network

MDPI

Article

An Empirical Study of Deep Learning Models for LED Signal Demodulation in Optical Camera Communication

AbdulHaseeb Ahmed, Sethuraman Trichy Viswanathan, MD Rashed Rahman and Ashwin Ashok *

Department of Computer Science, Georgia State University, Atlanta, GA 30303, USA; aahmed43@student.gsu.edu (A.A.); sethuramantv001@gmail.com (S.T.V.); mrahman19@student.gsu.edu (M.R.R.)
* Correspondence: aashok@gsu.edu

Abstract: Optical camera communication is an emerging technology that enables communication using light beams, where information is modulated through optical transmissions from light-emitting diodes (LEDs). This work conducts empirical studies to identify the feasibility and effectiveness of using deep learning models to improve signal reception in camera communication. The key contributions of this work include the investigation of transfer learning and customization of existing models to demodulate the signals transmitted using a single LED by applying the classification models on the camera frames at the receiver. In addition to investigating deep learning methods for demodulating a single VLC transmission, this work evaluates two real-world use-cases for the integration of deep learning in visual multiple-input multiple-output (MIMO), where transmissions from a LED array are decoded on a camera receiver. This paper presents the empirical evaluation of state-of-the-art deep neural network (DNN) architectures that are traditionally used for computer vision applications for camera communication.

Keywords: visible light communication; deep learning; deep neural networks; optical camera communication; quantization; multi-input multi-output

Citation: Ahmed, A.; Trichy Viswanathan, S.; Rahman, M.R.; Ashok, A. An Empirical Study of Deep Learning Models for LED Signal Demodulation in Optical Camera Communication. *Network* **2021**, *1*, 261–278. https://doi.org/10.3390/network1030016

Academic Editors: Alexey Vinel, Jaume Comellas and Amitava Datta

Received: 19 May 2021
Accepted: 24 August 2021
Published: 27 October 2021

1. Introduction

The use of optical frequencies for communication through the concept of VLC or visible light communication (400–800 THz) has garnered significant interest over the last decade. With the current usage of radio spectrum and data rate demands, it is evident that next generation 6G wireless technology requires bands beyond millimeter wave (mmWave) at 20–60 GHz. VLC bands are one of the most natural bands to integrate into the next generation mainstream mobile wireless technologies. However, even with the state-of-the-art advancements in VLC research and development, e.g., LiFi technology [1] and IEEE 802.15.7 [2] and 802.11 TGbb [3] standardization efforts, the technology does not seem to be completely ready for full-scale deployment due to the fundamental wireless communication system challenges yet to be solved. On a relevant note, solid-state lighting, VLC's sister technology, is finally taking off after three decades of fundamental research [4]. Arguably, VLC also needs several more decades of research to mature as a mainstream wireless technology. On the positive note, VLC research has become quite sizable over the last two decades [5], with increasing activity and a growing number of potential applications including sensing and localization.

The fundamental idea behind VLC systems is to use a light source, such as an light emitting-diode (LED), to transmit the binary representation of data by varying either the intensity or state of the LED (ON or OFF). On the receiving end is an optical detector, which is usually a photodiode that detects photons from the light source and converts it to an electrical current which is mapped, or demodulated, to a binary representation [6]. In this regard, the photodiode serves to demodulate the transmitted data from the LEDs at the receiver end of the system. However, because of the ubiquity and prevalent use of cameras, many VLC systems are now using a camera as the receiver instead of or in parallel to a

photodiode. This configuration is categorized as Optical Camera Communication (OCC) and is being considered as an additional modality for VLC reception [7].

Camera communication finds a plethora of applications, particularly for mobile systems. For example, indoor positioning using cameras [8] significantly improves when using active transmissions from the ceiling LEDs, and vehicles can communicate with one another using built-in brake-light LEDs and cameras [9]. It can also open new opportunities for networking across nodes in vehicular platoons [10]. The key advantage of camera communication is that it doubles as a communication receiver. Cameras equip a lens which provides the receiver a 120–180 (depending on the lens type) FOV. This helps in retaining the signal within line-of-sight (LOS) under mobility. It is possible to multiplex information across multiple (LED) transmitters within the field-of-view (FOV) of the camera. By encoding information across multiple LEDs, the effective data rate can be increased using the spatial dimension and achieve multicast networking simultaneously. However, the key challenge in OCC is the low sampling rate of camera receivers. In OCC, the sampling rate corresponds to the image frame sampling rate or frame rate (FPS: frames-per-second). Video capture or frame-sampling in today's cameras have significantly improved, with a capacity of 240 FPS being available in almost every mobile (smartphone) camera. Cameras with frame rates of the order of 1000 FPS are also available, however, they are very expensive and not necessarily available as embedded devices, unlike mobile cameras. Even with such frame-rate advancements, which help improve traditional photography and video capture quality, the sampling rates are still limited to the order of 0.1–1 KHz. This significantly limits the practical data rates achievable by OCC, and the dependency is heavy on spatial multiplexing in order to improve the data rates. The other challenges in OCC are carried over from those in computer vision, such as impact of lighting conditions (day vs. night, morning vs. evening and sunny vs. cloudy), weather conditions (fog, rain or snow) and mobility (variable perspectives between transmitter and receiver) on the quality of LED signal registered on the camera image.

We posit that given the fact that each data sample in OCC receiver is a camera image frame, frame-rate limitation essentially manifests as a quality reduction problem in computer vision. For example, camera sampling rates can vary slightly during capture and the discrepancies can result in lost signals or partial signals, which manifest as blur on the image. Mapping the signal to a binary 1 or 0 heavily depends on the demodulation algorithm, which is image processing and computer vision tools dependent. In addition, the other challenges in OCC are also carried over from those in computer vision, such as the impact of lighting conditions (day vs. night, morning vs. evening and sunny vs. cloudy), weather conditions (fog, rain or snow) and mobility (variable perspectives between transmitter and receiver) on the quality of LED signal registered on the camera image.

Considering the advances made in the computer vision and image analysis areas with the help of deep neural networks (DNN), in this paper, we hypothesize that DNN models can be helpful in addressing the OCC challenges. In this regard, we present a systematic empirical evaluation of existing state-of-the-art DNN models used for classification problems in computer vision towards demodulation of the LED signal registered on a camera image. The problem statement of the work in this paper is to "classify" whether the (single) LED state as registered on a set of camera image pixels corresponds (demodulated) to a particular quantization level of the symbol (we evaluate two (1 bit), three (1.59 bits) and four (2 bits) levels in this paper). We conduct the evaluations based on the classification accuracy of the signal levels using different state-of-the-art DNN models (with or without our customization). Next, we select the best performing DNN model towards evaluating OCC under a multiple-input (LED array) multiple-output (camera pixels) configuration presented as two case-studies: (CASE STUDY A) communicating one page of text multiplexing using a 2×2 RED color LED array to the camera and (CASE STUDY B) communicating using a diffused BLUE color five LED array to a camera in underwater setting and using diversity (all LEDs transmit the same bits). The overall goal

of the research conducted in this work is to investigate the feasibility and robustness of DNN image object classification models on OCC LED signal demodulation.

The rest of the paper is structured as follows: Section 2—Related Work; Section 3—DNN Based Demodulation Evaluation; Section 4—Case Study A: LED Array MIMO Multiplexing; Section 5—Case Study B: Underwater OCC under MIMO Civersity; Section 6—Conclusions.

2. Related Work

Table 1 summarizes the related work in the area using DNN in PD based VLC systems. In all these works, the notion of DNN usage is primarily as an assistive tool to conduct channel assessment under static conditions. While few works have recently started to explore DNNs for tracking the LED signals in camera communication, the emphasis has largely been to use DNN as an auxiliary assistance for the primary traditional signal processing pipeline in the VLC system. Similarly other researchers have used DNNs to detect the region of interest by using DNN first and then applying other techniques to interpret the LED signal. Such a system was experimented by the authors of [11], who used DNNs to find the region of interest in a vehicle for infrastructure settings.

