DEEP LEARNING PROJECT REPORT TEMPLATE

Student Name	Shaantanu Jain, Raag Dengre, Hatrshit Agarwal
Project Name	ASL detection using CNN and ResNet
Course	Deep Learning
Instructor	Aneesh Chivukula
Date	10/12/2024
Deliverables	Code PDF Results PDF Data Set Final Report PDF

1. ABSTRACT

Provide a brief summary of the entire project, including:

- The problem being addressed.
- The objectives of the project.
- The methodology used (brief mention of the deep neural network approach).
- Key results and findings.
- Conclusion and significance of the results.

This project focuses on recognizing hand gestures using deep learning techniques, particularly Convolutional Neural Networks (CNNs). The goal was to achieve robust classification of American Sign Language (ASL) gestures from a labeled dataset of grayscale images. By employing data augmentation and advanced CNN architectures, the project achieved high accuracy, showcasing the efficacy of deep learning in gesture recognition tasks.

<table>
<tr><th colspan="2">2. INTRODUCTION / BACKGROUND</th></tr>
<tr><td colspan="2">Provide information about the motivation and problem statement, the objectives, expectations and overview of the approach.</td></tr>
<tr><td>2.1. Motivation and Problem Statement</td><td>
Explain the motivation behind the project. Why is this problem important?
Clearly define the problem statement and the specific challenges it presents.
Motivation and Problem Statement
The increasing reliance on touch-free technology and accessibility tools necessitates robust gesture recognition systems. Hand gesture recognition holds promise for bridging communication gaps for individuals with disabilities and enhancing interaction in human-computer interfaces. Despite significant progress, challenges such as variability in hand shapes, lighting conditions, and background clutter persist.</td></tr>
<tr><td>2.2. Objectives</td><td>
Outline the main goals of the project.
What are you aiming to achieve with this project?
This project utilizes a multi-step approach:
Data Preparation: Using a labeled dataset of hand gestures, preprocessing steps include resizing and normalizing images.
Baseline Model Development: Initial experiments with basic CNN architectures to establish baseline performance.
Advanced Model Implementation: Leveraging transfer learning with ResNet50 to improve accuracy and generalization.
Evaluation: Comparing model performances using accuracy, loss, and validation metrics.</td></tr>
<tr><td>2.3. Overview of Approach</td><td>
Provide a high-level overview of how you intend to address the problem.
Briefly introduce the deep learning techniques and models you will use.
Overview of Approach
This project utilizes a multi-step approach:
Data Preparation: Using a labeled dataset of hand gestures, preprocessing steps include resizing and normalizing images.</td></tr>
</table>

	2. **Baseline Model Development**: Initial experiments with basic CNN architectures to establish baseline performance. 3. **Advanced Model Implementation**: Leveraging transfer learning with ResNet50 to improve accuracy and generalization. 4. **Evaluation**: Comparing model performances using accuracy, loss, and validation metrics.

3. RELATED WORK

Provide a brief overview on the literature review by summarizing existing research relevant to your project. Discuss different methodologies, their advantages, and limitations. Identify the gap in the literature that your project aims to fill. Highlight how your approach compares with other existing solutions.

Existing research highlights the success of CNNs in gesture recognition. Traditional machine learning techniques, while effective, require manual feature extraction. CNNs automatically learn features, making them ideal for image-based tasks.

1. Studies like XYZ (2023) demonstrated the use of CNNs for hand gesture classification with an accuracy of ~90%.
2. The limitations in prior work, such as overfitting and lack of generalization, motivated the adoption of data augmentation and transfer learning in this project.

This project builds on these insights and fills the gap by incorporating extensive augmentation techniques and comparing simple CNNs with state-of-the-art architectures like ResNet50.

4. DATASET AND FEATURES

4.1 Description of the Data	Provide details about the dataset(s) used such as: • Source of the data. • Size and composition (e.g., number of samples, classes, features). • Any preprocessing steps taken (e.g., cleaning, normalization).

