ECE-GY 6143: Introduction to Machine Learning

Fall 2020, Sundeep Rangan

Project Report

Tommy Azzino ta1731

Kubilay Ülger ou2007

Özlem Yıldız zy2043

December 14, 2020

Contents

1 Introduction

Wireless communication systems are including Millimeter-Wave frequencies (at 30 - 300 GHz) in the next generation systems due to a shortage in the conventional cellular bands [1]. However, transitioning to Millimeter-Wave frequencies has its own difficulties. One of them is that they suffer from path-loss, blockage, and shadowing more than the conventional ones, so obtaining the optimum beam direction in the beamforming process became an important topic [2]. The large number of antennas required to perform robust beamforming translates into overhead required for beam training [2]. Therefore, beam-tracking is one of the most important challenges in Millimeter-Wave systems. In this regard, the idea of exploiting machine learning techniques to proactively predict the best beam for a certain user exploiting a sequence of observed beams (and visual data) became an interesting research topic.

The motivation to work in vision aided wireless communications arises from the increasing availability of data collected from RGB/depth cameras or Lidar sensor and the great achievements obtained in computer vision through deep learning [3]. Also, note that beam prediction done by exploiting previous beams would struggle in real life due to the environment, blockages, and multiple users [4]. This is why the visual data is suggested as a supplementary source to enhance the prediction process.

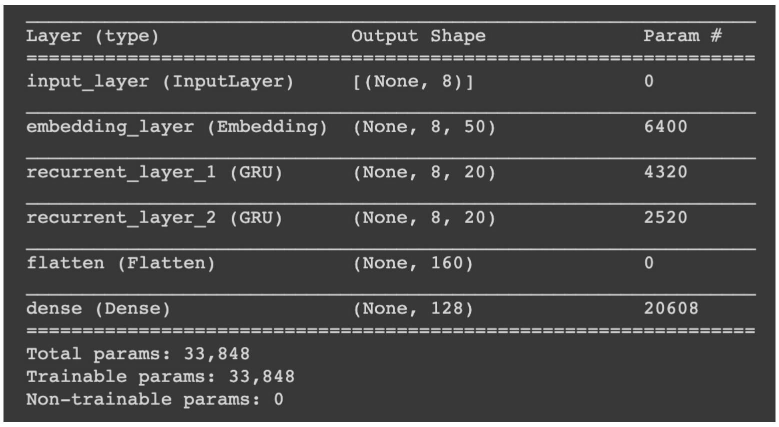
This problem is addressed in the Vision-Aided Millimeter-Wave Beam Tracking (ViWi-BT) competition at ICC 2020 [4]. Datasets for the project are also taken from the competition and they include image-beam sequences. The setting in here consists of two base stations with a mmWave uniform linear array antenna and three RGB cameras each [4]. The base station has a fixed beam steering codebook and beam-forming vectors are described by their indices of the beam. Previous 8 beams and images of the observed beams of the user form the input to the prediction algorithm. The prediction task included 3 levels in the competition: predicting one, three, or five successive beams. Thus, 13 pairs of beam and images are given in the dataset.

In this project, we investigate the prediction of the next beam in the sequence from the previously observed beams and visual data. We, therefore, neglect the prediction of the successive 3 or 5 beams in the sequence.

2 Methodology

In this project, we focused on predicting the 9th beam by exploiting a history of the previous 8 beams. Since these 9 beams occur sequentially in time, we have chosen to use Recurrent Neural Networks (RNNs) to train our model. In RNNs, the presence of an internal memory state makes them suitable for time series data analysis. We used 281100 training data samples and 120468 validation samples. Our target has 128 different classes that we want to predict (i.e. the codebook size is 128). We used 100 epochs and a batch size of 1000.