Table 1. Related works in using DNN for VLC systems.

Ref.	Year	Area of Focus
Langer et al. [12]	2007	Using DNN for photodiode (PD) based VLC
Goodfellow et al. [13]	2016	End–end learning using autoencoders
Pachpande et al. [14]	2018	Improving signal detection in quadrature amplitude modulation
Hoon Lee et al. [15]	2018	Using DL for improved binary signal reception
Wang et al. [16]	2018	Reinforcement learning for a hybrid camera communication system
Wang et al. [17–19]	2019	Focused on learning the channel conditions and improve indoor positioning
He et al. [20]	2019	Remove impact of ambient noise
Chi et al. [21,22]	2019	Improve receiver demodulation
Miao et al. [23]	2019	Non-linearity mitigation in OFDM
Turan et al. [24]	2020	Vehicle–vehicle VLC channel modeling
Xi Wu et al. [25]	2020	Channel estimation in camera communication
Pham et al. [26]	2020	Using DL for vehicular camera communication

Other research that utilizes DL models with VLC systems includes the work of [14], who used autoencoders to mitigate the LED nonlinearity for orthogonal frequency division multiplexing (OFDM)-based VLC systems. The authors performed simulations of adjusting the field of view from 70 degrees to tilts of 5 degrees. Their model is trained to minimize the error in the recovery of the transmitted data. The closest research related to this study was that of [20], where the authors used Convolutional Neural Networks to filter out ambient light from a VLC receiver. Their proposed system showed an improved BER by 40% compared to a system with the same signal-to-noise ratio. The authors of [21] summarized some of the other areas of VLC research where ML/DL models are being incorporated. They described that there are four main applications of ML/DL models in VLC systems, which include non-linearity mitigation, jitter mitigation, modulation discrimination and phase estimation. Models and algorithms such as K-means, Support Vector Machines, logistic regression and DNNs [27–29] can all be used to perform these tasks. In [30], the authors leverage a 2D CNN for data recovery; however, this was carried out by learning features between color channels and neighboring symbols in the rolling shutter based OCC,

which was prone to errors in real time systems with wide variations in lighting conditions. Thus, there is no clear evaluation of whether and how DNNs are suitable in a VLC receiver, particularly for demodulating signals in OCC. The dearth of publicly shared datasets in VLC adds to the challenges in advancing the explorations of using deep learning for VLC. This paper essentially presents an undertaking of investigating the suitability of deep learning classification algorithms for demodulation in OCC and makes a dataset available to the community for further studies in this space.

The key challenge in using traditional non-ML computer vision techniques is that these techniques depend on feature extraction from the object's region of interest. However, LED is not a feature-rich "object". Feature extraction techniques typically fail to locate any feature within the LED region as the LED visually is just a bright blob with no clear distinct geometric or photometric characteristics. While some detectors such as ORB (state-of-the-art) [31] are able to locate some features along the perimeter of the LED (provided that the LED circular perimeter is clearly visible), the same features are also detected in other non-LED pixels in the scene (see Figure 1 for an example). The fact that the LED region is not clearly detected makes the demodulation process very challenging, as the pixels used for differentiating a ON pixel from OFF pixels are inherently very noisy. In such cases, an optimal threshold for intensity detection is impossible without restrictions (e.g., only static settings with fixed distances).

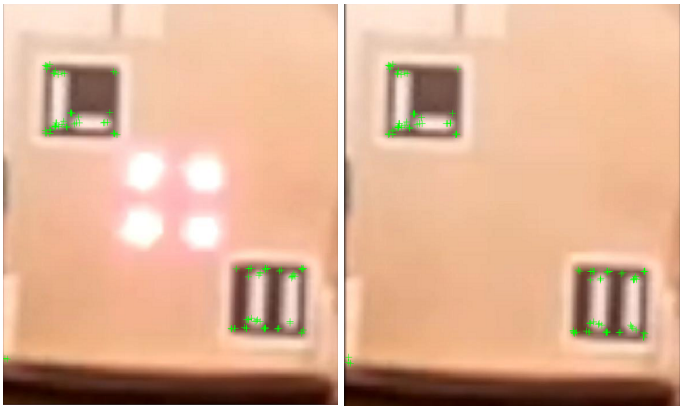

Figure 1. Features extracted using ORB techniques in both LED ON (**left image**) and LED OFF (**right image**) states; no key points can be found on LED regions.

3. DNN-Based Demodulation Evaluation

We evaluate the performance of different state-of-the-art DNN models towards classifying or demodulating the LED signal state from camera sampled images in OCC. In particular, we consider two levels (binary), three levels and four levels of LED intensity quantization levels. The evaluation follows the following key objectives:

1. Evaluate the performance of different supervised learning DNN models under transfer learning. We consider VGG16, Resnet50 and InceptionV3 deep learning architectures that are state-of-the-art for vision object classification.
2. Evaluate the performance of a simple customized deep learning architecture that treats LED as a new class. We consider a convolutional neural network (CNN) architecture for this evaluation.
3. Compare the DNN results with basic thresholding-based demodulation of LED intensity levels.

Figure 2 illustrates the LED setup in which a Raspberry Pi was used to control a red LED and LED signals were recorded on a handheld smartphone device. In particular, we used a solid-state CREE red LED with 1 W optical power output. We used a Google Pixel Android smartphone as the camera device. The data consisted of frames from camera capture videos of the LED operated at 15 Hz and captured by a camera recording at 30 frames per second (FPS). Each video was recorded for approximately 15 s and

at distances ranging from 1 to 7 m in increments of 1 m. The original frame size was 1920 × 1080 × 3 pixels, which was cropped to 128 × 128 × 3 and 32 × 32 × 3 pixels for the experiments. For the remainder of the study, 128 × 128 × 3 frames represent frames of the LED with some background and are, thus, called full frames, and frames of size 32 × 32 × 3 represent cropped frames of only the LED and are, thus, called cropped frames. The data were labeled into three different datasets, one for the two level case, ON or OFF; one for three level case, Fully ON-Slightly ON-Fully OFF; and one for four level case, Fully ON-Slightly dimmer than Fully ON-Slightly ON-Fully OFF. Table 2 illustrates what the data looked like before and after data augmentation in order to ensure that there is no bias in the dataset, and that the sampled images dataset is representative of different orientations. The dataset includes 4-level, 3-level and 2-level data. Figure 3 depicts how four level data were labeled and what each intensity/class was at each distance. The blur effects are seen as the region shown is cropped from the original image and resized to 32 × 32 resolution.

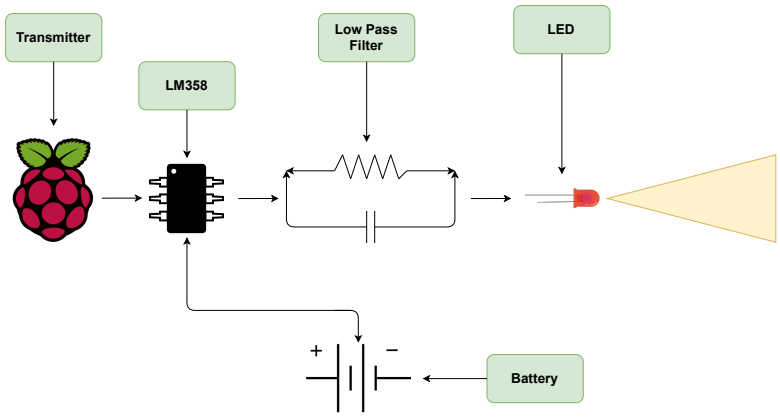

Figure 2. LED Transmitter Setup.

Table 2. Number of Data Points For Each LED Transmit Level in the Dataset.

