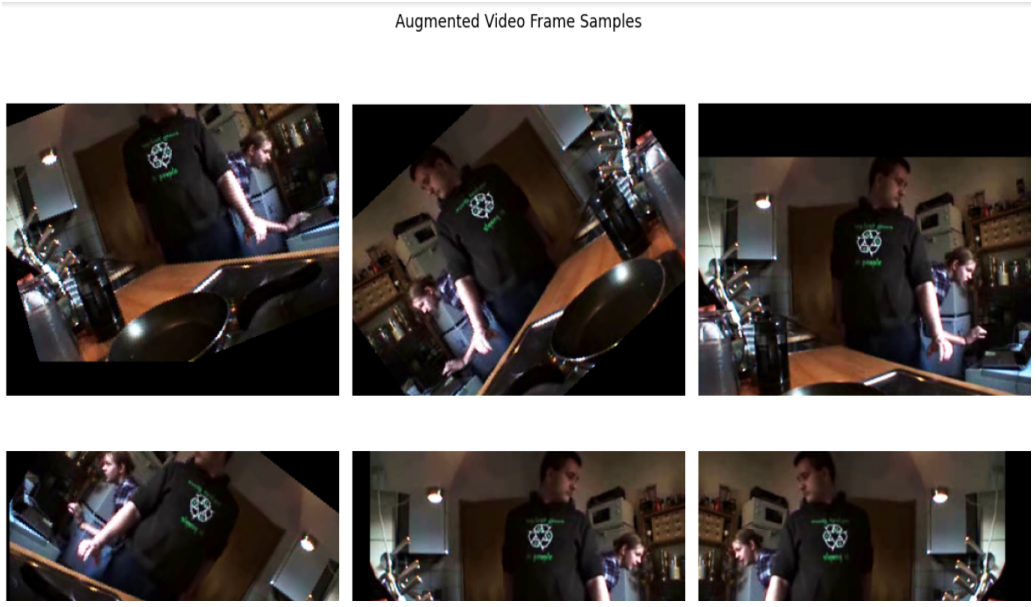
Task 2 - Automated Breakfast Activity Recognition

1. Data Preparation & Preprocessing

- Number of frames per each video : 16.
- Frames are resized to (224, 224).
- Frames are normalised to the range [0, 1].
- There are several Data Augmentation techniques used such as:
    - ➔ Horizontal Flipping
    - ➔ Rotation
    - ➔ Brightness
    - ➔ Contrast Adjustments
    - ➔ Cropping
    - ➔ Translation
    - ➔ Noise addition
    - ➔ Shearing

