Guidelines for CS428 Mid-term Presentation

This document provides a comprehensive guideline for the mid-term presentation. You are expected to experiment with various prebuilt deep learning models, datasets, and architectures to perform hands-on tasks and present their findings. Each student will present for 15 minutes, explaining their chosen dataset, model, preprocessing techniques, results, and key observations.

1. Dataset Selection and Preparation

You are encouraged to select a dataset relevant to their area of interest. Some suggested datasets include:

- **Hand Gesture Recognition Dataset**: Contains images of various hand gestures for classification tasks.
- **T-LESS Dataset**: A dataset for detecting and recognizing texture-less 3D objects, typically used in industrial applications.
- **Custom Dataset**: You may choose or create their own dataset based on their project goals or interests.

### Guidelines for Dataset Preparation:
- **Resizing**: Resize the images to match the input size expected by the model (e.g., 224x224 for MobileNet).
- **Grayscaling**: Convert the images to grayscale if needed, especially for datasets where color information is not crucial.
- **Data Augmentation**: Use augmentation techniques like random rotation, flipping, and scaling to increase the variety of the training data. This can help improve model robustness and generalization.
- **Normalization**: Scale pixel values to a range of [0, 1] or normalize using dataset-specific statistics (e.g., mean and standard deviation).

Example code for preprocessing:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
import cv2

# Example: Preprocessing an image
def preprocess_image(image_path, target_size=(224, 224), grayscale=False):
    # Load and resize the image
    img = load_img(image_path, grayscale=grayscale, target_size=target_size)
    img = img_to_array(img)  # Convert to numpy array

    # Normalize the image
    img /= 255.0

    # Expand dimensions to match model input shape
    img = np.expand_dims(img, axis=0)

    return img

# Example: Data Augmentation
datagen = ImageDataGenerator(
    rotation_range=20,        # Randomly rotate images by 20 degrees
    width_shift_range=0.2,    # Randomly shift images horizontally by 20%
    height_shift_range=0.2,   # Randomly shift images vertically by 20%
    shear_range=0.2,          # Shear the image
    zoom_range=0.2,           # Zoom in/out on the image
    horizontal_flip=True,     # Randomly flip images horizontally
    fill_mode='nearest'       # Fill missing pixels with nearest value
)
```

2. Model Selection and Usage

You can choose from a variety of prebuilt models, such as ResNet, MobileNet, Inception, and more. Each model has unique characteristics and strengths. You should explore and compare different models based on their specific dataset and task. Use transfer learning if appropriate.

### Guidelines for Model Selection and Usage:
- **Input Shape**: Ensure the input shape matches the dataset dimensions (e.g., (224, 224, 3) for RGB images).
- **Number of Classes**: Modify the output layer of the model to match the number of classes in the dataset.
- **Optimizer**: Choose an appropriate optimizer such as Adam, SGD, or RMSProp. Try adjusting learning rates and observe the impact.
- **Loss Function**: Use categorical cross-entropy for multi-class classification or binary cross-entropy for binary classification.
- **Metrics**: Track metrics such as accuracy, precision, recall, and F1-score.

Example code for loading a prebuilt model and modifying it:

```
from tensorflow.keras.applications import MobileNetV2, ResNet50
from tensorflow.keras import layers, models

# Example: Load a prebuilt MobileNetV2 model
base_model = MobileNetV2(input_shape=(224, 224, 3), include_top=False,
weights='imagenet')

# Freeze the base model layers
base_model.trainable = False

# Add custom layers for fine-tuning
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(1024, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(10, activation='softmax')  # Output layer for 10 classes
])

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

3. Experimentation and Observations

### Guidelines for Experimentation:

- Try different datasets, models, and architectures.
- Experiment with different optimizers and learning rates.
- Observe the effect of data augmentation and preprocessing on model performance.
- Track and visualize training and validation accuracy/loss using tools like matplotlib or TensorBoard.

You should document their key observations, such as:
- How does the model perform on different datasets?
- What happens when you change the optimizer or learning rate?
- How does data augmentation impact the model's performance?
- What are the main challenges faced and how were they addressed?

4. Presentation Format

Each student is expected to present for 15 minutes. The presentation should cover the following points:

1. **Introduction and Motivation**: Explain why you chose this dataset and model. What problem are you trying to solve?
2. **Dataset Preparation**: Describe the preprocessing steps, data augmentation techniques, and any challenges encountered.
3. **Model Selection and Architecture**: Explain the model you chose and any modifications made. Why did you choose this particular model?
4. **Experimentation**: Describe the experiments conducted and the results observed. How did changing hyperparameters affect the performance?
5. **Key Observations and Contribution**: Summarize your key findings. What insights did you gain from these experiments?
6. **Conclusion**: What were the main takeaways from your experiments? How would you improve the model or experiments in the future?

You should use visual aids like plots, charts, or diagrams to illustrate their results and support their findings.

5. Additional Tips

Here are some additional tips to help you with your presentation:
- **Use Visualizations**: Include graphs and charts to show training and validation metrics, such as loss and accuracy over epochs.
- **Be Clear and Concise**: Explain your choices and results clearly. Avoid using too much technical jargon without explanation.
- **Time Management**: Make sure to practice your presentation to ensure it fits within the 15-minute timeframe.
- **Prepare for Questions**: Be ready to answer questions about your experiments, choices, and observations.