Data Level	Raw Frames per Class Label	Balanced Frames per Class Label
4 Level	0 s = 406	0 s = 1624
	1 s = 309	1 s = 1545
	2 s = 940	2 s = 1540
	3 s = 1545	3 s = 1545
	Total = 3200	Total = 6254
3 Level	0 s = 406	0 s = 2436
	1 s = 309	2 s = 2472
	2 s = 2485	2 s = 2485
	Total = 3200	Total = 7393
2 Level	0 s = 715	0 s = 2860
	1 s = 2485	2 s = 2795
	Total = 3200	Total = 5654

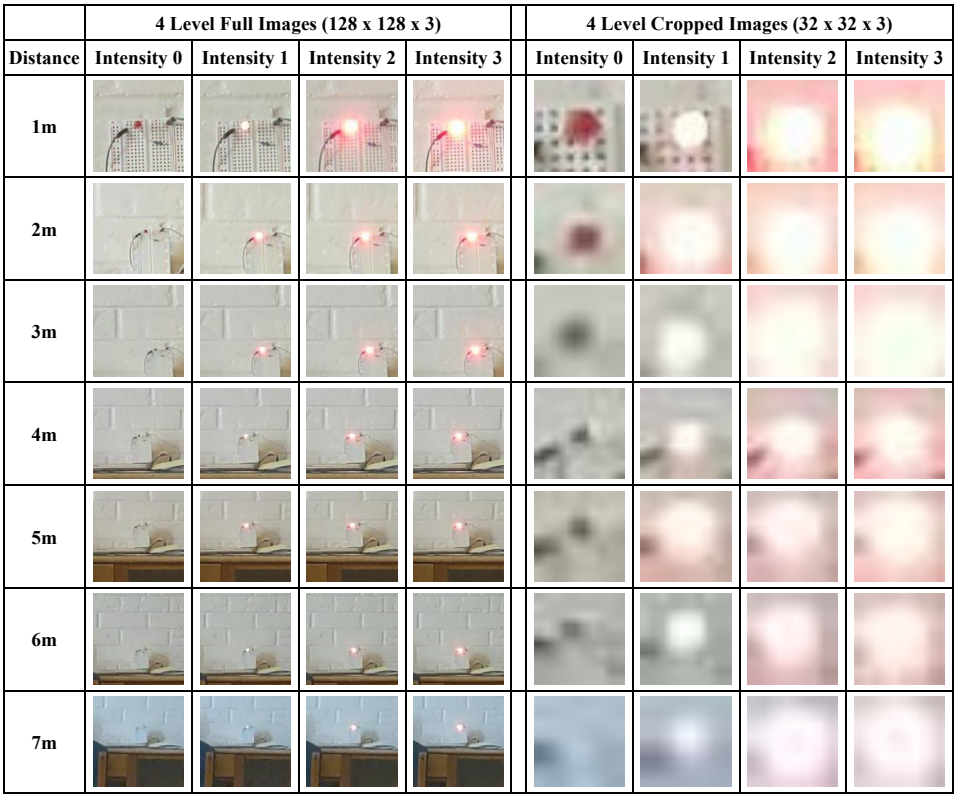

Figure 3. Four level data representation.

3.1. Basic Thresholding Based Approach

The basic thresholding based approach was conducted to determine if a classical approach to determining the LED state from the frame was any better than the classification of the DNN. In doing so, the average pixel intensity of each class was determined, and then that value was used to classify all images in the dataset. Due to the different distances at which the LED was captured, in addition to the fact that the video was captured via mobile camera, there was slight light change in some of the frames; thus, instead of a hard value for the average pixel intensity of each class, a range was established. In addition, three types of croppings of each frame were evaluated: (1) LED with some background—noted as "RGB with Background"; (2) LED cropped out and resized to $32 \times 32 \times 3$—noted as "RGB cropped/Resized"; (3) LED cropped and not resized—noted as "RGB Cropped". The results of the thresholding experiment can be found in Table 3.

Table 3. Average basic thresholding-based approach results.

	RGB With Background	RGB Cropped/Resized	RGB Cropped
2 Level	97.37%	97.20%	97.10%
3 Level	96.90%	97.18%	96.98%
4 Level	90.41%	83.11%	83.17%

3.2. Evaluation Methodology

Four different supervised DNN models were implemented and evaluated on the collected LED data. These supervised models included the VGG16 model, Resnet50 model, Inception V3 model and a simple three layer convolution model. The reason why the first three models were chosen was because these models have demonstrated good performance in many image classification problems. The first three supervised DNN models were utilized as a transfer model and had all their previous layers frozen and

the last three dense layers removed. This was then followed by adding three new dense layers and two new dropout layers to the end of each of these models. For reference, these models were implemented with the Keras library, and the pre-trained weights were from when these models were trained on the ImageNet database [32]. Figure 4 illustrates the graphical representation of the VGG16 model, Resnet50 model and Inception V3 model architecture with the appended layers. Although the input of this network is a 128 × 128 × 3 frame, which represents the full frames, this same architecture was applied for the 32 × 32 × 3 frames. Therefore, only two types of frames were evaluated with the DNN models: the full-frame, which was a 128 × 128 × 3 image, and a cropped version, which was resized to 32 × 32 × 3. This is different from the basic thresholding-based approach in which cropped frames that were not resized to 32 × 32 × 3 were also evaluated. However, cropped frames used for the Inception V3 model were resized from the 32 × 32 × 3 cropped versions to 75 × 75 × 3 because that was the smallest image size that the Keras library allowed when using their Inception V3 model as a transfer model [32].

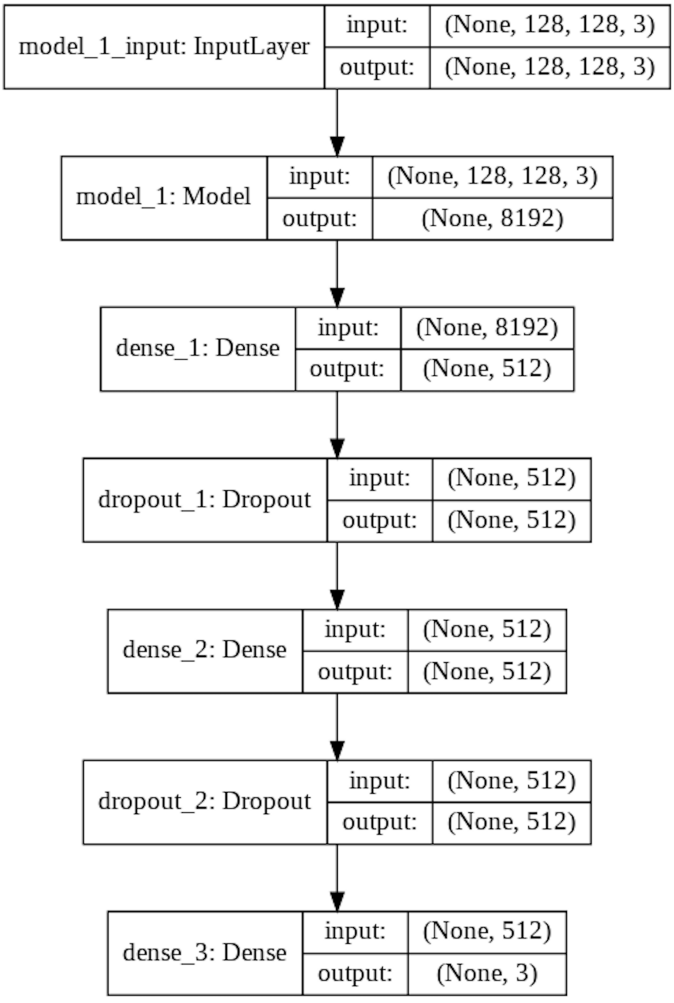

Figure 4. Appended layers for the deep neural networks.

