.
- Sync: two modulated up-chirps that can be used for differentiating LoRa signals and can be set to specific values to distinguish public and private networks for LoRaWAN.
- Start of frame delimiter: two and a quarter down-chirps.
- Data: series of modulated up-chirps.

See Figure 2 for a LoRa packet example in a spectrogram view.

In total, LoRa supports 70 variations, composed of ten bandwidths and seven spreading factors, which we refer to using the shorthand notation (BW, SF). This results in a very challenging class set as we need multiple time-frequency representations for adequate visualization resulting in much higher computational load. See Figure 3.

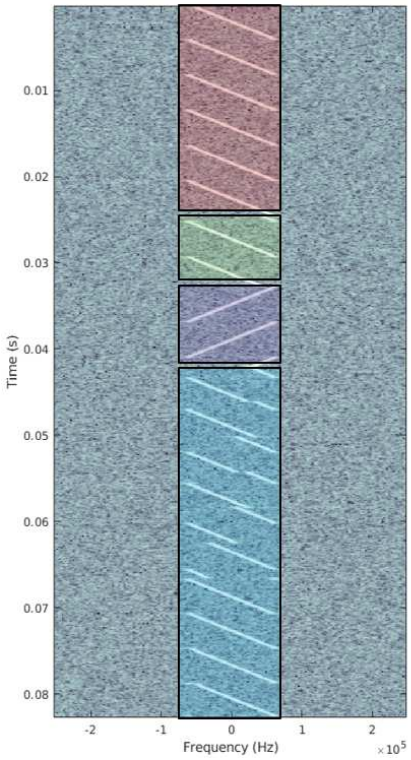

Figure 2. LoRa packet example. The signal is comprised of a preamble (red), sync word (green), start of frame delimiter (purple), and data (cyan).

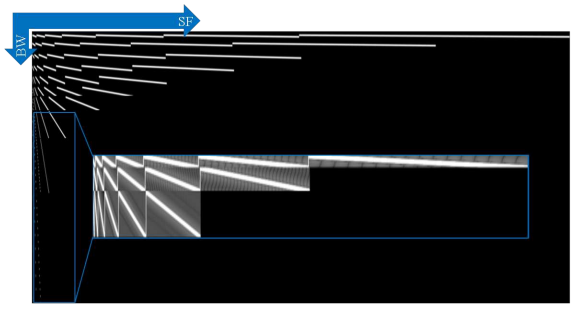

Figure 3. Visualization of a single LoRa symbol at varying bandwidths and spread factors. The bottom four rows representing bandwidths 62.5 to 500kHz are enhanced for visibility.

For example, consider the narrowest and slowest variant $(7.8, 12)$ with the widest and fastest $(500, 6)$. To provide enough time and frequency resolution to visualize the $(7.8, 12)$ signal at 32 MHz sampling rate, we could use a large FFT size of 65536. A single symbol would span 16 FFT bins and 256 rows. However, a single symbol of $(500, 6)$ would span 1024 FFT bins and less than a single

Region	Band	(BW (kHz), SF)
Europe	EU433, 863-870	(125,7:12), (250,7)
China	CN470-510, 779-787	(125,7:12)
United States	US902-928	(125/500,7:12)
Australia	AU915-928	(125/500,7:12)

Table 2. LoRaWAN regional parameters. The band frequencies include additional countries that adopt FCC regulations for that associated ISM band.

row. Likewise, using a smaller FFT of 8192 would represent the 500 kHz signal with 128 FFT bins, but only 2 for the 7.8 kHz signal. To mitigate this problem, we limit our class set with some rough guidance from LoRaWAN regional parameters, which are restricted to a much smaller subset of bandwidths and spread factors [20]. See Table 2.

Several other IoT signals of interest that were investigated include Zigbee, Bluetooth, and Wi-Fi. Zigbee is an IEEE 802.15.4-based specification that leverages direct sequence spread spectrum along with both binary phase-shift keying in the 868/915 MHz bands and offset quadrature phase-shift keying in the 2.4 GHz band. Bluetooth uses frequency hopping spread spectrum between 2.4 and 2.485 GHz where each hop is modulated using Gaussian frequency-shift keying. And lastly, Wi-Fi refers to the radio technologies based around the IEEE 802.11 standards.

