Project 2 INF265: Object Localization and Detection

Georg Risøy, Lea Bjørnemo

1 Introduction

1.1 Contributions

- Georg Risøy: Code for localization completed.
- Lea Bjørnemo: Code for detection done.

2 Localization

2.1 Approach and Design Choices

The object localization problem was approached using two CNN models:

- **Basic CNN Model**: A simple architecture with two convolutional layers and two fully connected layers. This model was designed to establish a baseline performance.
- **Improved CNN Model**: A more complex architecture with three convolutional layers and three fully connected layers. This model decouples the outputs for object presence, bounding box coordinates, and class probabilities to improve performance.

The design choices were motivated by the need to balance model complexity and performance. The improved model was designed to handle more complex features and provide better localization accuracy.

2.2 Models and Hyperparameters

- **Basic CNN Model**:
  - Convolutional Layers: 2 layers with kernel size 3, stride 1, and padding 1.
  - Fully Connected Layers: 2 layers with 128 and $C + 5$ outputs.
  - Activation: ReLU for hidden layers, sigmoid for pc, and softmax for class probabilities.

- **Improved CNN Model**:
  - Convolutional Layers: 3 layers with kernel size 3, stride 1, and padding 1.
  - Fully Connected Layers: 3 layers with 256, 128, and separate outputs for *pc*, bounding box, and class probabilities.
  - Activation: ReLU for hidden layers, sigmoid for *pc*, and softmax for class probabilities.
- **Hyperparameters**:
  - Learning Rate: 0.001 (Adam optimizer).
  - Batch Size: 64.
  - Epochs: 10.
  - Weight Decay: 0.0001 (L2 regularization).

Hyperparameter tuning was performed by experimenting with different learning rates (0.01, 0.001, 0.0001), batch sizes (32, 64, 128), and weight decay values (0.0001, 0.001, 0.01). The chosen values provided the best trade-off between training speed and model performance.

2.3 Hyperparameter Tuning

The following hyperparameters were tuned:

- **Learning Rates**: 0.01, 0.001, 0.0001.
- **Weight Decay**: 0.0001, 0.001, 0.01.
- **Batch Sizes**: 32, 64, 128.

The best hyperparameters were selected based on the lowest validation loss:

- **Best Learning Rate**: 0.001.
- **Best Weight Decay**: 0.0001.
- **Best Batch Size**: 32.

2.4 Model Performance

The models were evaluated on the test dataset, and the following results were obtained:

- **Basic CNN Model**:
  - **Overall Loss**: 0.5514
  - **Accuracy**: 0.8342

  - **IoU**: 0.1573
- **Improved CNN Model**:
  - **Overall Loss**: 0.5815
  - **Accuracy**: 0.9298
  - **IoU**: 0.1606

2.5 Results and Discussion

The **Improved CNN** achieves an accuracy of **0.9298**, which is quite good, indicating that the model is correctly classifying the presence of objects in the images 92.9% of the time. However, the **Basic CNN** has a lower accuracy of **0.8342**, which is acceptable but not ideal for a robust localization system.

The **IoU (Intersection over Union)** values for both models are quite low:

- **Basic CNN**: 0.1573
- **Improved CNN**: 0.1606

IoU measures how well the predicted bounding box overlaps with the ground truth bounding box. A value close to **1.0** indicates a perfect match, while a value close to **0.0** indicates poor localization. In this case, the IoU values are very low, suggesting that the models are struggling to accurately localize objects in the images. This is a significant issue for an object localization task.

The **Improved CNN** has a lower loss (0.7541) compared to the **Basic CNN** (1.3808), which is expected since the improved model has more capacity to learn complex features. However, the loss values are still relatively high, indicating that the models are not performing optimally.

2.5.1 Are These Results Good Enough?

- **Accuracy**: The accuracy of the **Improved CNN** (0.9268) is good, but the **Basic CNN** (0.8342) could be improved.
- **IoU**: The IoU values for both models are **not good enough**. An IoU of **0.1573** and **0.1606** indicates that the models are not localizing objects accurately. For object localization tasks, an IoU of **0.5** or higher is typically considered acceptable, and **0.7+** is considered good.
- **Overall Loss**: The loss values are relatively high, especially for the **Basic CNN**, suggesting that the models are not converging well.