As mentioned earlier, only the appended layers of the transfer models were trainable because the previous layers for the VGG16 model, Resnet50 model and Inception V3 model were all frozen. As for the simple three layer convolutional model, it had all its layers set to trainable. Each time the model is compiled, the trainable layers received a new set of weights. The new weights affect the manner the model performs because of how the gradient adjusts the parameters of the model as the back propagates. This is why, if upon compile-time a "good" set of weights are acquired, the model can quickly learn the features, and if the weights are "bad" then it takes longer for the model to figure out the

features. However, because the weights are randomly chosen at the compile-time, there is no guarantee of "good" weights. In order to circumvent this, each model was compiled and ran three times to obtain an average. For each model, both Softmax and Sigmoid activations were experimented with as the last activation function for the output layer. In addition, three different optimizers and three different learning rates were explored. The optimizers included RMSprop, SGD and Adam and for the learning rate 0.001, 0.0001 and 0.00001 were used. The F1 score was also taken to obtain a more intuitive understanding of the model's performance. Each model was trained for 10 epochs and dropout was set to 0.2, although other dropout values were used to reduce overfitting of some models. However, the results presented later are for a dropout set to 0.2. Each of the data levels had 500 frames set aside for validation and 500 frames set aside for testing; the rest were used for training. This was performed for the 4 Level, 3 Level and 2 Level balanced datasets.

3.3. Two Level Supervised Model Analysis

From the results presented in Table 4, it is clear that, for the most part, the VGG16 model performed best on 2 Level, full-color data, followed by the 3 layer convolution model. The results for the VGG16 model show that the accuracy was nearing 99% consistently and was well above 95% with the different frame types and activation function for the output layer. In terms of which activation function performed best in the output layer, it is clear that the Softmax function performed better than the Sigmoid function, across all model types, although technically in a binary case, Sigmoid is just a special case of Softmax. In terms of what type of frame performed better between full frames and cropped frames, the full frames evaluated better. The reason why this might be the case is that the full frames distinguished the LED clearly, whereas the cropped frames were completely polarized by the LED. Therefore, the LED was more clearly segregated in the image, and its feature was, therefore, more easily learnable. However, further exploration into what the models are actually learning in each layer is needed to verify this intuition. Considering that the common bit error rates (BER) for data communication systems are 99×10^{-13}% or greater, it would be more appealing to observe results closer to 99.999% and above. However, considering that the models achieved this level of accuracy with only 10 epochs of training and less than 10,000 frames of training data, this begs the question of what the potential could be. By tweaking the hyperparameters and other parameters and perhaps training the model for a longer duration than 10 epochs, as well as training on a larger and more pristine dataset, it might allow the BER to be closer to the ideal case.

Table 4. 2 Level Full Color Model Evaluations.

	Softmax Full Image	**Softmax Cropped Image**	**Sigmoid Full Image**	**Sigmoid Cropped Image**
VGG16	98.89%	98.69%	97.99%	96.60%
Resnet50	95.60%	68.20%	81.00%	57.00%
Inception V3	76.80%	51.23%	63.30%	39.34%
3 Layer Convolution	97.56%	93.24%	94.20%	99.00%

3.4. Three Level Supervised Model Analysis

From the results presented in Table 5, it is clear that the VGG16 model performed best with accuracy well above 95% for both cropped and full-sized frames and for both Softmax and Sigmoid activations in the last layer of the VGG16 model. As for the second-best model, it is the three layer convolution model with an accuracy well above 90% and below 95%. As a whole, full-size frames performed better than cropped frames, especially with Softmax as the activation function in the last layer. In comparison to 2 Level full-sized frames, the results for the 3 Level full-color frame model evaluations were much less accurate than the 2 level data. This was actually something that was expected because as quantization of the system increases, the risk of mistakes in interpreting the received signal also increases.

However, 3 level data seem promising given a more refined model with, perhaps, a large training session and larger training data size.

Table 5. 3 Level Full Color Model Evaluations.

	Softmax Full Image	**Softmax Cropped Image**	**Sigmoid Full Image**	**Sigmoid Cropped Image**
VGG16	98.00%	96.79%	95.99%	96.60%
Resnet50	77.60%	53.20%	79.00%	50.00%
Inception V3	42.80%	32.23%	31.60%	32.82%
3 Layer Convolution	94.99%	94.40%	91.20%	93.00%

3.5. Four Level Supervised Model Analysis

From the results presented in Table 6, it is clear that the VGG16 model again performed the best, however, the VGG16 model only performed over 90% when the activation function of the output layer of the model was set to Softmax. Interestingly, the pattern of the VGG16 model performing the best regardless of the activation function of the output layer did not hold for this dataset as it did with the previous two datasets. There was about a 20% reduction in accuracy when Sigmoid was used as the activation function for the output layer of the VGG16 model, whereas for the two previous datasets, which are the 2 Level and 3 Level full-color frame datasets, the accuracy reduction was roughly 5% or so between Softmax and Sigmoid. This is probably because Sigmoid is not meant to handle multiclass classification. Similar to 3 Level and 2 Level data, the second-best model was the three layer convolution model. Moreover, similar to 3 level and 2 level model evaluation, full frames did better than cropped frames, with the exception of the 3 layer convolution. Similarly to 2 Level and 3 Level data, Softmax in the output layer performed better than Sigmoid in the output layer on average across all models. In terms of quantization, 4 level data will need to be further reviewed and refined in order to guarantee adequate data transmission.

Table 6. 4 Level Full Color Model Evaluations.

	Softmax Full Image	**Softmax Cropped Image**	**Sigmoid Full Image**	**Sigmoid Cropped Image**
VGG16	93.50%	95.99%	75.59%	73.79%
Resnet50	69.99%	35.60%	66.00%	34.40%
Inception V3	37.59%	35.19%	28.00%	23.00%
3 Layer Convolution	50.59%	75.00%	70.99%	74.59%

3.6. Supervised Model Analysis Summary

As for all the frame results as a collective, 2 Level data performed the best, followed by 3 Level data and followed by 4 Level data. There are several reasons for this but the one that is the most compelling is that the intensity level between some class labels might have been too minute that it was difficult for the model to distinguish between them. As for Resnet50 and Inception V3 models, these two models performed pretty poorly across all three level types. One thing to note which is not described in the tables above is that the training accuracy for these two models would consistently reach an accuracy of 90+% and validation/test set accuracy of less than 40%. Basically, these models were overfitting more often than not. As mentioned earlier, these models all had their dropout layers set to 0.2, i.e., "drop" 20% of the neurons in the layer. However, after collecting the results above, a sample of dropout values were used for both Inception V3 and Resnet50, which showed that the model stopped overfitting as dropout increased, but the validation/test set accuracy remained the same, which is why those results are not presented here. Another intuition/idea that explains the results for the Inception V3 model and Resnet50 model's

poor classification results is that the size of the LED dataset was too small as these models require a lot of data to learn the granularities in the data distribution, especially in the case of Inception V3 which uses adaptive learning via filter sizes. Perhaps having a larger a dataset might improve these model's performance.

In an ideal case, the three major DNNs should have performed well above 90% on the validation/test set. In addition, the least accurate model should have been the simple three layer convolution as it did not have any special techniques nor was it a deep model. In any case, if the Resnet50 and Inception V3 model are disregarded, which were hypothesized to have failed because of lack of data, the ideal case holds for VGG16 and the simple three layer convolution. With that in mind, it can also be said that in terms of neural network theory, the Resnet50 model should have performed the best as it is the deepest. This would be followed by Inception V3 and then the VGG16 model and lastly the three layer model. This scenario occurs only if the criterion for success is based on how deep the network is. Another point to make is that perhaps the VGG16 model performed well because of its unique architecture wherein only 3×3 filters are used and the fact that more non-lienar activations are used per step in the model. This would actually be pivotal because the data represents concentrated areas within the image. Thus, extracting the most relevant information would mean that a small receptive field would be best to employ.