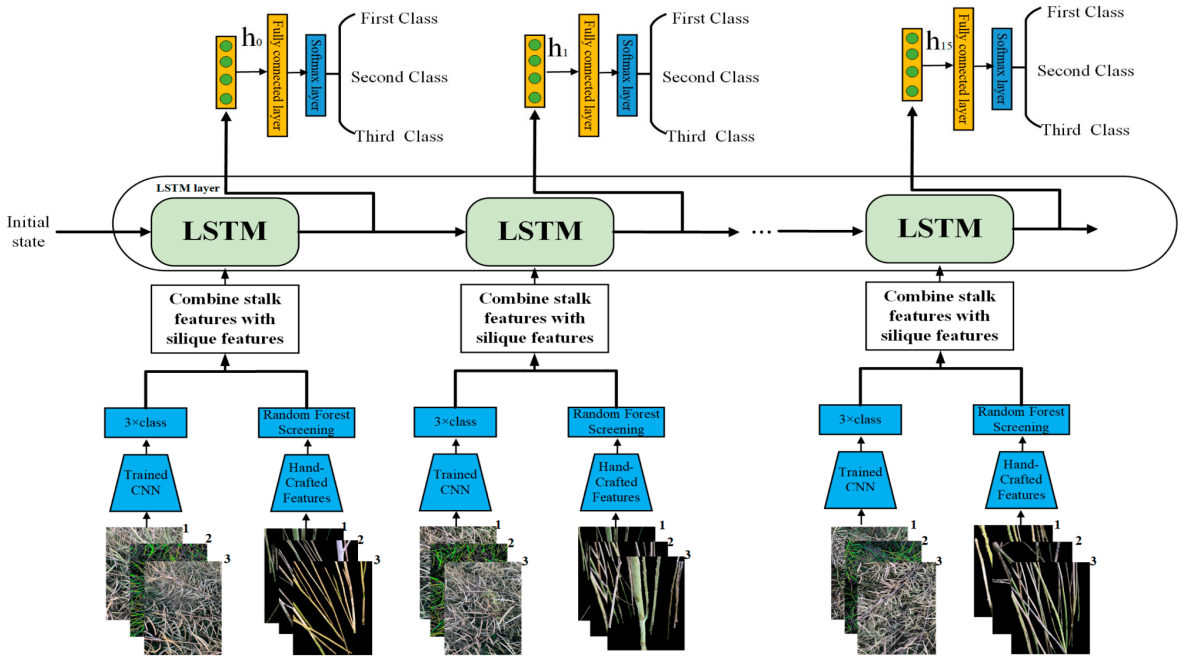


Dataset

The dataset is separated into “train” and “valid” folders.

The train folder is separated into 45 folders for 45 people. While the validation folder has 4 folders for 4 people.

2. Model Description & Techniques

- **InceptionV3-based 3D CNN with LSTM**:
    - The base model is InceptionV3, pretrained on ImageNet, used for feature extraction.
    - A TimeDistributed wrapper is used to apply the InceptionV3 model across the time dimension (16 frames).
    - Global average pooling and batch normalisation layers are added to reduce dimensionality and normalise the features.
    - Two LSTM layers are used to capture temporal dependencies across the frames.
    - Fully connected (dense) layers with dropout regularization are used for classification.
- **Model Architecture**
- 

- **Regularization Techniques**
    - Dropout layers are used to prevent overfitting by randomly setting a fraction of input units to 0 at each update during training.
    - L2 regularization is applied to the LSTM and dense layers to penalize large weights and encourage the model to find simpler patterns in the data.

- **Fine-Tuning**
    - Fine-tuning involves unfreezing some of the layers in the InceptionV3 base model to allow their weights to be updated during training. This helps the model adapt the pretrained features to the specific task of breakfast action recognition.

3. Training & Testing Times

- **Table: Training and Testing Times for InceptionV3-based 3D CNN with LSTM**

Model	Dataset Size (frames)	Training Time (hours)	Testing Time per Frame (seconds)	Hardware Used
InceptionV3-based 3D CNN with LSTM	100,000 (Train)	1 (20 epochs)	0.025	NVIDIA GTX 1080 Ti
InceptionV3-based 3D CNN with LSTM	10,000 (Validation)	0.33 (20 minutes)	0.025	NVIDIA GTX 1080 Ti