Initially, we trained our model using RNNs without optimizing the main hyper-parameters involved. However, we have seen that we could train the model in a reasonable time and decided to optimize the parameters. We have determined 6 important parameters: learning rate, number of recurrent layers, embedding size, hidden size of the recurrent layer, dropout value, and the *Return Sequence* parameter of a recurrent layer. The embedding size represents the output size of the first layer of the network. This layer is an Embedding layer that receives as input the history of beams. The hidden size parameter refers to the output dimensionality associated with a recurrent layer. The *Return Sequence* parameter (related to the last layer of the recurrent block) enables the output of the entire time sequence from the final recurrent layer, otherwise only the last output will be returned. Ultimately, the dropout parameter regulates the dropout rate for the units in the network.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_layer (InputLayer)     [(None, 8)]               0
_________________________________________________________________
embedding_layer (Embedding)  (None, 8, 50)             6400
_________________________________________________________________
recurrent_layer_1 (GRU)      (None, 8, 20)             4320
_________________________________________________________________
recurrent_layer_2 (GRU)      (None, 8, 20)             2520
_________________________________________________________________
flatten (Flatten)            (None, 160)               0
_________________________________________________________________
dense (Dense)                (None, 128)               20608
=================================================================
Total params: 33,848
Trainable params: 33,848
Non-trainable params: 0
```

Figure 1: Summary of the RNN (The architecture might be different depending on the values of the selected parameters)

Since we potentially have many parameters to test but limited resources, we have

Parameters	Value 1	Value 2	Value 3
Learning Rate	2.10^{-4}	10^{-3}	5.10^{-3}
Number of Layers	2	3	4
Return Sequence	True	False	
Embedding Size	50	100	
Hidden Size	20	40	
Dropout	0.2	0.5	

Table 1: Pool of selected parameters for the grid search

Parameters	Best Value
Learning Rate	10^{-3}
Number of Layers	3
Return Sequence	True
Embedding Size	100
Hidden Size	40
Dropout	0.2

Table 2: Best parameters through grid search

chosen to adopt 2 or 3 values for each hyper-parameter and to perform a grid search to obtain the best values. Table 1 shows the parameters we chose to do grid search on. We trained our model a total of 144 times (i.e. the total number of combinations of each parameter) to pick up the best values.