In terms of how well the DNN supervised models performed compared to the basic thresholding based approach, it is somewhat unclear. The threshold method evaluated each distance, whereas the models evaluated all distances at once. However, on a more general scale, the full color frame model classification seemed to be on par with the thresholding method for full color frames. Another intuition that seemed to fail was that the models, when evaluating the cropped images, should have been more robust towards the LED not being centered than the threshold method, yet they still performed worse.

Concerning quantization, the experiments conducted show that more symbols in data transmission are possible when the receiver is a camera; however, the models have to be trained better or entirely new models need to be used. This is because, as the number of symbols increases, the accuracy decreases. In addition, better resolution frames that can distinctly separate the different intensities of the LED need to be used. Without actually being able to separate the intensities, it is not possible to perform quantization in VLC systems when the receiver is a camera. Up to this point, only two, three and four symbols were experimented with, and the accuracy already dipped about 5–10% when the the number of symbols increased from two to four levels. Therefore, obtaining symbols of up to 255 or a data transmission rate of 2^8 would be impossible without major modifications to the current model performances.

4. Case Study A: LED Array MIMO Multiplexing

The next phase of this study is to empirically evaluate proof of concept of DNN-based OCC demodulation. In this regard, the first case study involves studying the performance of DNN demodulation of VGG16 for a 2×2 LED array under multiplexing configuration. We consider packetized communication to conform with traditional communication modalities.

Figure 5 illustrates the experiment setup. As can be observed, a 2×2 LED grid was used with spacing of 3 inches in the vertical and horizontal direction and a Raspberry Pi 4 was used to operate the LEDs. The receiver was a Raspberry PiCamera V2 which was controlled by the same Raspberry Pi as the LEDs. The top left and bottom right markers are ARUCO markers, which are synthetic square marker similar to QR codes but for object detection and location [33]. They were used to consistently locate the LEDs in each frame captured by the camera. The LEDs operated at 30 Hz and the camera recorded at 60 FPS, following the Nyquist criterion. However, the exact synchronization of the system was inaccurate and caused problems when transmitting a large volume of data in the order of 20 packets or more. This was primarily because the OS running on the Raspberry Pi was not a time-critical (Real-Time) OS, which meant that the accuracy of the operations could

be variable to a degree and, therefore, causes synchronization issues between when the LED flashed and when the camera captured a frame.

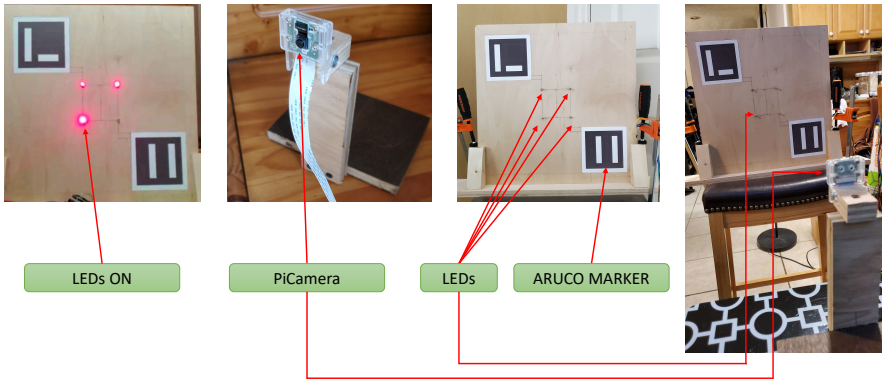

Figure 5. A 2 × 2 LED—Camera Communication MIMO Setup.

Figure 6 illustrates what each packet looked like. In addition, the transmitter and receiver software files were both written in python and ran from the Raspberry Pi 4 python IDLE. The transmitter processing and receiver processing for analysis were conducted as follows.

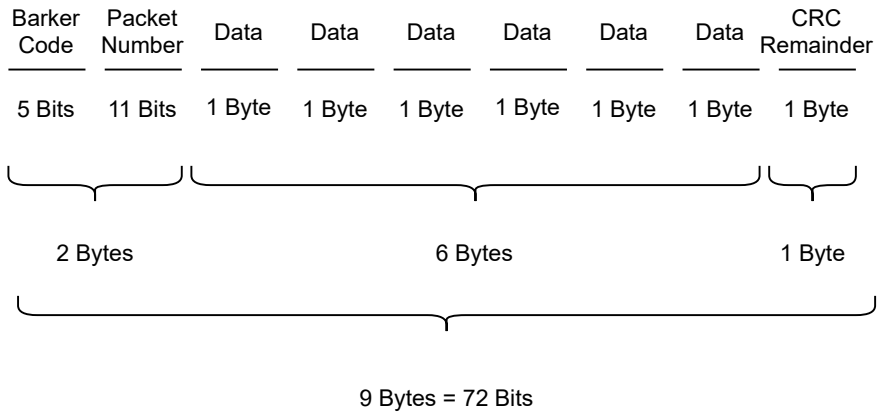

Figure 6. Packet Structure.

4.1. Transmitter

Stage 1. Prepare data: After obtaining the text file, all the Non-ASCII values are removed. Then, each of the characters are converted to 8-bit binary data which are appended with starting and ending marker bits.

Stage 2. Assign each LED data: 48 bits from the binary data for each LED (4 transmitters, thus, in total 192 bits) were parsed and a 11-bit Packet Number is assigned to each chunk of 48 bits of data. The 8-bit frame check sequence is computed over the Packet Number and data and repeated until there is no more input data available.

Stage 3. Packet formation: To generate the packet, 5-bit barker codes are appended at the beginning of each packet and then the 11-bit Packet and 6 bytes of data were added. Finally, the 8-bit frame check sequence is also added and stored packets with the associated LEDs.

4.2. Receiver

Stage 1. Preprocess received transmission: After detecting and localizing the ARUCO markers, each of the four LEDs is cropped into separate frames to be used as input to the DNN model for classifying the LED state. Each of the classifications are stored separately, and

noise and outliers are removed. Then, we need to find the first occurrence of a bit equal to one in the predicted list for each LED, and all the previous bits in the predicted list are removed.

Stage 2. Process packet metadata for each LED: From the pre-processed data, the Barker index needs discovered and parsed with classified packets in each LED List. Then, Packet Number bits are checked along with CRC remainder calculation to verify transmission correctness. The parsed data bytes from the packet are stored in memory.

Stage 3: Data demodulation: In this final demodulation step, data bytes are combined from each LED list in order and compared with every 6 bytes of ground truth. Every byte error in every packet is counted and reported as packets with the Barker code Error, Packet Number Error, Data Error and CRC Errors. Finally, the starting and ending markers are removed to convert every 6 bytes of data to ASCII representation.

4.3. Experimentation

Due to the underlying hardware issues, a new methodology had to be taken to show that the system, specifically the model, is capable of working even with the hardware issues. To showcase this, the following steps were taken:

- Step 1: Find the number of characters, which will be called the chunk size for the remainder of the study, that can be successfully transmitted without any type of error (Camera or LED). As mentioned earlier, for shorter transmission the hardware inconsistencies are not prevalent;
- Step 2: Transmit a 1000 word file in a single transmission. This was the original/ideal target for the empirical study to transmit with as few errors as possible;
- Step 3: Evaluate the 1000 word file transmission by dividing it into chunks established from Step 1 and evaluate each chunk;
- Step 4: Transmit the 1000 word file in several videos in which each video sends one chunk of data where the chunk size is defined by the the steps used in Step 1;
- Step 5: Analyze the results.

The reason why the chunk size is established is to showcase the effectiveness of the model as independently as possible, rather than with the entire system. Similarly the reason why a single transmission of a 1000 word file is compared versus a fragmented transmission of a 1000 word file is, again, to showcase the models performance but, more importantly, to illustrate how the hardware issues can cause serious problem for this communication system.

