Final Report

Data Preparation Process / Preprocessing

1. Data Loading:

The data preparation process began with loading video files from the specified directories for training and validation. Each video file was associated with a label based on its class, extracted from the file name.

2. Transformations:

- **Resizing to 128x128 pixels:**
    - **Reason:** The frames were resized to 128x128 pixels to standardize the input size for the model. This uniformity ensures that the model receives consistent input dimensions, which is crucial for training.
    - **Advantages:**
        - **Computational Efficiency:** Smaller frames reduce the computational load, allowing faster processing and training.
        - **Memory Management:** Smaller frame sizes require less memory, enabling the use of larger batch sizes and reducing the risk of running out of GPU memory.
        - **Consistency:** Ensures that all videos, regardless of their original resolution, are treated uniformly.
- **Converting Frames to Tensor Format:**
    - **Reason:** Conversion of frames to tensor format is necessary because PyTorch models operate on tensor data. Tensors are the primary data structure used in PyTorch for building and training neural networks.
    - **Advantages:**
        - **Compatibility:** Tensors are compatible with PyTorch's data loading, model building, and training processes.
        - **GPU Acceleration:** Tensors can be easily transferred to and processed on GPUs, leveraging hardware acceleration for faster computation.
        - **Automatic Differentiation:** PyTorch's autograd system requires tensor inputs to compute gradients during backpropagation.

3. Frame Handling:

- **Conversion from BGR to RGB:**
    - **Reason:** OpenCV loads images in BGR format by default, whereas most deep learning models, including those in torchvision, expect images in RGB format.
    - **Advantages:**
        - **Standardization:** Ensures that the input data conforms to the standard expected by pre-trained models and common image processing libraries.
        - **Accuracy:** Prevents potential errors in color representation that could negatively impact model performance.

Models and Techniques

Model Used: r3d_18

- **Reason:** The `r3d_18` (ResNet3D-18) model is specifically designed for video classification tasks, extending the 2D convolutional architecture of ResNet to 3D convolutions.
- **Advantages:**
    - **Temporal Dynamics:** 3D convolutions in `r3d_18` allow the model to capture both spatial and temporal features, making it well-suited for video data where motion and sequence information are crucial.
    - **Pre-training:** The `r3d_18` model comes pre-trained on the Kinetics dataset, a large-scale action recognition dataset. This pre-training provides a good starting point and accelerates convergence during fine-tuning.
    - **Proven Architecture:** The ResNet architecture, on which `r3d_18` is based, is a proven and widely used model that effectively handles vanishing gradient problems through residual connections.

Video Classification Performance Metrics for Training and Validation

Performance Metrics Used: Accuracy

- **Reason:** Accuracy is a straightforward metric that measures the proportion of correctly classified samples out of the total samples. It is often used as a primary metric in classification tasks.
- **Advantages:**
    - **Simplicity:** Accuracy is easy to compute and interpret. It provides a clear indication of the model's overall performance.
    - **Relevance:** For balanced datasets where all classes are equally important, accuracy is a suitable metric.
    - **Baseline Metric:** Accuracy serves as a good baseline metric, which can be complemented with other metrics if needed.

