Optical Character Recognition of Tamil Language Using Deep Learning Techniques To Aid NLP Systems

Team Member(s):
22MIC0101 (Varshan Manish)

Report Submitted For The First Project Review Of:
Course Code: CSI4001
Course: Natural Language Processing And Computational Linguistics
Slot: C1 Slot

Professor: Dr. Sarmila K Banu

PROBLEM STATEMENT:

Despite the advancements in Optical Character Recognition (OCR) systems for Latin-based scripts, effective recognition of South Indian languages like Tamil remains a challenging task due to the script's complex structure, numerous characters, and high inter-class similarity. Existing OCR engines often underperform when handling handwritten Tamil characters, especially under conditions of noise, low resolution, and varying writing styles.

This project addresses the lack of robust OCR systems tailored for the Tamil language by developing a deep learning-based character recognition model. Using a Convolutional Neural Network (CNN) trained on the HP 2005 Isolated Handwritten Tamil Character Dataset, the aim is to classify 156 unique Tamil characters accurately. The solution seeks to improve regional language accessibility, enhance multilingual NLP pipelines, and contribute to the digitization of Tamil literary resources.

MODULES USED:

1. **PIL (Python Imaging Library):**
   Used for image loading, manipulation, inversion, and applying filters such as thickening to enhance character visibility.
2. **NumPy:**
   Used for array manipulations, pixel-wise operations, and image data conversion to numerical format suitable for model input.
3. **SciPy (ndimage module):**
   Used for image preprocessing tasks such as calculating the center of mass to center characters on a canvas.
4. **Torchvision (datasets, transforms):**
   Provides tools for image preprocessing and loading datasets efficiently, with transformations like resizing and normalization.
5. **PyTorch:**
   Core deep learning framework used for building, training, and evaluating the Convolutional Neural Network model.
6. **Pandas:**
   Used for organizing and analyzing tabular data such as training logs, accuracy reports, and class labels.
7. **Matplotlib:**
   Utilized for visualizing training progress, including plots of loss, accuracy, and confusion matrices.
8. **Torchvision (general):**
   Used for additional tools such as visualizing images and potentially loading pre-trained models.
9. **Torch.nn:**
   Provides neural network layers and building blocks like convolutional, linear, and batch normalization layers.
10. **Torch.nn.functional:**
    Used for applying functions like activation (ReLU), pooling, and loss calculations during the forward pass.
11. **Torch.optim:**
    Contains optimizers like Adam used for training the neural network by updating weights.

12. **Google Colab files:**
    Enables file uploads and downloads within the Google Colab environment, such as saving model checkpoints or uploading data.

METHODOLOGY ADAPTED:

The methodology adopted in this project follows a structured deep learning pipeline to recognize handwritten Tamil characters using a Convolutional Neural Network (CNN). The approach includes five main stages: data collection, preprocessing, model design, training, and analysis.

1. Data Collection

The dataset used is the IWFHR 2006 Online Tamil Handwritten Character Recognition Dataset. It includes samples of 156 isolated Tamil characters (vowels, consonants, and compound characters), written by multiple individuals.

2. Data Preprocessing

To ensure consistency and improve model performance, each image in the dataset undergoes a series of preprocessing steps:

- **Grayscale Conversion:** Images are converted to grayscale to focus on shape rather than color.
- **Inversion and Thickening:** The images are inverted to make the characters white on a black background, then slightly thickened using a max filter to highlight finer strokes.
- **Resizing and Centering:** Images are resized proportionally (maintaining aspect ratio) so that the largest side is 48 pixels. They are then centered on a 64x64 canvas using the center of mass to ensure uniform placement across samples.

This results in a clean, fixed-size input that is well-suited for CNN processing.

3. Model Architecture

The model is a deep convolutional neural network (CNN) designed to automatically extract spatial features from the input images and classify them into one of the 156 Tamil character classes. The architecture includes multiple convolutional layers followed by batch normalization and ReLU activation, interleaved with max pooling layers to progressively reduce spatial dimensions. These layers are followed by fully connected (dense) layers, ending with a final classification layer.

The model was designed to balance complexity and generalization, with careful tuning of the number of filters and neurons at each stage. Batch normalization after each layer improves stability during training.

4. Model Training and Evaluation

The model was trained using the cross-entropy loss function and optimized with the Adam optimizer. A learning rate scheduler was used to improve convergence. The dataset was divided into training, validation, and test sets to evaluate the model's generalization. Performance metrics such as overall accuracy, confusion matrix, and per-class accuracy were used to assess the results.