To achieve Step 1, several videos of varying character length were transmitted at each distance. After computing the results, 100 character transmissions were found to be the best chunk size. One hundred character transmissions were roughly 17 packets; 18 packets if the starting and ending markers are included. The results showed that across five trials at each distance from 1 m to 7 m, there are no hardware issues nor model misclassifications except for when the distance is at 7 m. This is when a few model misclassifications start to appear. Out of the nearly 6200 frames collected across the five trials at a distance of 7 m, there were only 16 model misclassifications. this illustrates another point which is that as the LED moves farther and farther away, the model is susceptible to more misclassifications for multiple reasons, some of which include the resolution of the image becoming too degraded.

Steps 2 and 3 was performed by transmitting the 1000 word file at each distance twice. Then, every 17 packets, which is roughly equal to 100 characters which equals 1 chunk, were analyzed for byte errors only for the data and not for the packet metadata. In the 1000 word file, there were about 50 chunks or about 850 packets worth of data. After demodulating the transmission, the 50 chunks were checked for the total number of packets with any number of errors, and the Cumulative Average Packet Error Rate was also calculated. The reason why the cumulative average was used was because, as will be discussed later, the hardware issues would pop up randomly during the transmission and distort the transmission. Thus, to obtain an intuitive idea of the success of the entire transmission, the cumulative average was utilized. The formula for the Cumulative Average Packet Error

Rate is given in Equation (1). Then, the average of both the number of packet errors and Cumulative Average Packet Error Rate between the two trials was calculated.

$$PER(chunk\ number\ N) = \frac{\#\ of\ packets\ in\ error\ for\ N\ chunks}{total\ number\ of\ packets\ in\ N\ chunks} \quad (1)$$

For Step 4, instead of transmitting a single 1000 word file in 1 video, the 1000 word file was sliced such that approximately 100 characters were transmitted in each video. This resulted in having to transmit 50 videos to obtain the entire 1000 word file. This was performed twice at each distance and then the same procedure was used to calculate the total number of packet errors per chunk (in this case, 1 video), and the Cumulative Average Packet Error Rate was calculated with the average between the two trials that were also included.

Overall, we observed that the PER for the 1000 word file transmission was close to 15% at its maximum and about 10% on average. To understand if this is a model error or a hardware error, analysis has to be performed packet by packet. In doing so, to check for hardware error, the transmission was checked for the time difference between consecutive captured frames. This is where the problem actually lay. For example, in one particular transmission, there were actually eight frames where the difference between it and the previous frame was well above 16.666 milliseconds, i.e., below 30 Hz. These eight locations had the following time difference between themselves and the previous frame: 697.68, 1079.739, 33.224, 249.171, 33.223, 1046.517, 531.565 and 647.844 in milliseconds. As it can be observed, some of these values correspond to whole seconds between frame captures, which has the effect of distorting the received data by shifting the data bits and losing whole chunks of data. This is why the received text is actually less than the actual text. The way to correlate these time difference errors back to the location in the message is by correlating the frame number of where the time difference error occurred relative to the chunk that it occurred in. Basically, the following procedure is performed: find the first time difference error-index and then multiply the index of the error by four for the number of LEDs, then divide by 72 for the number bits in each packet and then multiple by 6, which represents the number of bytes in each packet. The returning value should be equal, or close, to the number of characters that are transmitted correctly before the sequence of garbage text begins. In the current case, the previous calculation yields 520 characters correctly transmitted considering the index of the first error is 1559th frame. This is about the same as in the demodulated text. Once the frame capture error occurs, the rest of the transmission is out of sync, which is why the demodulation returns garbage characters. What this proves is that the major issue with this system is the hardware issue, specifically the camera not capturing at a regular interval.

To confirm that the issue is truly a hardware issue and not a model issue, the transmission of the 1000 word file was broken into chunks of 100 characters, and then one video at a time was transmitted at about 100 characters each. This produced 50 videos each transmitting about 100 characters. Presented here is just the result of the videos captured at a distance of 1 m. When the data to transfer is small, the average PER reduced to 1% and lower. However, there were a few cases where the PER was higher (5% at maximum) in comparison to the rest of the transmission. To understand if these are a model issue or a hardware issue, the consecutive frame time difference should be checked. In doing, it is discovered that there are camera issues present for some of the chunks and not all. For instance, in the case of Trial 1, chunks 1 and 12 had camera issues near the end, which is why both had one packet error each. However, the other chunks did not have camera issues, which means that these were probably caused by model misclassifying.

The same trend in both the 1000 word single transmission and 1000 word multi-transmission can be observed in the rest of the distances from 2 m to 7 m. However, as distance increases, the model misclassifications increase. However, by comparing the one large transmission versus the 50 multi transmissions, it is clear that the model error is secondary to the hardware issue.

5. Case Study B: Underwater OCC under MIMO Diversity

In regards to future exploration, another avenue that this research study is beginning to explore is in the use case of a camera-based VLC system in an underwater setting. The practical application would be related to network communication between different submarines and boat to a submarine, etc.

The following is an evaluation of the underwater LED data with a DNN. The data that were obtained was of an LED, blue in this case, operating at 50 Hz for about 17 min while a stereo camera operated at 100 FPS to capture the LED signal. The transmission was not meant to carry any information but rather only conducted to capture the LED blinking to create a repository of data for DNN training and analysis. The LED used was an LED bar that consisted of six individual LEDs rather than a single or grid of LEDs. The camera in this setup is Stereolab's ZED-2 stereo camera which achieves 100 FPS at VGA (640 $\times$ 480) pixels resolution. The distance between the camera and LED was 4.5 m.

For purposes of analyzing and using the data, the first step was to extract all the frames from the video and then crop the frames so that only the LED was within the frame. This is because the region of interest in this study has been identifying the LED state, i.e., ON or OFF. Since the camera is a stereo camera, within each frame are two images of the LED, as shown in Figure 7. Basically, each image is 1280 $\times$ 480 pixels, but there are two frames within each image of size 640 $\times$ 480 pixels. Therefore, in order to increase the number of samples available to train and validate later, the LED in each subframe of the raw image was cropped and stored separately. After cropping the LEDs from each frame, it was resized to 32 $\times$ 32 $\times$ 3 because that was the input dimension size set for the DNNs used in this research. Figure 8 shows an example of the LED cropped and resized. In total, there were 199,716 raw frames available for analysis, where half were from the left LED in the image and the other half were from the right LED in the image. Next, the images of the cropped LEDs were labeled, however, due to the time constraint, instead of manually labeling the images by hand, the maximum pixel intensity of the image was used as a measure to determine if the LED was ON or OFF.

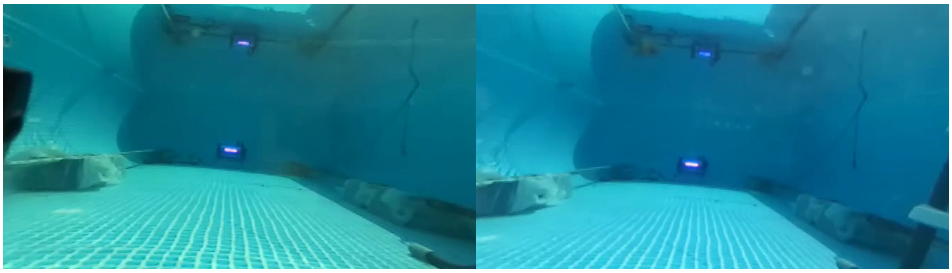

Figure 7. Zed Camera Frame.

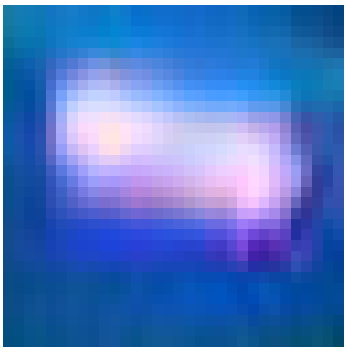

Figure 8. Cropped LED From Zed Frame.