5. Data, training, and testing setup

To develop a robust training set and mitigate the tedium of manual annotation, we created an end-to-end simulation pipeline. Initial testing was done on simulated data with real-world performance evaluated on over-the-air LoRa collects.

5.1. Simulation

Starting with the RF, background collects were taken at various frequencies corresponding to some of the LoRaWAN regional parameters discussed earlier. Then, LoRa signals of various (BW, SF) were digitally simulated and inserted with random SNR, automatically generating bounding box annotations. Because of LoRa's high resistance to in-band interference, we allowed the inserted LoRa signals to be "overlaid" on top of pre-existing communications in the background. Spectrogram images were then generated from these waveforms and we randomly adjusted the noise floor to simulate various receiver scenarios. Additionally, to simulate events such as random device failure and dropped LoRa transmissions, we inserted LoRa fragments that do not conform to the general packet structure, i.e. no preamble, no modulated data. Lastly, to further increase our training data diversity, we implemented domain-specific image augmentations that simulate realistic receiver and channel transmission conditions. For each training batch, the images were randomly augmented with simulated narrowband interfering signals, exponential chirps, multipath channel models, and analog-to-digital converter saturation. We also included the default YOLO image augmentations to simulate various lighting conditions but removed random cropping and flipping as they are not relevant for our problem set.

BW	Spread Factor						
(kHz)	6	7	8	9	10	11	12
0.8	64	32	16	8	4	2	1
10.4	85.3	42.7	21.3	10.7	5.3	2.7	1.3
15.6	128	64	32	16	8	4	2
20.8	170.7	85.3	42.7	21.3	10.7	5.3	2.7
31.3	256	128	64	32	16	8	4
41.7	341.3	170.7	85.3	42.7	21.3	10.7	5.3
62.5	512	256	128	64	32	16	8
125	1024	512	256	**128**	**64**	**32**	**16**
250	2048	1024	512	256	**128**	**64**	**32**
500	4096	2048	1024	512	256	**128**	**64**

Table 3. Maximum number of symbols representable. This assumes a sampling rate of 32MHz, 4096-FFT, decimation rate of 4, and image size of 1024x1024. The bold highlighted cells correspond to the nine classes currently supported.

The receiver's sampling rate is set to 32 MHz, and the spectrogram images are generated using a 4096-FFT, skipping every three rows to eventually form a fixed image size of 1024x4096. We then exclude the edge frequencies corresponding with the receiver's roll-off filtering, roughly 20% of the total receive bandwidth. The remaining usable bandwidth of 24 MHz is subsequently divided into three 1024x1024 images. We retain a square image to preserve aspect ratio when resizing. Based on this frequency resolution, a 125/250/500 kHz bandwidth signal can be represented by 16/32/64 bins respectively. The maximum number of representable symbols for each (BW, SF) combination is shown in Table 3 and provides insight into how big or small our signal objects can be.

At inference time, each 1024x1024 image is resized according to the desired network size. Figure 4 depicts how (BW, SF) can affect visual features with different resizing factors. As the resized image gets smaller, the representation fidelity of the signal degrades. This is more noticeable for some (BW, SF) combinations compared to others.

5.2. Collection

To send and verify over-the-air LoRa signals, we used the transmitter and relay shown in Figure 5. The transmitter was controlled via a switch that sends $(125, 9)$ LoRa packets continuously for as long as the switch is depressed and the relay, upon successful reception of the transmitter signal, sends a short acknowledgement reply that is also $(125, 9)$.