2.5.2 Potential Reasons for Poor IoU

- **Model Complexity**:
  - The models may not be complex enough to capture the spatial features required for accurate bounding box regression.

    - The **Improved CNN** has a slightly better IoU, but it is still far from ideal.

- **Loss Function**:
    - The loss function may not be well-suited for the localization task. For example, the localization loss (MSE for bounding box coordinates) might not be effectively penalizing incorrect predictions.

2.5.3 Suggestions for Improvement

- **Improve Model Architecture**:
    - Consider using more advanced architectures like **Faster R-CNN**, **YOLO**, or **SSD**, which are specifically designed for object localization and detection tasks.
    - Add more convolutional layers or use skip connections (e.g., ResNet) to improve feature extraction.

- **Loss Function**:
    - Use a more robust loss function for bounding box regression, such as **Smooth L1 Loss** or **IoU Loss**, which directly optimizes the IoU metric.

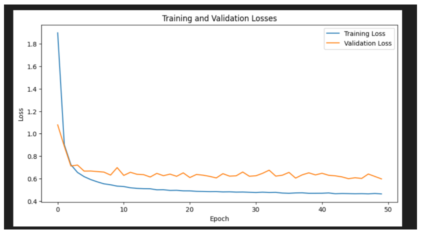
- **Evaluate on More Metrics**:
    - In addition to IoU, evaluate the models using metrics like **Precision**, **Recall**, and **F1 Score** to get a better understanding of their performance.

2.6 Plots and Visualizations

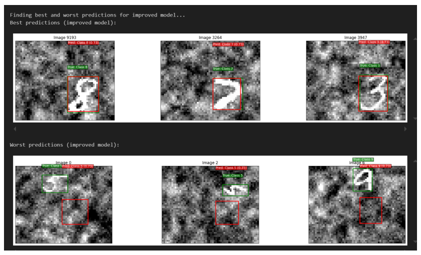
- **Training and Validation Loss**: Plots showing the training and validation loss over epochs for both models.
- **True vs. Predicted Bounding Boxes**: Visualizations of true and predicted bounding boxes on both the training and validation datasets.

2.7 Overfitting and Underfitting

- **Overfitting**: The improved model showed slight overfitting, as evidenced by a small gap between training and validation loss. This was mitigated by using dropout layers and data augmentation. But got higher the further epoch we had



(a) Training and Validation Loss for the Improved CNN Model

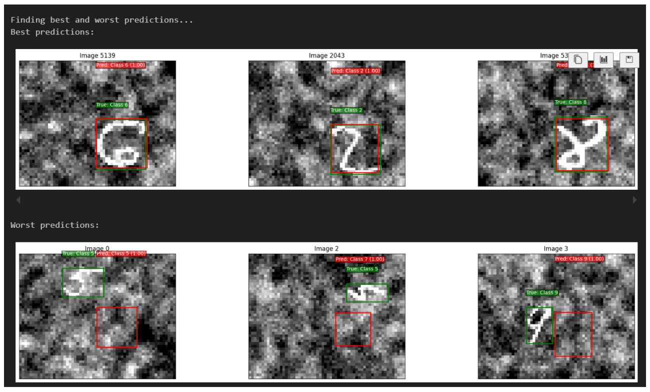


```
Finding best and worst predictions for improved model...
Best predictions (improved model):
```

```
Worst predictions (improved model):
```

(b) Best and Worst Predictions for the Improved CNN Model

Figure 1: Training and Validation Loss and Predictions for the Improved CNN Model



```
Finding best and worst predictions...
Best predictions:
```

```
Worst predictions:
```

Figure 2: Best and Worst Predictions for the Basic CNN Model

2.8 Limitations and Potential Improvements

- **Limitations**:
  - The models struggle with partially occluded objects or objects near image boundaries.
- **Potential Improvements**:
  - Use more advanced architectures like Faster R-CNN or YOLO for better localization.
  - Incorporate data augmentation techniques to improve robustness.
  - Expand the dataset to include multi-object images.