5. Observations and Analysis

The CNN model performed well, achieving high classification accuracy on the test set. It effectively learned the distinguishing features of various Tamil characters, even in cases where certain classes appeared visually similar. The preprocessing pipeline was particularly effective in enhancing image quality and improving classification outcomes.

Some misclassifications occurred for visually similar compound characters. Future improvements could include attention mechanisms or hybrid models incorporating recurrent layers for better context understanding.

DATASET DESCRIPTION:

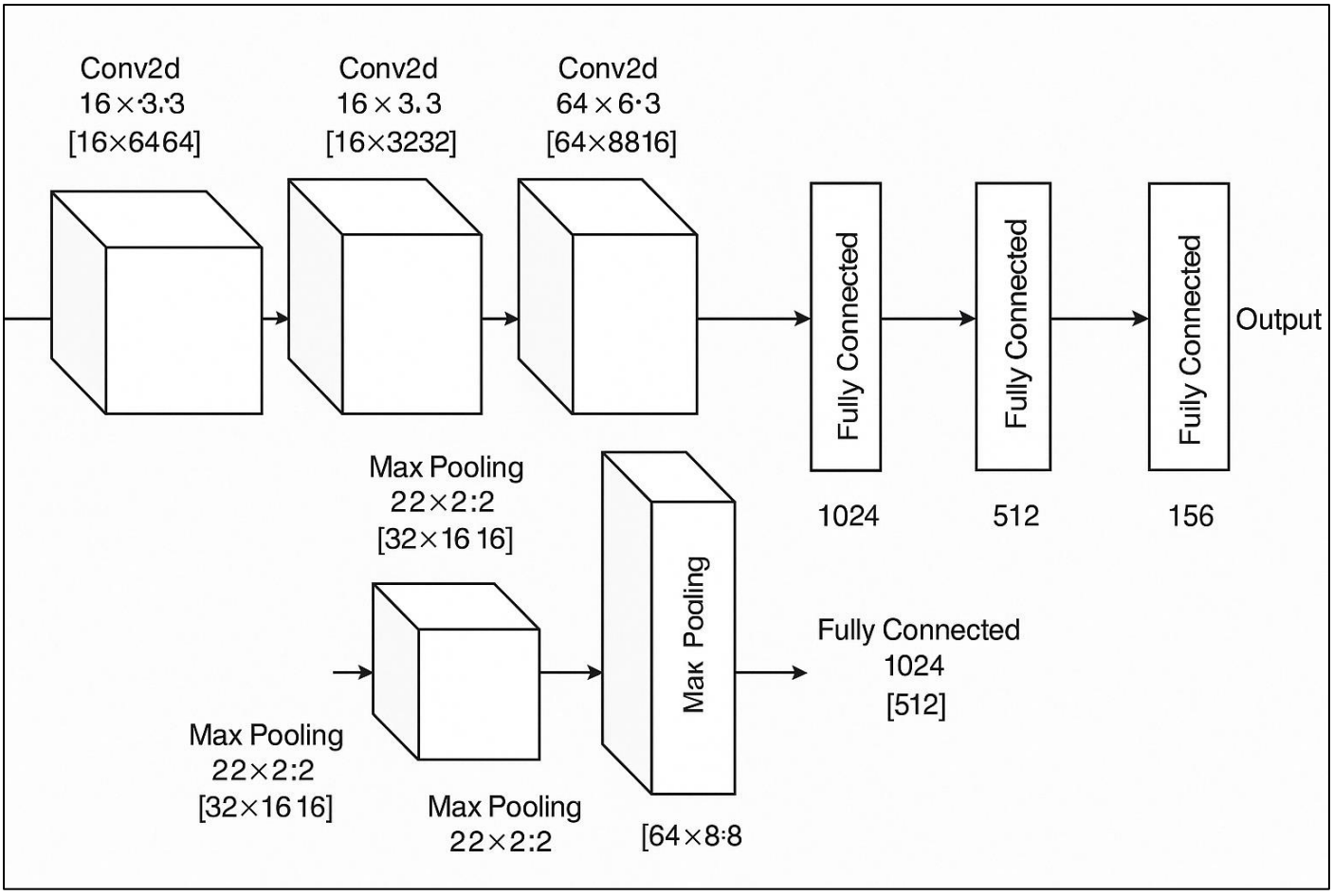
Presented at the 9th International Workshop on Frontiers in Handwriting Recognition (IWFHR), 2006. Created by HP Labs India in collaboration with academic institutions.