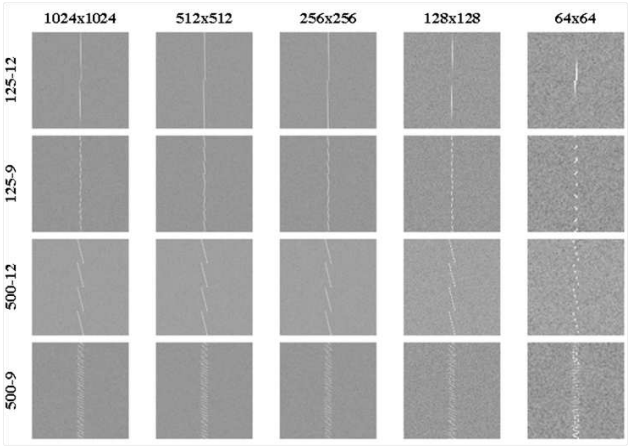

Figure 4. Visualization of LoRa representation fidelity when resizing. The top two rows are LoRa signals with $(125, 12)$ and $(125, 9)$, while the bottom two rows are $(500, 12)$ and $(500, 9)$. Each original image of 1024x1024 on the left represents the same amount of time. As symbol time directly corresponds with bandwidth and spread factor, we can compare the maximum representable symbols. Progressing from left to right, each column represents reducing the image size by an integer factor of 2.

Figure 5. LoRa transmitter (right) and acknowledgement relay (left).

The collect setup, see Figure 6, is comprised of an omni antenna connected to a DRS Polaris SDR tuned to 915 MHz with 32 MHz receive bandwidth. A high-speed Thunderbolt interface is used to transfer the IQ samples from the receiver and spectrograms are computed in software. The acknowledgement relay was located near the receiver and collects were taken at various intervals with distances ranging from 325 m to 4750 m. In total, this "lorafarm" dataset is composed of just over 2000 images.

6. Results

We trained YOLO with the nine classes emphasized in Table 3. The object and coordinate weights for the loss function were increased from the original YOLO values to emphasize detection and localization as we are less concerned with false positives. Nine custom anchors were generated to provide better priors and improved training stability. We also experimented with using a fixed training network size of 416 ($S = 13$) as well as multi-scale training ($S = 10, \ldots, 19$). Default values for B, batch size, momentum, decay, and learning rate schedule were adopted from [13]. Lastly, we used both pre-trained weights on the COCO detection dataset as well as randomly initialized weights. The random initialization yielded superior results both in terms of detection performance, but also dramatically reduced training epochs. This may be because our signal objects look markedly different from traditional image datasets comprised of objects such as people, animals, and various inanimate objects, thereby reducing transfer learning effectiveness.

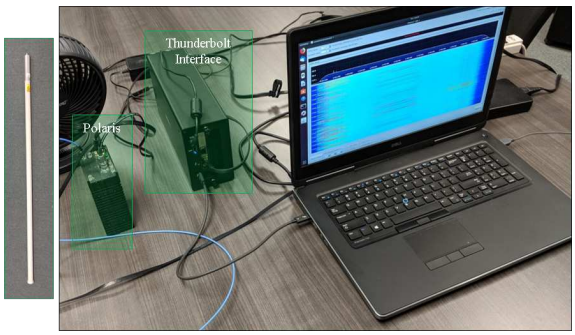

Figure 6. Collect setup using an 8 dBi omnidirectional antenna (left) paired with a DRS 2-channel Polaris wideband tuner with Thunderbolt interface (right).

To test our models, we used a simulated set of 1000 images and the lorafarm set. Many of the simulation parameters discussed in Section 5 were altered and expanded to provide a more unbiased test evaluation. For all our tests, we set the detection NMS threshold to 0.3. On the lorafarm dataset, we achieved 81.82% mAP with real-time processing using YOLO with a network size of 288, with a peak mAP of 90.91% at 512. On our simulated dataset, we achieved 63.69% at 288 and 89.49% mAP at 512. Tiny-YOLO had 77.27% and 82.25% peak mAP on lorafarm and simulated datasets respectively. See Figure 7 for more details.

We expected multi-scale to outperform fixed-size training at network sizes other than 416 and, for most cases, this assumption held true. Interestingly, multi-scale training outperformed fixed training even at 416 for both simulation and lorafarm datasets. See Figure 8 for more details.