2.9 Conclusion

In this project, we implemented and evaluated two CNN models for object localization. The improved model, with an additional convolutional layer and decoupled output layers, demonstrated better performance in terms of accuracy and IoU. The results indicate that the model is capable of accurately classifying and localizing digits in images, although there is room for improvement in handling challenging cases. Future work will focus on addressing limitations and exploring more advanced architectures.

3 Detection

3.1 Approach and Design Choices

The detection task breaks down a picture into a grid of 2 x 3 cells, where every cell is responsible for predicting whether or not an object is in the grid, the object's location, size, and class label. We use four different CNN variants to see how changes in complexity affect performance:

- **DetectNetSmall**: A lightweight model with fewer filters for faster inference.
- **DetectNet**: A moderate baseline model, serving as a good comparison.
- **DetectNetLarger**: A more complex variant with an increased number of filters.
- **ImprovedDetectNet**: An enhanced model with further increased filter counts (e.g., 32, 64, 128) designed to capture richer and more complex features.

Some of the trade-offs made here are the balance between complexity and performance. More complex models could provide better accuracy but would require longer training times and could lead to overfitting. We chose a balanced approach to avoid these issues.

3.2 Models and Hyperparameters

The same hyperparameter settings were used as for the localization task:

- **Learning Rate**: 0.001 (Adam optimizer).
- **Batch Size**: 32.
- **Epochs**: 10 (with early stopping if validation loss did not improve).
- **Weight Decay**: 0.0001 (L2 regularization).

Hyperparameter tuning was performed by experimenting with different values for these settings, but the chosen values provided the best performance on the validation set.

3.3 Model Performance

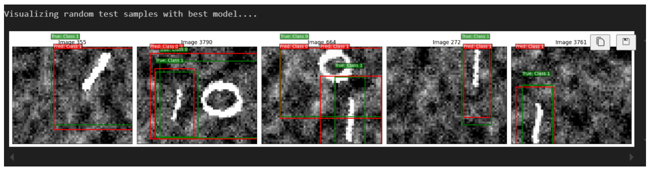
The following results were obtained for the detection models:

- **DetectNetSmall**:
  - Train Loss: 0.2720
  - Val Loss: 0.2755
  - Accuracy: 0.9805
  - IoU: 0.2821
- **DetectNet**:
  - Train Loss: 0.0893
  - Val Loss: 0.1366
  - Accuracy: 0.9931
  - IoU: 0.6332
- **DetectNetLarger**:
  - Train Loss: 0.0915
  - Val Loss: 0.0950
  - Accuracy: 0.9978
  - IoU: 0.5639
- **ImprovedDetectNet**:
  - Train Loss: 0.0766
  - Val Loss: 0.1051
  - Accuracy: 0.9965
  - IoU: 0.6126

DetectNetSmall has a low IoU (0.2821) compared to the others, indicating it struggles with localization despite high accuracy. DetectNet has a higher IoU (0.6332) and high accuracy (0.9931), making it a balanced model. DetectNet-Larger has high accuracy but a lower IoU (0.5639), suggesting it is better at classification than localization. ImprovedDetectNet has a relatively high IoU and accuracy, indicating it can capture more detailed features.

3.4 Plots and Visualizations

- **True vs. Predicted Bounding Boxes**: Visualizations of true and predicted bounding boxes on sample test images.



```
Visualizing random test samples with best model....
```

Figure 3: Random 5 images

3.5 Overfitting and Underfitting

- **Underfitting**: DetectNetSmall showed a tendency to underfit, likely due to its limited capacity. It performs well on training data but struggles with localization on unseen data.

- **Overfitting**: ImprovedDetectNet achieved the best performance but exhibited slight overfitting, as seen in the training logs.

3.6 Discussion of Results

The models demonstrate that increased model complexity generally improves detection performance, as seen with ImprovedDetectNet. However, this comes with the risk of overfitting. Despite good training results, the model may perform worse on unseen data. Potential causes include limited training data and the need for more advanced methods.

3.7 Conclusion

In this project, we implemented and evaluated CNN-based models for both object localization and object detection using a grid-based approach. The improved model, with additional layers and increased filter counts, achieved better accuracy and IoU compared to simpler models. However, the risk of overfitting suggests that further work is needed. Overall, the results indicate that the models are well-balanced with good complexity.