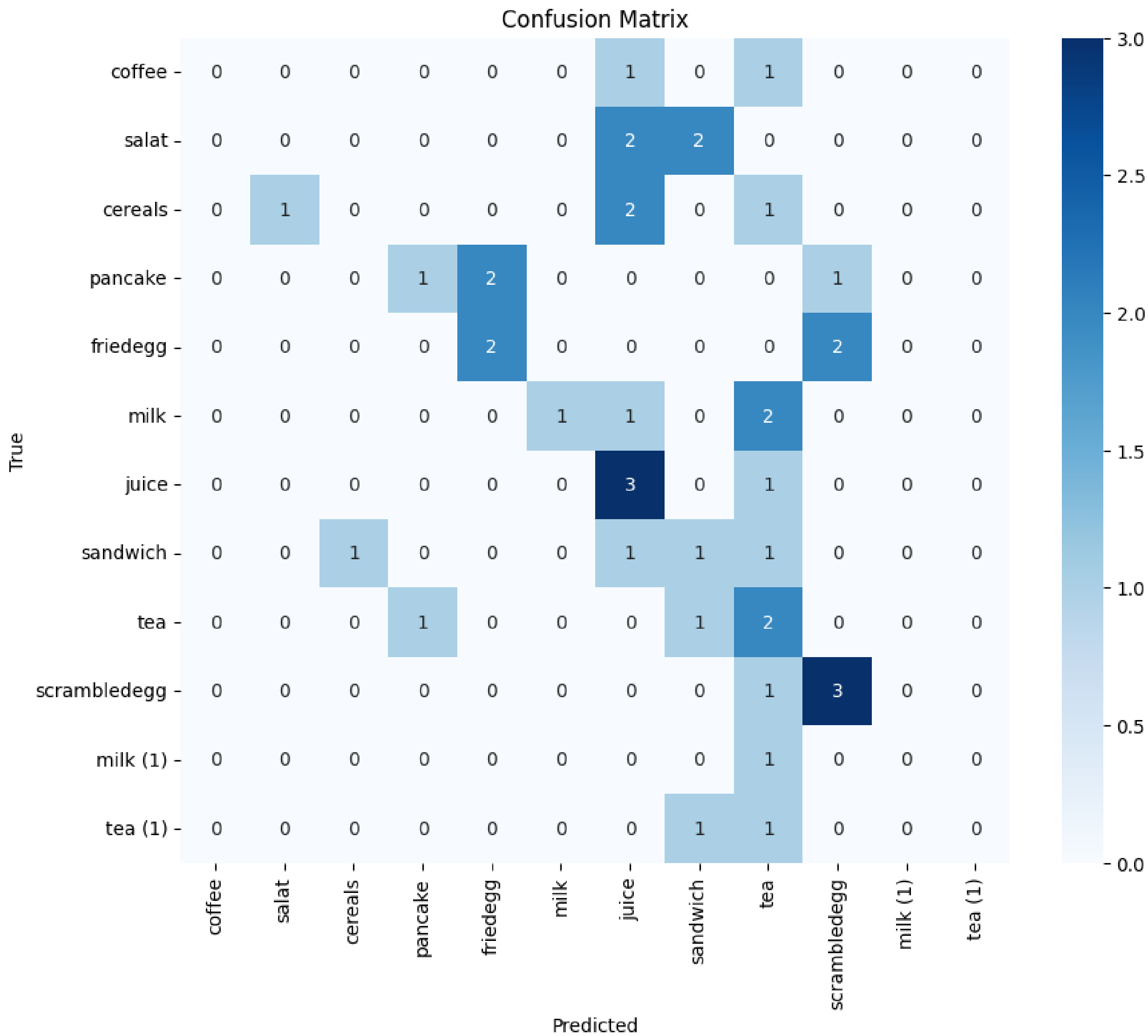
Other Considered Metrics:

- **Precision, Recall, and F1-Score:** These metrics are particularly useful when dealing with imbalanced datasets, as they provide a more nuanced view of the model's performance on each class.
- **Confusion Matrix:** Provides a detailed breakdown of true positives, false positives, true negatives, and false negatives, helping to identify specific areas where the model may be misclassifying.

Training and Testing Times

- **Training Time for the r3d_18 Model:** 43 minutes
- **Testing Time for the r3d_18 Model:** 17 seconds

Screen Shot for Confusion Matrix :



Screen Shot For Validation Accuracy :

Validation Accuracy

```
[ ]  validation_accuracy = accuracy_score(all_labels, all_preds)
     print(f'Validation Accuracy: {validation_accuracy:.4f}')
```

```
Validation Accuracy: 0.3171
```