Several curated examples of the LoRa detection system outputs are shown in Figures 9-11. We also trained a YOLO model on Zigbee, Bluetooth, and Wi-Fi (802.11b) and an example output is depicted in Figure 12. For illustration purposes only, some of these images have had their brightness and contrast adjusted by up to 40% to better display

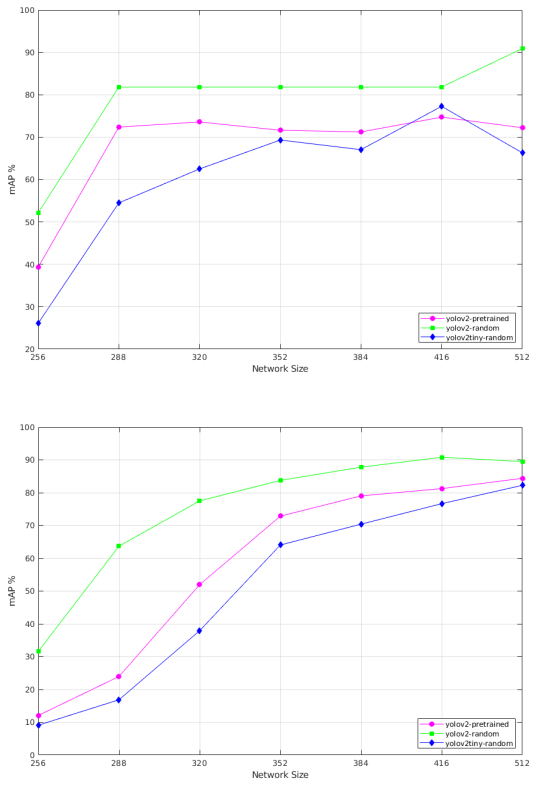

Figure 7. Detection results for lorafarm (top) and simulated (bottom) data sets for YOLO and Tiny-YOLO with various weight initializations.

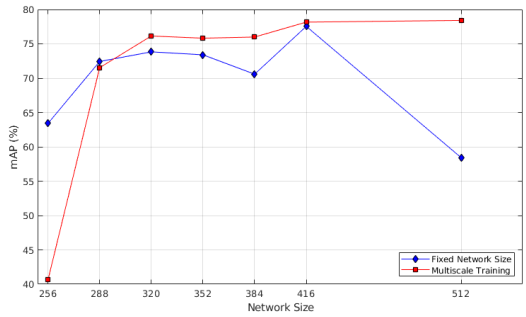

Figure 8. Results on lorafarm comparing multi-scale and fixed network size training. Performance did not scale linearly with distance due to changing line-of-sight conditions.

background details.

Using the RF collect settings described in Section 5, a set of three images needs to be processed in under 524.88 ms to achieve real-time processing on our hardware platform. Table 4 shows our timing benchmarks using TensorFlow YOLO on an Intel® Core™ i7-4700EQ mobile processor at 2.4GHz with 8 GB RAM without GPU support. While

Size	Pre-proc. Time (ms)	Inference Time (ms)	Total Time (ms)	RT Diff. (ms)
256	41.300	120	401.300	+123.580
288	41.369	150	491.369	+34.511
320	41.245	170	551.245	-24.365
352	41.298	210	671.298	-143.418
388	41.348	260	821.348	-292.468
416	41.300	280	881.300	-351.420

Table 4. Timing breakdown for SWaP constrained platform. Preprocessing time includes FFT calculations, grayscale conversion and image resizing. The inference times are for a single image evaluation and total time assumes three images processed sequentially. RT (real-time) differential is the amount of timing overhead left before needing to process the next set of images for real-time performance.

Tiny-YOLO could run at real-time speeds at the larger network sizes, the detection results were unacceptably low and thus not considered for the final detection system.

7. Conclusion

We have presented a robust, lightweight system for IoT discovery using the YOLO object detection algorithm by exploiting the unique time-frequency visualization of LoRa class signals. Through careful analysis of the problem space and extensive image augmentation and simulation, the resultant detection system can learn generalizable representations of these signal objects.

Ultimately, just as a human would need different resolution images to distinguish between all the LoRa variants, we will need a variety of time-frequency resolution spectrograms. We hope to explore this in the future along with ensemble DNNs. Additionally, to manage increased computational burden, there are many software optimizations that can be implemented such as batch inference, quantization to int8 or int16 precision, and utilizing Intel's SIMD instruction set [21]. Leveraging these techniques would also enable our current system to increase inference throughput and detection performance with higher resolution images.

References

[1] T. O'Shea and J. Hoydis. An introduction to deep learning for the physical layer. *IEEE Transactions on Cognitive Communications and Networking*, pages 563–575, 2017.

