COMP34212 Cognitive Robotics

Chittesh Kumar Singore

April 28, 2024

1 Introduction

The report discusses the role of deep learning within the context of state-of-the-art robotics and showcases the skill to develop and evaluate a deep learning network (CNN) to classify images from **iCub World dataset**. Firstly, we start with learning about the applications of DNN to different domains, and then we move on to evaluating a CNN model over images from the dataset on different experiments.

2 Literature Review

Object recognition is an essential capability for operating in unstructured situations. Deep learning especially Convolutional Neural Networks (CNNs) transformed different domains of robotics, excelling in extracting features and handling high-dimensional data [8].

Applications for robotics span from industrial bin-picking[7] to service robotics handling everyday domestic things (for example; Samsung Ballie Robot[1]). Deep learning has made a significant impact across several core domains within robotics:

- CNNs have shown exceptional performance in visual tasks such as objection and scene segmentation. The SLAM technique[6] enables robots to function in congested situations by comprehending a wide variety of objects to create complex structured maps of the environment for navigation.
- The neural networks can also be used for predicting safe paths and avoiding obstacles from the visual inputs in real-time [2]. Deep Learning reinforcement techniques such as Deep Q-Networks (DQNs)[5], enable robots to learn complicated behaviors based on a reward system to be able to adjust their navigation strategies
- DNNs used for Human-Robot Collaboration[4] to analyze human motion, gestures, and even physiological data (eye-tracking), enabling robots to seamlessly adapt their actions, resulting in more efficient and intuitive collaboration in shared workspace.

- DNNs are finding novel applications in the field of swarm robotics. By researching fish behavior, researchers created a DNN-based model for robot interaction[11]. This model uses a novel method called LVPS to manage complicated situations. Simulation demonstrates that it accurately replicates fish behavior and effectively manages large-scale swarms. This study highlights how DNNs can transform how we design swarming behaviors.

AlexNet[10](2012) and ResNet[3](2015) highlight the advancements in object recognition in cognitive robotics. AlexNet, with significant performance gains in large-scale image recognition tasks, introduced techniques like dropout regularization and data augmentation. ResNet elevated the CNN design by enabling deeper networks with its residual blocks, overcoming the vanishing gradient problem.

Deep learning models are trained on huge, diverse datasets and demonstrate exceptional generalization to new objects and contexts. Deep learning has transformed different domains of object recognition in robotics, allowing robots to navigate complicated environments, avoid obstacles, and work efficiently with humans. It has the potential to generate significant future breakthroughs.

3 Methodology

Before performing the experiments, two different models were set up: a simple CNN model and a complex CNN model from the lab exercises. Three experiments were performed by modifying hyperparameters such as the number of epochs, different optimizers, and dropout values to examine the performance of the dataset using both models. Following is the network description of each model:

1. Simple CNN Model Architecture:

   Input + Conv1 + ReLU + MaxPooling1 + Dropout

   Flatten + Dense1 + ReLU + Dropout

   Dense2 + Softmax + Output

2. Complex CNN Model Architecture:

   Input + Conv1 + ReLU + Conv2 + ReLU + MaxPooling1 + Dropout

   Conv3 + ReLU + Conv4 + ReLU + MaxPooling2 + Dropout

   Flatten + Dense1 + ReLU + Dropout

   Dense2 + Softmax + Output

4 Experiments

The iCub World 1.0 dataset[9] and CNN models from the lab exercises have been used for the experiments.

4.1 Experiment-1: Number of Epochs

The experiment investigates how epoch numbers influence two CNN models with varied complexity. Figure 1 illustrates the simple CNN model's training accuracy approaching 70%, whereas test accuracy peaks at 65%. This performance disparity shows that the training data has been overfitted. In contrast, Figure 2 shows that a complex model makes rapid gains in both training and test accuracy (over 50% and 45% respectively). The smaller gap between these accuracies suggests better performance of the complex model compared to the simple model.

Overall, while the simple CNN trains to higher accuracy faster, the complex model's more modest performance might suggest better generalizability potential, particularly when trained for more epochs under the safeguard of EarlyStopping mechanisms.

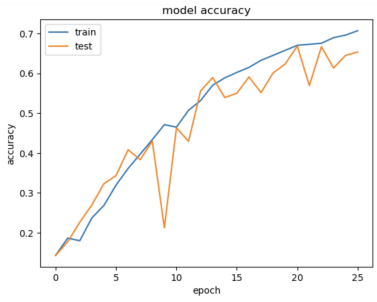

Figure 1: Simple CNN Model

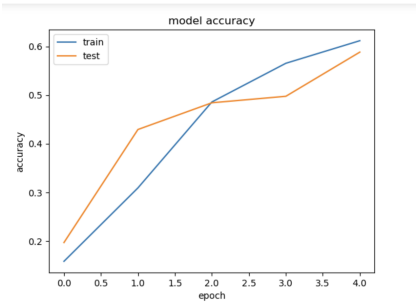

Figure 2: Complex CNN Model

4.2 Experiment-2: Different Optimizers

This experiment explores the performance of a complex CNN model with different choices of optimizer. Three different optimizers were chosen to test: SGD, Adam, and RMSProp. The epoch value was kept constant at 5 along with another parameter. To prevent overfitting, EarlyStopping was used to terminate training when validation accuracy stopped improving.