An example of the RNN architecture adopted in this problem is depicted in Figure 1.

3 Results

In this section, we will present the results we have obtained using the parameters mentioned in the previous section. The best validation accuracy, we achieved, is 0.8510 using the parameters shown in Table 2. This result is very close to the one obtained by the author of [4] (the authors of the competition), where they were able to get 0.86 accuracy.

Figure 2 shows the error rate vs epoch number plots for different learning rates. We have decided to use the error rate as a parameter instead of accuracy and plot it in a semi-log graph to clearly show the difference. As expected, if we use a low learning rate the model's convergence is slower and with a high learning rate, we experienced instability during the learning process. By using a learning rate of 10^{-3} (default value for the Adam optimizer), our model was able to converge quickly and smoothly.

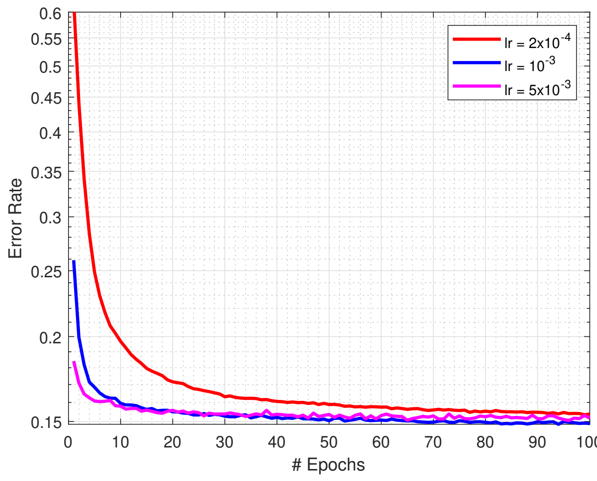


Figure 2: Error Rate wrt Different Learning Rates

Our second parameter is the number of recurrent layers. Figure 3 shows the validation accuracy vs epoch number for 3 different number of layers. As can be seen, the performance does not change that much but, at the end of 100 epochs, we have seen that using 3 layers gives the best result by a small margin.

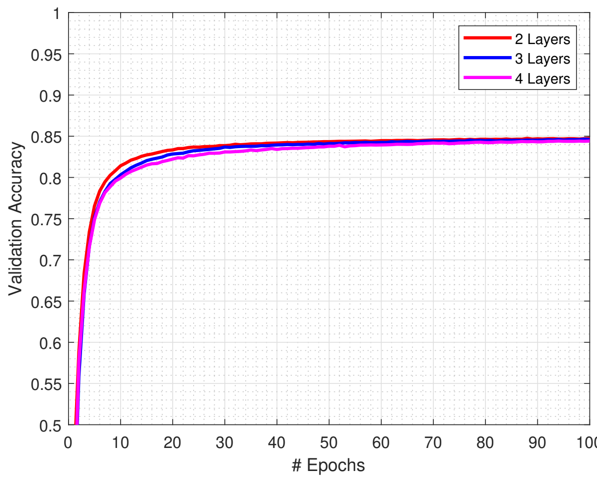
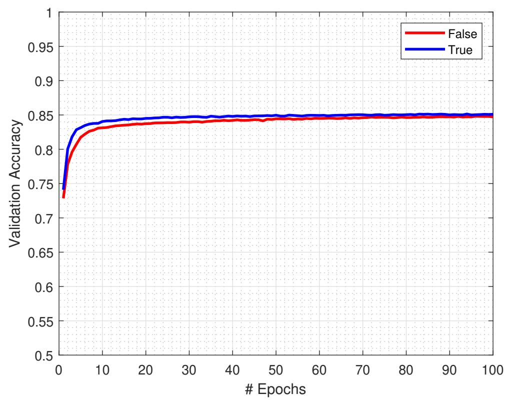


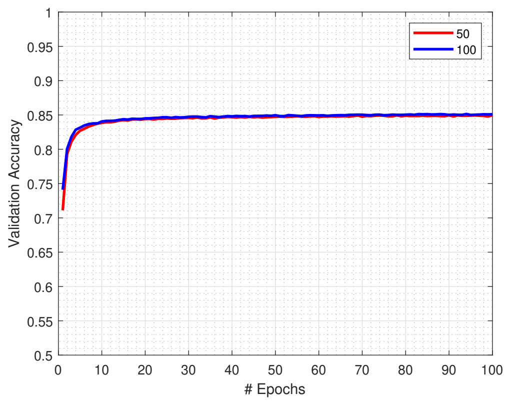
Figure 3: Validation Accuracy wrt Number of Layers

One of the parameters of our RNN model is the value *Return Sequence*: True or False. Figure 4a shows the difference. Clearly, returning the entire output of the time sequence from the last recurrent layer seems a better option.

On the other hand, Figure 4b shows that an increasing of the embedding size value from 50 to 100 guarantees a small performance improvement.



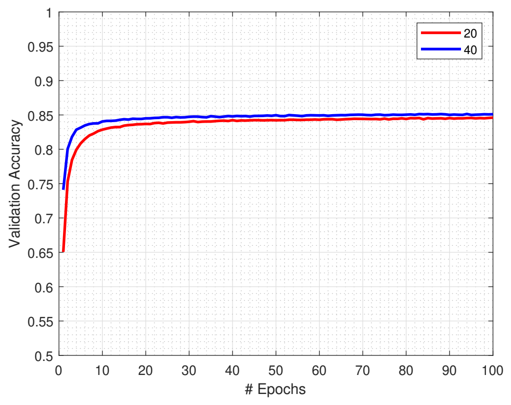
(a) Retention Sequence



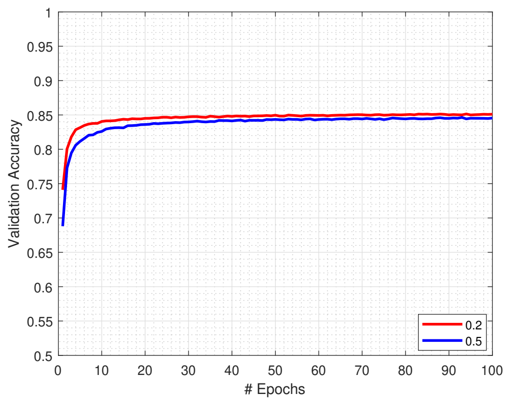
(b) Embedding Size

Figure 4: Validation Accuracy wrt Retention Sequence and Embedding Size

The last 2 parameters we tested, the hidden size and dropout also have minimal effect on the validation accuracy. The plots for these parameters can be seen in Figure 5. From these results we decided to go with a hidden size of 40 and a dropout of 0.2.



(a) Hidden Size



(b) Dropout

Figure 5: Validation Accuracy wrt Hidden Size and Hidden Size