After the raw frames were both extracted, cropped, resized and labeled, 10% of the dataset was randomly chosen and upsampled via augmentation. In total, 219,516 frames were valuable to validate and train with. After all the frames were preprocessed, two

evaluations were conducted: (1) performance of the empirical study model against this new dataset and (2) a newly trained model on this dataset, which is validated against data from the underwater dataset. To evaluate the performance of the MIMO model against this new dataset, all the processed frames were given to the model, predicted by the model and then cross-checked with the ground truth to observe how well the model performed. This analysis was conducted to observe how transferable the model was from one application/environment to another. Figure 9 shows the confusion matrix of the MIMO model validation results against the underwater LED dataset. Overall the model did no better than a flip of a coin with an overall accuracy of approximately 58.31%, which illustrates that the model is data specific.

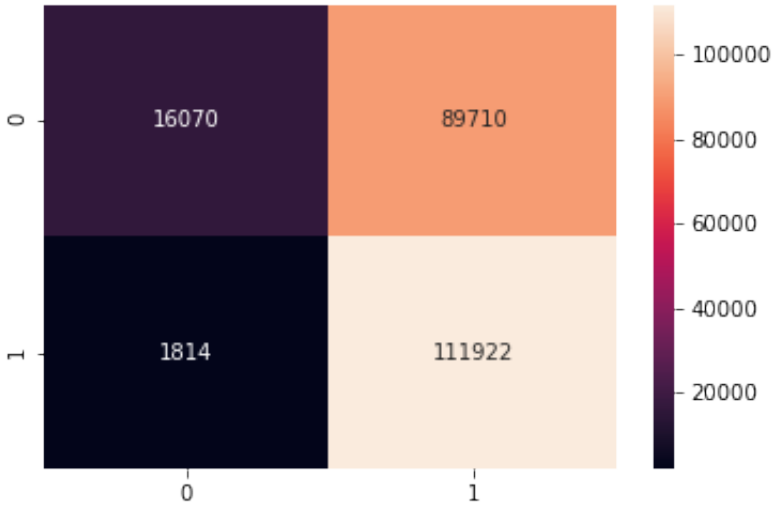

Figure 9. MIMO Model validated against the underwater LED dataset.

Next, the same model, i.e., the same architecture and same parameters/hyperparameters, that was used in the empirical study was trained on just the underwater LED dataset and then validated. This was performed to investigate if a model trained on the specific data performed better with the new data. Therefore, 60% of the underwater LED dataset was used for training the model, and the other 40% was used for validating the dataset. Similar to the model from the empirical study, the model was trained for 50 epochs with all the same parameters and hyperparameters, i.e., no hyperparameter searching. Refer to Figure 4 for details of the DNN architecture. After training the model, the model was validated against the remaining 40% of the dataset and the results are presented in Figure 10. As can be observed, when the model trains on the specific dataset that it will be validated against, it performs much better with an overall accuracy of approximately 99.68%.

With that being said, the conclusions that can be drawn here are that the DNN is primarily data specific and does not transfer over to a different environment or object, i.e., the LED type very well. However, the model architecture with its parameters is capable of being trained on a new dataset very well and is able to perform very well on the new specific data. As this was just to bridge a new area in camera-based VLC systems that incorporate DNNs, a lot more research studies and analyses are required to draw definite conclusions, but the main purpose was to expose the reader to another application of DNNs in a camera based VLC system.

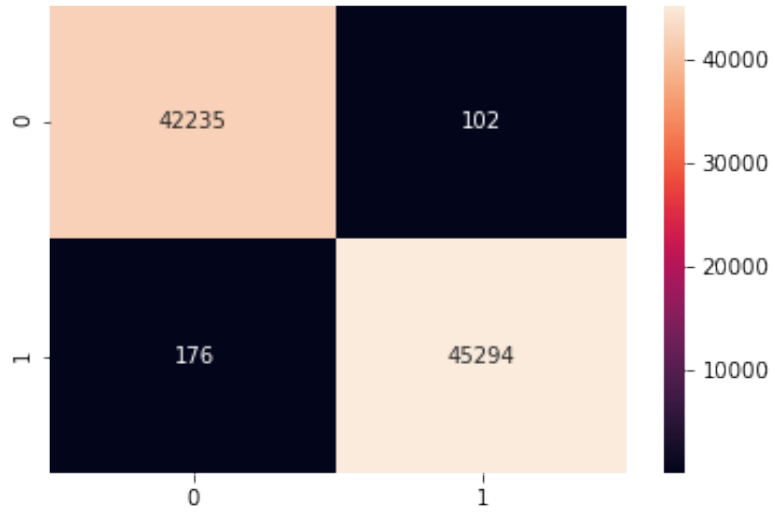

Figure 10. Trained model validated against hold out set.

6. Conclusions

The key insights gained from this empirical study include the following: (a) It is feasible to use DNN for demodulation in optical camera communication, however, the robustness of the model is largely dependent on quality, size and diversity of the dataset. DNN cannot easily extract subtle features from LED ON/OFF signals as they appear as blobs with minimal or no structural feature. Novel spatio-temporal techniques that study the LED signal reception variability across space and time must be incorporated to build novel DNN models for VLC. Hardware inconsistencies (b) in the camera and computing unit (e.g., Raspberry Pi version) can severely impact the performance of DNN demodulation as transmission and reception in VLC are largely incoherent (transmission and receiver samplings are not synchronized). Identifying the exact head and tail of each packet and the detection/estimation of the missed camera frame (samples) are key for improving demodulation accuracy.

In conclusion, this study has provided a baseline for further work in incorporating DNNs for camera based VLC system that serve as a demodulation mechanism of the transmitted data from the LED. This study has evaluated the performance of various DL/ML models to explore the potential of these models in deciphering an LED state from a regular binary state up to four levels of intensity. Then, to verify these model's performance, a trivial method to solving the same problem was explored to compare the model's performance. This study showed that, with minimal training, the models are able to achieve between 95% to 99% accuracy in classifying the LED intensity/state with respect to the number of intensities. The empirical evaluations using a prototype OCC system showed that DNNs are indeed a very useful for demodulation but require an understanding of what the model is learning in order to effectively create a robust system. Hardware issues presented a major challenge and illustrated the need for specific hardware for this type of communication network. The results from the empirical study demonstrate the effective use of DNNs but only in a set environment; therefore, a dynamic setting requires further model training and perhaps large datasets, etc. Through the course of this study, we have addressed feasibility questions for DNN usage in camera communication. However, this work has opened up new avenues for research questions across the design, development and implementation of novel DNN models and DNN-inspired algorithms for improving receiver performance in camera communication.

Author Contributions: Conceptualization and methodology, A.A. (AbdulHaseeb Ahmed) and A.A. (Ashwin Ashok); part supervision, A.A. (AbdulHaseeb Ahmed) and M.R.R.; investigation and original draft preparation, A.A. (AbdulHaseeb Ahmed); software, experiment and writing assistance, S.T.V. and M.R.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Science Foundation (NSF), grant numbers 1901133 and 2000475.