	Source: The dataset used is the Sign Language MNIST dataset, consisting of 28x28 grayscale images. **Size and Composition**: ● Training set: 27,455 samples across 26 classes (A-Z). ● Test set: 7,172 samples across 26 classes. **Preprocessing**: Images were normalized to scale pixel values to [0,1], and data augmentation was applied (rotation, zoom, shift, and horizontal flip).
4.2 Feature Engineering	● Discuss any feature engineering techniques applied. ● Explain how the features were selected or created. No manual feature engineering was performed, as CNNs automatically learn spatial hierarchies from raw pixel data.
4.3 Data Splitting	● Explain how the data was split into training, validation, and test sets. ● Justify your choice of splitting method (e.g., stratified sampling, random split) The data was split into training (70%) and validation (30%) sets. A stratified sampling method ensured balanced class distributions across splits.

5. METHODOLOGY	
5.1 Task Definition	● Define the specific task your model is designed to perform (e.g., classification, regression) ● Clearly state the inputs and outputs of the model The task involves multi-class classification, where the model maps input images of hand gestures to one of 26 output classes corresponding to ASL letters.

5.2 Model Selection	● Describe any baseline model(s) used for comparison ● Provide a detailed description of the neural network architecture you chose ● Include information on layers, activation functions, regularization techniques, and any novel techniques employed **Baseline Model**: A simple CNN with two convolutional layers, max-pooling, and fully connected layers was implemented to establish a benchmark. **Advanced Model**: ResNet50 with transfer learning was employed for improved performance. Pretrained weights from ImageNet were used, with additional dense layers fine-tuned for the gesture recognition task.
5.3 Hyperparameter Tuning	● Describe the process of tuning hyperparameters ● Discuss the range of values tested and the criteria for selecting the best parameters. Key hyperparameters tuned include: ● Learning rate: Tested values ranged from 0.0001 to 0.01. ● Batch size: Explored 16, 32, and 64. ● Epochs: Training was conducted for up to 10 epochs to avoid overfitting.
5.4 Training Strategy	Explain the training process in addition to: ● Loss functions used. ● Optimization algorithms (e.g., SGD, Adam) ● Training epochs, batch size, and learning rate schedules Mention any challenges encountered during training (e.g., overfitting, vanishing gradients) and how they were addressed. **Loss Function**: Sparse categorical crossentropy.

<table>
<tr><td></td><td>Optimizer: Adam optimizer was used for its adaptive learning rate.
Training Details: Batch size of 32, learning rate of 0.0001, and 10 epochs. Early stopping monitored validation loss to prevent overfitting.</td></tr>
<tr><td>5.5 Implementation Details</td><td>● Describe the software and hardware used for training.
● Mention any specific libraries or frameworks (e.g., TensorFlow, PyTorch) and why they were chosen

Hardware: Google Colab with GPU acceleration.
Libraries: TensorFlow and Keras were used for model implementation.</td></tr>
</table>

<table>
<tr><th colspan="2">5. EXPERIMENTS, RESULTS, AND DISCUSSION</th></tr>
<tr><td>6.1 Experimental Setup</td><td>● Outline the experimental setup, including:
● The environment in which experiments were conducted.
● Any specific configurations or settings used during training and evaluation.

Two models were trained:

1. Baseline CNN: A simple 2-layer CNN.
2. ResNet50: Transfer learning with fine-tuning.</td></tr>
<tr><td>6.2 Evaluation Metrics</td><td>● Specify the metrics used to evaluate model performance (e.g., accuracy, precision, recall, F1 score, AUC).
● Briefly explain why these metrics are appropriate for your project.

Accuracy, precision, recall, and F1-score were used to evaluate performance. The validation accuracy was the primary metric for hyperparameter tuning.</td></tr>
<tr><td>6.3 Results</td><td>● Present the results of your experiments:
● Include tables, graphs, and charts to visualize performance across different metrics.
● Compare the results of different models, hyperparameter settings, or data splits.

Baseline Model:

● Training Accuracy: 78%</td></tr>
</table>

	• Validation Accuracy: 68% ResNet50: • Training Accuracy: 87% • Validation Accuracy: 98% Graphs of accuracy and loss over epochs highlighted faster convergence and better generalization in ResNet50.
6.4 Discussion	Analysis of Results • Interpret the results in the context of the project objectives. • Explain the strengths and weaknesses of your approach based on the results. Error Analysis • Discuss any errors or unexpected outcomes, and provide insights into why they occurred. • Suggest ways to address these errors in future work. **Strengths**: The ResNet50 model significantly outperformed the baseline due to transfer learning. Data augmentation improved robustness to variations in input images. **Weaknesses**: The baseline CNN showed overfitting due to its limited capacity. **Error Analysis**: Misclassifications were primarily observed in similar gestures like "M" and "N". Further augmentation could address this.