Content:

- Contains isolated handwritten Tamil characters written by multiple individuals.
- Comprises 156 character classes, including:
  - 12 vowels (உயிரெழுத்துகள்)
  - 18 consonants (மெய்யெழுத்துகள்)
  - Compound characters (உயிர்மெய்யெழுத்துகள்)
- Each character is scanned and digitized into grayscale images.

Dataset Link: https://shiftleft.com/mirrors/www.hpl.hp.com/india/research/penhw-interfaces-1linguistics.html

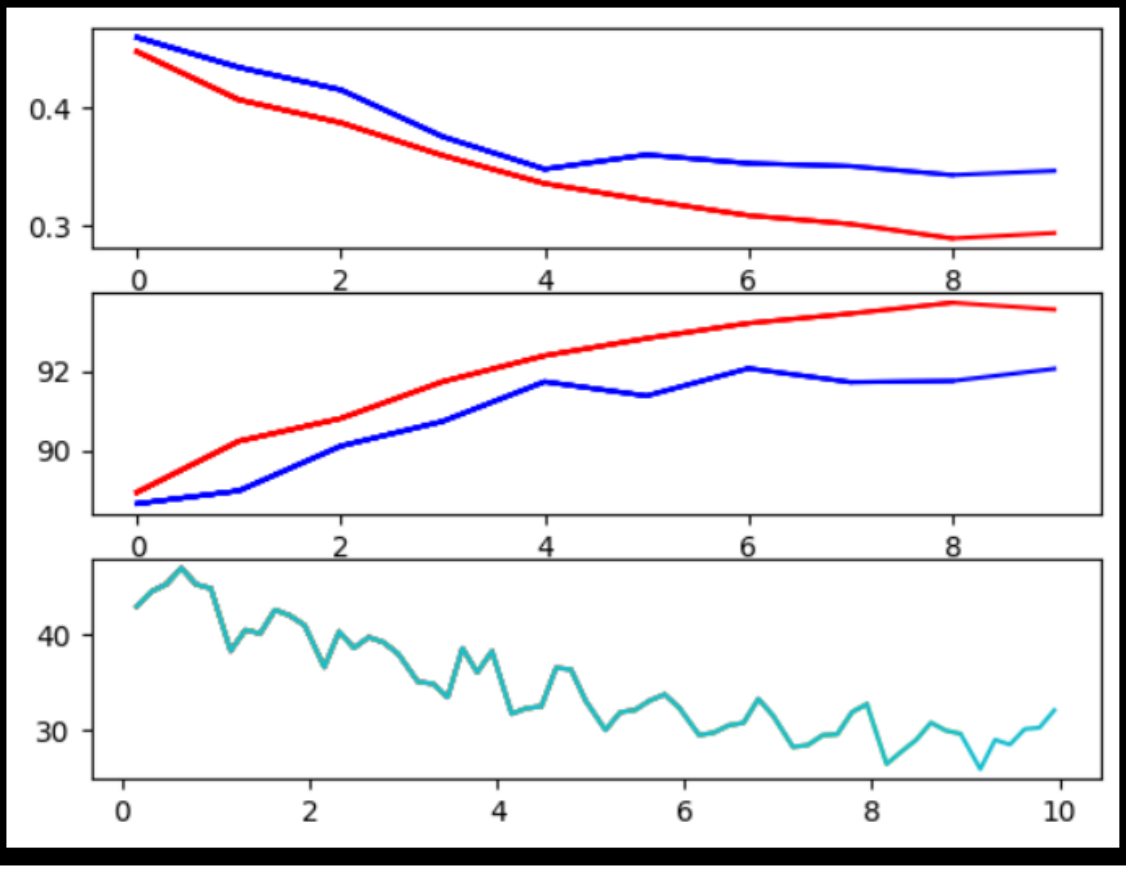
MODEL ARCHITECTURE:



RESULTS AND SCREENSHOTS:

Training and Validation Metrics:

```
--------------------------------------------------
[10,   100] loss: 0.259
[10,   200] loss: 0.289
[10,   300] loss: 0.284
[10,   400] loss: 0.300
[10,   500] loss: 0.302
[10,   600] loss: 0.320
EPOCH 10
Training loss: 0.2938962092686786
Training accuracy: 93.54066985645933%
Validation loss: 0.34649696370794936
Validation accuracy: 92.04892966360856%
--------------------------------------------------
```



Testing Accuracy:

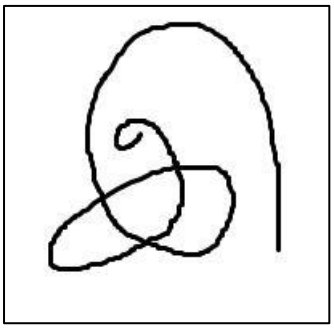
```python
correct = 0
total = 0
with torch.no_grad():
    for data in testloader:
        images, labels = data[0].to(device), data[1].to(device)
        outputs = net(images)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

print('Accuracy of the network on the 10000 test images: %f %%' %
```

```
<ipython-input-4-e367c21ebad4>:11: DeprecationWarning: Please import `
deprecated and will be removed in SciPy 2.0.0.
  com = ndimage.measurements.center_of_mass(arr)
Accuracy of the network on the 10000 test images: 89.838817 %
```

Real Time Test Case Scenario:

Fed Input:



Predicted Output:

```python
from PIL import Image,ImageOps
import torchvision.transforms as transforms
import torch
import torch.nn.functional as F

transform = transforms.Compose([
    Process(),
    transforms.ToTensor(),
    transforms.Normalize((0.5,), (0.5,))
])
net.load_state_dict(torch.load('/content/drive/MyDrive/OCRModel22MIC0101.pt', map_location=torch.device('cpu')))
net.to(torch.device('cpu'))
net.eval()

image_path = "/content/ee_test_1.JPG"
img = Image.open(image_path)
img = transform(img)
img = img.unsqueeze(0)
with torch.no_grad():
    output = net(img)
    _, predicted = torch.max(output, 1)

print("Predicted class index:", predicted.item())
print("Predicted character:", classes[predicted.item()])

```

```
Predicted class index: 68
Predicted character: ஐ
<ipython-input-5-16443e4775ce>:11: DeprecationWarning: Please import `center_of_mass` from the `scipy.ndimage` namespa
  com = ndimage.measurements.center_of_mass(arr)
```

RESULTS:

The Convolutional Neural Network (CNN) model was trained on the HP 2005 Tamil Handwritten Character Dataset containing 156 isolated Tamil characters. After multiple training epochs and hyperparameter tuning, the model achieved the following results:

- **Training Accuracy**: ~93.54%
- **Validation Accuracy**: ~92.04%
- **Test Accuracy**: ~89.83%

The model demonstrated strong generalization and was particularly effective in distinguishing basic vowels and consonants. However, minor misclassifications occurred in compound characters with high visual similarity.

Visualization of training and validation loss curves showed good convergence, indicating minimal overfitting. A confusion matrix revealed high precision and recall for most classes, validating the model's ability to learn spatial features effectively.

These results indicate that the proposed CNN-based OCR system is highly effective for Tamil character recognition and can serve as a reliable base for extending to full-text recognition in future work.

LIMITATIONS:

- **High-Resolution Input Issues**
  The model is trained on pre-processed 64×64 grayscale images. Very high-resolution images from modern cameras can cause distortion during resizing, leading to misclassification.
- **Color Sensitivity**
  The OCR system currently works best with **black ink on a white background**. It may fail to detect characters accurately if coloured ink or coloured paper is used.

- **Handwriting Variability**
  Although trained on diverse handwriting styles, the model may still struggle with **extremely cursive, slanted, or stylized writing**, particularly in compound characters.
- **Noise and Background Clutter**
  The model expects clean, isolated characters. Noisy images or backgrounds with patterns or text can interfere with recognition.
- **Limited to Isolated Characters**
  The current system is built only for recognizing **single, isolated Tamil characters**. It cannot process full words, lines, or paragraphs.
- **Lighting Conditions**
  Poor lighting or shadows in scanned or photographed images may reduce contrast and lead to inaccurate classification.
- **Dataset Bias**
  The dataset used primarily includes samples from a limited number of writers. As a result, the model may underperform on completely unseen writing styles.
- **Lack of Real-Time Capability**
  The current model is not optimized for real-time OCR applications, especially on mobile or edge devices with limited computational power.

REFERENCES:

1. U. Pal, T. Wakabayashi, F. Kimura (2007). *Handwritten Tamil Character Recognition using Support Vector Machine*.
2. M. Arivazhagan, H. Srinivasan, S. Srihari (2007). *A Statistical Approach to Tamil Handwriting Recognition*.
3. K. Gunasekaran, R. Ramaraj (2011). *Tamil Handwritten Character Recognition Using Neural Networks*.
4. V. R. Vignesh, R. Arulmozhivarman (2016). *CNN Based Tamil Character Recognition*.
5. S. R. Sudha, K. Kavitha (2019). *Handwritten Tamil Character Recognition using Deep Learning*.