After choosing the best parameters in our grid search, we decided to look at the validation and train accuracy we obtained. Figure 6 shows train and validation accuracy vs epoch number when we use the best parameters. It can be seen that both train and validation accuracy lie just above 0.85. There is not much difference between validation accuracy and train accuracy. As expected, the final training accuracy is higher than the corresponding validation accuracy.

Ultimately, we re-trained the model using the optimal parameters of Table 2, 200 epochs, and a decaying learning rate to obtain the final validation accuracy value of 0.85257.

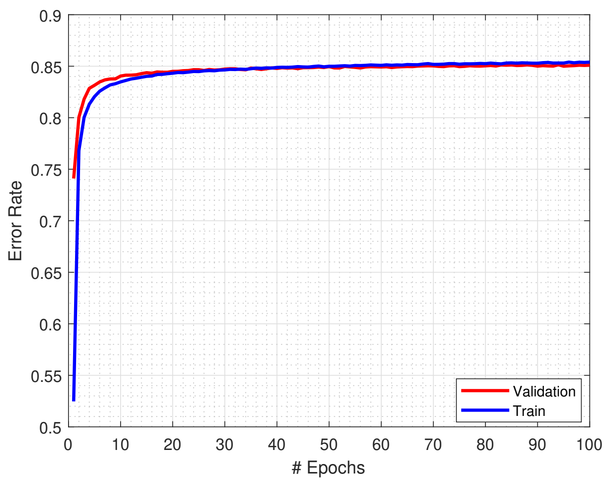


Figure 6: Validation and Training Accuracy with best Parameters

4 Future Work

As a final consideration, in this project, we also implemented models that combined the beam and the visual data to predict the next beam for the target user. These models are based on Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. However, we found a bottleneck during the training process related to the loading of images during each batch. The training time was extremely slow. We were able to run these models for a few epochs but we did not find any improvement with the addition of visual data. The code related to these implementations is contained in the GitHub repository of the project. Therefore, this part of the project is left as future work.

5 Conclusion

In conclusion, the main goal of the project was to investigate Machine Learning techniques in order to overcome the challenges of beam-tracking in the next-generation wireless systems at Millimeter-Wave frequencies. In such systems, predicting the best beam with an optimum method is essential due to the overhead, blockage, path loss, etc. The project was based on the competition at ICC 2020 [4]. In our project, we predicted the 9th beam according to the given first 8 beams. In this prediction, we have used RNNs. Moreover, we selected the best parameters for the model using grid search. The best validation accuracy we have obtained is 0.85257. Finally, we left the second part of the project as future work due to some technical difficulties we have encountered.

References

[1] M. R. Akdeniz, Y. Liu, M. K. Samimi, S. Sun, S. Rangan, T. S. Rapaport, and E. Erkip, "Millimeter wave channel modeling and cellular capacity evaluation," 2014.

[2] M. Alrabeiah, J. Booth, A. Hredzak, and A. Alkhateeb, "Viwi vision-aided mmwave beam tracking: Dataset, task, and baseline solutions," 2020.

[3] M. Alrabeiah, A. Hredzak, and A. Alkhateeb, ""millimeter wave base stations with cameras: Vision aided beam and blockage prediction," 2019.

[4] "The viwi-bt competition." https://www.viwi-dataset.net/challenge$_test_set.html.Accessedon2020-10-15$.