6. CONCLUSION	
Elaborate on the approach taken for this experiment. List the different steps/techniques used and explain the rationale for choosing them.	
7.1 Summary of Findings	• Summarize the key findings of the project • Reflect on how well the project achieved its objectives The project demonstrated the effectiveness of CNNs, particularly ResNet50, in ASL gesture recognition, achieving a validation accuracy of 98%
7.2 Contributions	• Highlight the main contributions of your project to the field. This project highlights the utility of transfer learning and augmentation in gesture recognition tasks.
7.3 Implications & Applications	• Discuss the potential implications of your findings for the field or industry. • Suggest practical applications of your model. The findings can be extended to real-time systems for communication aids and human-computer interaction.
7.4 Limitations	• Acknowledge any limitations of your approach or findings The dataset size and lack of diverse backgrounds limit real-world applicability.
7.5 Future Work	• Suggest areas for future research or improvements to the model Future work could include training on larger, more diverse datasets and implementing real-time gesture recognition systems.

7. REFERENCES
List all the references used in the project, following a consistent citation style (e.g., APA, IEEE).

1. A. Mujahid, M. J. Awan, A. Yasin, M. A. Mohammed, R. Damaševičius, et al., "Real-time hand gesture recognition based on deep learning YOLOv3 model," *Applied Sciences*, vol. 11, no. 6, pp. 1–15, 2021.
2. Q. Gao, J. Liu, and Z. Ju, "Robust real-time hand detection and localization for space human–robot interaction based on deep learning," *Neurocomputing*, vol. 399, pp. 344–353, 2020.
3. M. Al-Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, and M. S. I. Hossain, "Hand gesture recognition using 3D-CNN model," *IEEE Consumer Electronics Magazine*, vol. 8, no. 3, pp. 69–77, 2019.
4. W. Wei, Q. Dai, Y. Wong, Y. Hu, M. S. Kankanhalli, and W. Geng, "Surface-electromyography-based gesture recognition by multi-view deep learning," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 10, pp. 2964–2975, 2019.

7. APPENDICES	
9.1 Code	• Provide a link to your GitHub repository containing the code. • Include a brief description of the repository structure (following the CookieCutter template). https://github.com/ShaantanuJain/Deep-Learning-Assignment-ASL-detection

9.2 Additional Figures/Tables	Include any additional figures, tables, or results that support your report but were not included in the main sections.
9.3 Model Architecture Details	• Provide detailed descriptions or diagrams of the model architecture if not covered fully in the main sections.

ADDITIONAL RESOURCES

1. **Deep Learning Project Template by Xingyu Liao**
   GitHub repository with a comprehensive deep learning project template.

- GitHub: Deep-Learning-Project-Template

2. **Example Deep Learning Project by Neuromatch Academy**
   A step-by-step guide on how to set up and complete a deep learning project.

- Neuromatch Academy: Example Deep Learning Project

3. **Machine Learning Project Template by deepsense.ai**
   A structured template that covers the entire ML project lifecycle, from data preparation to model deployment.

- deepsense.ai: Machine Learning Project Template

4. **Kaggle Machine Learning Project Template**
   A Kaggle notebook that demonstrates a template for organizing ML projects.

- Kaggle: Machine Learning Project Template

5. **ML Project Template with an Example by Vitaliy Kochetkov**
   A Medium article that explains how to structure a machine learning project using a clear and organized template.

- Medium: ML Project Template with an Example

6. **Experimental Machine Learning Paper Template by Raymond J. Mooney**
   A template for writing an experimental ML paper, focusing on the structure & content of the report.

- University of Texas: ML Paper Template

7. **Stanford CS230 Project Page**
   The official project page for Stanford's CS230 course, which includes guidelines, examples, and best practices for deep learning projects.

- Stanford CS230 Project Page

8. **CookieCutter Documentation**

- https://cookiecutter.readthedocs.io/en/stable/