[2] T. Cheng, C. Liu, and W. Ding. Weak signal detection based on deep learning. *Proceedings of the 2019 4th International Conference on Multimedia Systems and Signal Processing*, pages 114–118, 2019.

[3] N. Bitar, S. Muhammad, and H. Refai. Wireless technology identification using deep convolutional neural networks. *IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017.

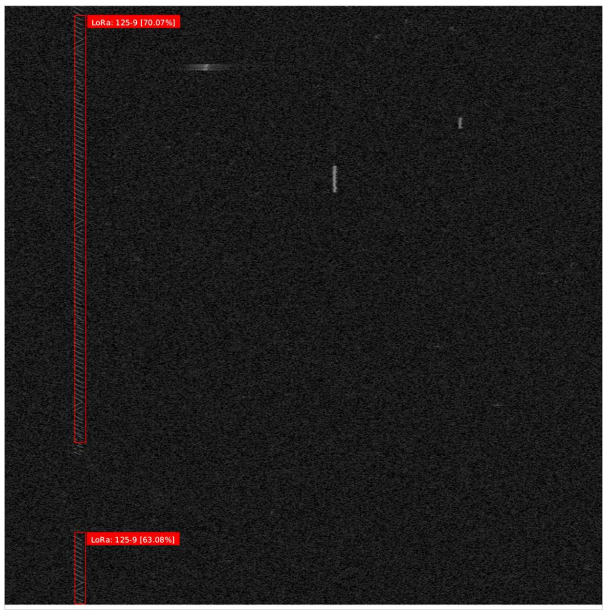

Figure 9. Example detection on lorafarm at 3150 m. Bounding box detections are drawn with unique colors and labeled with confidence scores. The small transient bursts are unknown communications.

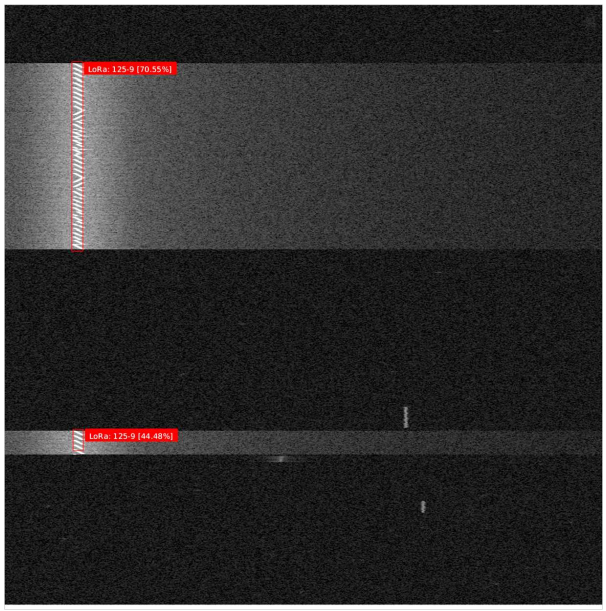

Figure 10. Example detection of lorafarm relay acknowledgement signal. Since the relay is physically located near the receiver, it has significant signal strength. The object detection system was not trained on signals of such high SNR or bursts containing only preamble symbols. The system still succeeds at identifying both anomalies correctly.

[4] Semtech application note AN1200.22 lora modulation basics.

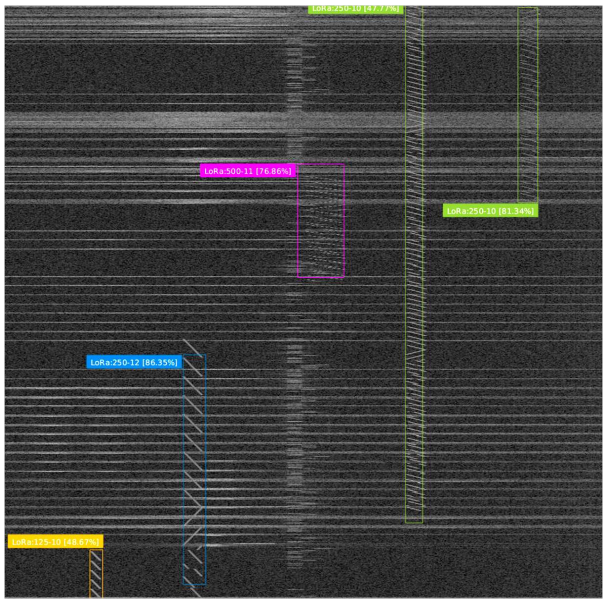

Figure 11. Example detection from simulation set. The background collect is from 783 MHz. Although the (500,11) signal, highlighted in magenta, has a suboptimal visualization (dotted chirp lines), the system can still localize and classify it correctly.