Data Availability Statement: We have made the raw data used in this paper available here: https://realityawarenetworks.com (accessed on 19 May 2021).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

VLC	Visible Light Communication
DL	Deep Learning
DNN	Deep Neural Network
OCC	Optical Camera Communication
LED	Light Emitting Diode
ML	Machine Learning
ANN	Artificial Neural Network
BER	Bit Error Rate
MIMO	Multi-Input Multi-Output
FPS	Frames per Second
Hz	Hertz
YOLO	You only look once

References

1. pureLiFi: The LiFi Technology. Available online: http://purelifi.com/ (accessed on 10 September 2021).
2. IEEE 802.15.7r1 Status, 2015. Available online: http://vlca.net/site/wp/wp-content/uploads/2016/01/2015_10_26_YeongMinJangICEVLC2015CC.pdf (accessed on 10 September 2021).
3. IEEE P802.11-Light Communication (LC) Task Group (TG). Available online: https://www.ieee802.org/11/Reports/tgbb_update.htm (accessed on 10 September 2021).
4. Azevedo, I.L.; Morgan, M.G.; Morgan, F. The Transition to Solid-State Lighting. *Proc. IEEE* **2009**, *97*, 481–510. [CrossRef]
5. Pathak, P.H.; Feng, X.; Hu, P.; Mohapatra, P. Visible Light Communication, Networking, and Sensing: A Survey, Potential and Challenges. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2047–2077. [CrossRef]
6. What is a Photodiode? Working, Characteristics, Applications. 2018. Available online: https://bit.ly/2QF3Oer (accessed on 10 September 2021).
7. Nguyen, T. What is Optical Camera Communication? 2018. Available online: https://opticalpress.com/2018/08/08/what-is-optical-camera-communication/ (accessed on 10 September 2021).
8. VLC-Centric Indoor Localization. Available online: https://www.ram-lab.com/visual-light-communication/ (accessed on 10 September 2021).
9. Ashraf, K.; Varadarajan, V.; Rahman, M.R.; Walden, R.; Ashok, A. See-Through a Vehicle: Augmenting Road Safety Information Using Visual Perception and Camera Communication in Vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 3071–3086. [CrossRef]
10. Thunberg, J.; Lyamin, N.; Sjöberg, K.; Vinel, A. Vehicle-to-Vehicle Communications for Platooning: Safety Analysis. *IEEE Netw. Lett.* **2019**, *1*, 168–172. [CrossRef]
11. Choi, D.N.; Jin, S.Y.; Lee, J.; Kim, B.W. Deep Learning Technique for Improving Data Reception in Optical Camera Communication-Based V2I. In Proceedings of the 28th International Conference on Computer Communication and Networks (ICCCN), Valencia, Spain, 29 July–1 August 2019; pp. 1–2. [CrossRef]
12. Langer, K.; Grubor, J. Recent Developments in Optical Wireless Communications using Infrared and Visible Light. In Proceedings of the 2007 9th International Conference on Transparent Optical Networks, Rome, Italy, 1–5 July 2007; Volume 3, pp. 146–151.
13. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016. Available online: http://www.deeplearningbook.org (accessed on 10 September 2021).
14. Pachpande, P.G.; Khadr, M.H.; Hussein, A.F.; Elgala, H. Visible Light Communication Using Deep Learning Techniques. In Proceedings of the 2018 IEEE 39th Sarnoff Symposium, Newark, NJ, USA, 24–25 September 2018; pp. 1–6.
15. Lee, H.; Lee, I.; Quek, T.; Lee, S.H. Binary signaling design for visible light communication: A deep learning framework. *Opt. Express* **2018**, *26*,18131–18142. [CrossRef] [PubMed]
16. Wang, C.; Wu, G.; Du, Z.; Jiang, B. Reinforcement Learning Based Network Selection for Hybrid VLC and RF Systems. *MATEC Web Conf.* **2018**, *173*, 03014. [CrossRef]
17. Wang, X.; Shen, J. Machine Learning and its Applications in Visible Light Communication Based Indoor Positioning. In Proceedings of the 2019 International Conference on High Performance Big Data and Intelligent Systems (HPBD IS), Shenzhen, China, 9–11 May 2019; pp. 274–277.

18. He, J.; Hsu, C.; Zhou, Q.; Tang, M.; Fu, S.; Liu, D.; Deng, L.; Chang, G. Demonstration of High Precision 3D Indoor Positioning System Based on Two-Layer Ann Machine Learning Technique. In Proceedings of the 2019 Optical Fiber Communications Conference and Exhibition (OFC), San Diego, CA, USA, 3–7 March 2019; pp. 1–3.
19. Dang, S.; Ma, G.; Shihada, B.; Alouini, M.S. Enabling Smart Buildings by Indoor Visible Light Communications and Machine Learning. *arXiv* **2019**, arXiv:cs.NI/1904.07959.
20. He, W.; Zhang, M.; Wang, X.; Zhou, H.; Ren, X. Design and Implementation of Adaptive Filtering Algorithm for VLC Based on Convolutional Neural Network. In Proceedings of the 2019 IEEE 5th International Conference on Computer and Communications (ICCC), Chengdu, China, 6–9 December 2019; pp. 317–321.
21. Chi, N.; Zhou, Y.; Zhao, Y.; Li, G.; Hu, F. High-speed Visible Light Communication Based on Machine Learning. In Proceedings of the 2019 IEEE Photonics Society Summer Topical Meeting Series (SUM), Ft. Lauderdale, FL, USA, 8–10 July 2019; pp. 1–2.
22. Ma, S.; Dai, J.; Lu, S.; Li, H.; Zhang, H.; Du, C.; Li, S. Signal Demodulation With Machine Learning Methods for Physical Layer Visible Light Communications: Prototype Platform, Open Dataset, and Algorithms. *IEEE Access* **2019**, *7*, 30588–30598. [CrossRef]
23. Miao, P.; Zhu, B.; Qi, C.; Jin, Y.; Lin, C. A Model-Driven Deep Learning Method for LED Nonlinearity Mitigation in OFDM-Based Optical Communications. *IEEE Access* **2019**, *7*, 71436–71446. [CrossRef]
24. Turan, B.; Coleri, S. Machine Learning Based Channel Modeling for Vehicular Visible Light Communication. *arXiv* **2020**, arXiv:eess.SP/2002.03774.
25. Wu, X.; Huang, Z.; Ji, Y. Deep neural network method for channel estimation in visible light communication. *Opt. Commun.* **2020** *462*, 125272. [CrossRef]
26. Pham, T.L.; Shahjalal, M.; Bui, V.; Jang, Y.M. Deep Learning for Optical Vehicular Communication. *IEEE Access* **2020**, *8*, 102691–102708. [CrossRef]
27. Chuang, Y.C.; Chow, C.W.; Liu, Y.; Yeh, C.H.; Liao, X.L.; Lin, K.H.; Chen, Y.Y. Using logistic regression classification for mitigating high noise-ratio advisement light-panel in rolling-shutter based visible light communications. *Opt. Express* **2019**, *27*, 29924–29929. [CrossRef] [PubMed]
28. Hsu, K.L.; Chow, C.W.; Liu, Y.; Wu, Y.C.; Hong, C.Y.; Liao, X.L.; Lin, K.H.; Chen, Y.Y. Rolling-shutter-effect camera-based visible light communication using RGB channel separation and an artificial neural network. *Opt. Express* **2020**, *28*, 39956–39962. [CrossRef]
29. Chow, C.W.; Liu, Y.; Yeh, C.H.; Chang, Y.H.; Lin, Y.S.; Hsu, K.L.; Liao, X.L.; Lin, K.H. Display Light Panel and Rolling Shutter Image Sensor Based Optical Camera Communication (OCC) Using Frame-Averaging Background Removal and Neural Network. *J. Light. Technol.* **2021**, *39*, 4360–4366. [CrossRef]
30. Liu, L.; Deng, R.; Chen, L.K. 47-kbit/s RGB-LED-based optical camera communication based on 2D-CNN and XOR-based data loss compensation. *Opt. Express* **2019**, *27*, 33840–33846. [CrossRef]
31. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
32. Keras Applications. Available online: https://keras.io/api/applications/ (accessed on 10 September 2021).
33. Rosebrock, A. Generating ArUco Markers with OpenCV and Python. 2020. Available online: https://bit.ly/3nyQTH8 (accessed on 10 September 2021).