From Figure 3, the validation accuracy improved significantly only with RMSprop, indicating better learning. In contrast, SGD and Adam improved a little. Validation loss fell remarkably with RMSprop (figure 4), although it remained high and constant for the others. This indicates that RMSprop effectively optimizes the complex CNN model, while SGD and Adam are less effective within same epoch range. RMSprop's adaptability to the model's architecture may explain its better performance in preventing overfitting and improving accuracy.

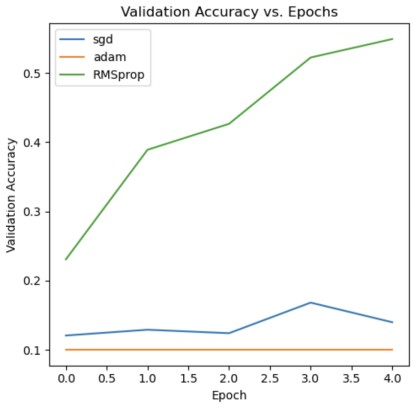

Figure 3: Validation Accuracy by Each Optimizer

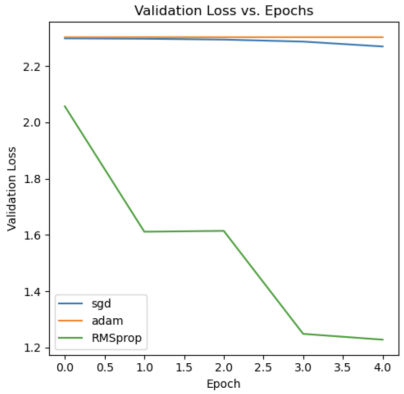

Figure 4: Validation Loss by Each Optimizer

4.3 Experiment-3: Different Dropouts Values

The experiment examines how the performance of a complex CNN model is impacted by the use of different dropout rates, a regularization technique used during training to prevent overfitting. Table-1 illustrate the findings as follows: A dropout rate of 0.1 leads to higher accuracy but with more loss i.e. the model might tend to overfit. At 0.25 dropout, the balance between accuracy and moderate loss is maintained. Dropout value of 0.4 shows underfitting due to low accuracy and loss. A high dropout rate of 0.7 significantly reduces accuracy and doesn't minimize loss effectively. Overall, the dropout rate of 0.25 is ideal in the above case.

Dropout Rate	Validation Accuracy	Validation Loss
0.1	0.6267	0.8792
0.25	0.6708	0.8838
0.4	0.5958	1.1098
0.7	0.4575	1.4478

Table 1: Validation Accuracy and Loss at Different Dropout Rates

5 Conclusion

In conclusion, the outcomes of the experiments with two CNN models (simple and complex) on the iCub World 1.0 dataset revealed that hyperparameters significantly impact model accuracy. The number of epochs, use of different optimizers, and distinct dropout rates influenced the accuracy of both CNN models. The simple CNN quickly reached high accuracy, while the complex

CNN showed greater generalizability with EarlyStopping. The optimal dropout rate was found to be 0.25, illustrating the importance of hyperparameter tuning and data handling in cognitive robotics.

iCub World 1.0 dataset from downloaded from its website[9]. And organized in the same directory as the Jupyter Notebook file. Link to the experiment code: `https://github.com/ChitteshKumar/Cognitive_Robotics`

References

[1] A. Brush, J. Albrecht, and R. Miller. Smart homes. *IEEE Pervasive Computing*, 19(2):69–73, 2020.

[2] J. O. Gaya, L. T. Gonçalves, A. C. Duarte, B. Zanchetta, P. Drews, and S. S. Botelho. Vision-based obstacle avoidance using deep learning. In *2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR)*, pages 7–12. IEEE, 2016.

[3] B. Koonce and B. Koonce. Resnet 50. *Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization*, pages 63–72, 2021.

[4] J. Mainprice and D. Berenson. Human-robot collaborative manipulation planning using early prediction of human motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 299–306. IEEE, 2013.

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[6] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.

[7] M. Ojer, X. Lin, A. Tammaro, and J. R. Sanchez. Pickingdk: A framework for industrial bin-picking applications. *Applied Sciences*, 12(18):9200, 2022.

[8] A. R. Pathak, M. Pandey, and S. Rautaray. Application of deep learning for object detection. *Procedia computer science*, 132:1706–1717, 2018.

[9] Robotology. icubworld 1.0. `https://robotology.github.io/iCubWorld/#icubworld-1-modal`. Accessed: April 20, 2024.

[10] Z.-W. Yuan and J. Zhang. Feature extraction and image retrieval based on alexnet. In *Eighth International Conference on Digital Image Processing (ICDIP 2016)*, volume 10033, pages 65–69. SPIE, 2016.

[11] H. Zhang and L. Liu. Intelligent control of swarm robotics employing biomimetic deep learning. *Machines*, 9(10):236, 2021.