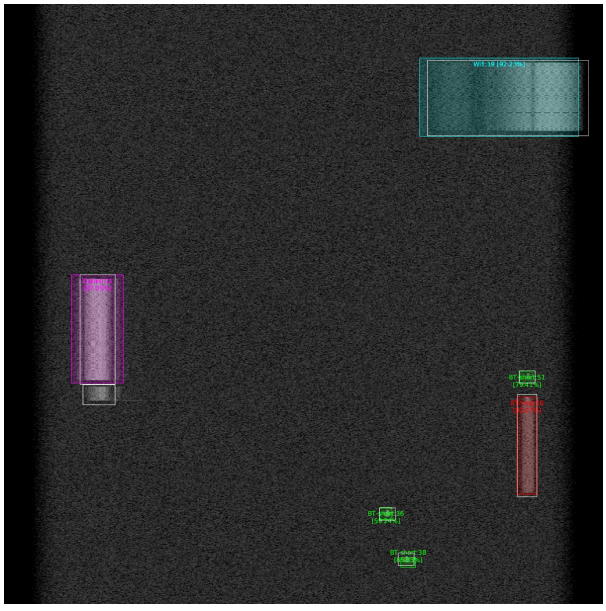

Figure 12. Example detection of Zigbee (magenta), Wi-Fi 802.11b (cyan) and Bluetooth at 2.432 GHz. The Bluetooth class was split into two classes representing short (green) and long (red) payloads. The white bounding boxes are the ground-truth annotations.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[6] J. Xie, C. Ding, W. Li, and C. Cai. Audio-only bird species automated identification method with limited training data based on multi-channel deep convolutional neural networks. *arXiv:1803.01107v1*, 2018.

[7] E. Sprengel, M. Jaggi, Y. Kilcher, and T. Hofmann. Audio based bird species identification using deep learning techniques. *Conference and Labs of the Evaluation Forum*, 2016.

[8] S. Fu, T. Hu, Y. Tsao, and X. Lu. Complex spectogram enhancement by convolutional neural network with multi-metrics learning. *IEEE 27th International Workshop on Machine Learning for Signal Processing*, 2017.

[9] P. Pham, J. Li, J. Szurley, and S. Das. Eventness: object detection of spectrograms for temporal localization of audio events. *arXiv:1712.09668v2*, 2018.

[10] V. Boddapati, A. Petef, J. Rasmusson, and L. Lundberg. Classifying environmental sounds using image recognition networks. *Procedia Computer Science*, 112:2048–2056, 2017.

[11] W. M. Lees, A. Wunderlich, P. Jeavons, P. D. Hale, and M. R. Souryal. Deep learning classification of 3.5 ghz band spectrograms with application to spectrum sensing. *IEEE Transactions on Cognitive Communications and Networking*, 5(2), 2019.

[12] T. O'Shea, T. Roy, and T. Clancy. Learning robust general radio signal detection using computer vision methods. *51st Asilomar Conference on Signals, Systems, and Computers*, 2017.

[13] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 13(1):6517–6525, 2017.

[14] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.Y. Fu, and A.C. Berg. Ssd: Single shot multibox detector. *arXiv:1512.02325v5*, 2016.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv:1311.2524v5*, 2014.

[17] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv:1506.01497v3*, 2016.

[18] Lora alliance lorawan 1.1 specification. 2017.

[19] Semtech application note an1200.22 lora modulation basics rev. 2. 2015.

[20] Lora alliance lorawan 1.1 regional parameters rev b. 2018.

[21] V. Vanhoucke, A. Senior, and M.Z. Mao. Improving the speed of neural networks on cpus. *Deep Learning and Unsupervised Feature Learning Workshop*, 2011.