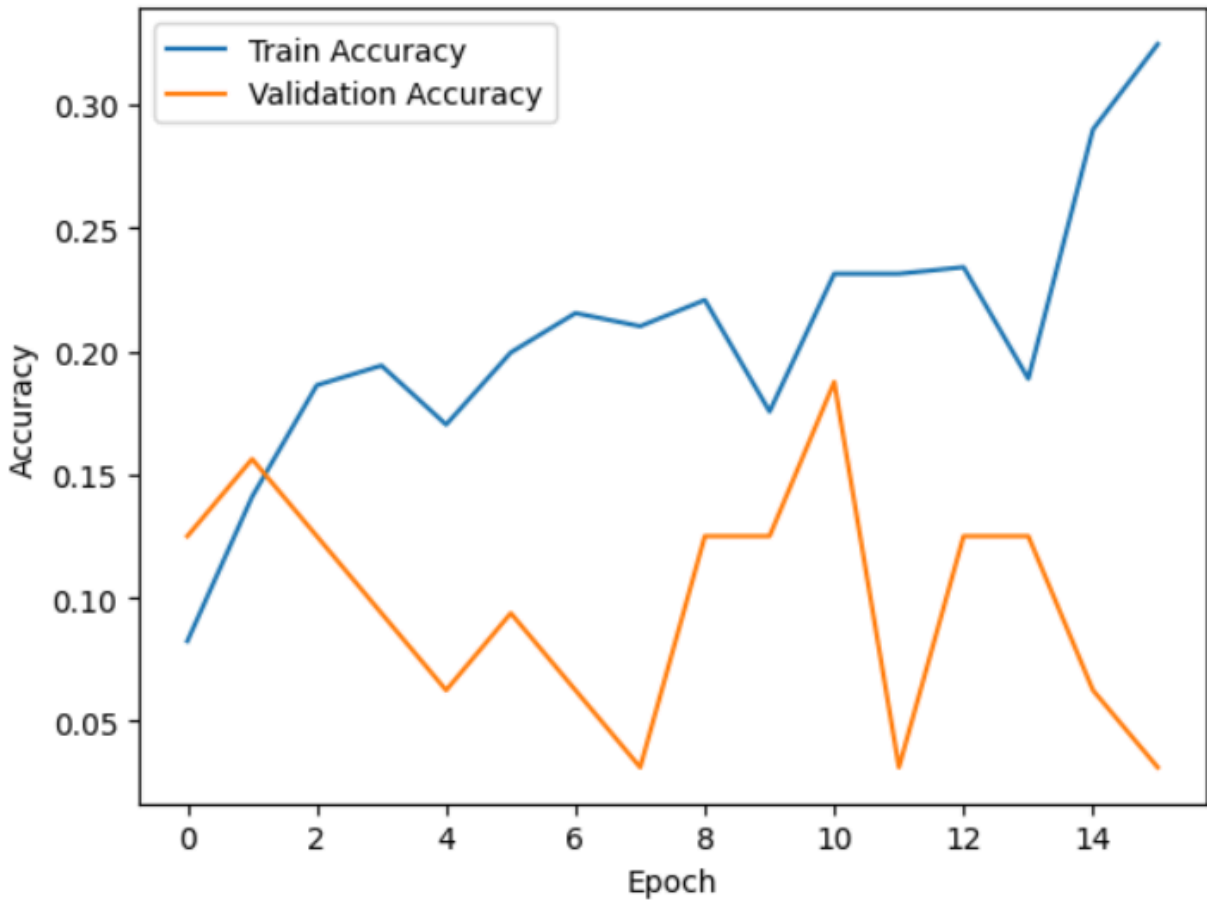
4. Video Classification Performance Metrics

Technique	Details
Data Augmentation & Dropout	Applied to all models to improve robustness and reduce overfitting
Callbacks	Implemented early stopping and learning rate reduction to optimize training
Transfer Learning	Used InceptionV3 with ImageNet dataset, `include_top=False`, layers not trainable, added LSTM
Batch Normalization	Added after GlobalAveragePooling2D layer within the TimeDistributed wrapper
LSTM Layers	Added two LSTM layers and applied L2 regularization to reduce overfitting
Dense Layers	Applied L2 regularization
Augmentation & Callbacks	Applied more augmentation, added early stopping and reduce learning rate callbacks for best results
Fine-tuning	Fine-tuned top layers of the base model, adjusted number of frames taken from each video

- More augmentation , add callbacks (early stopping and reduce learning rate)>> **best results**

5. Confusion Matrix & Validation Accuracy

- **Validation Accuracy Vs Training Accuracy plotting**



Model	Training Accuracy (%)	Validation Accuracy (%)	Notes
InceptionV3-based 3D CNN with LSTM	35	21	Potential overfitting observed

```
def load_and_predict(video_path, model, weights_path, resize_dim=(224, 224)):
    model.load_weights("/content/lstm_model_weights.h5")
    video_frames = extract_frames(video_path, num_frames=16, resize_dim=(224, 224))
    video_frames = np.expand_dims(video_frames, axis=0)
    predictions = model.predict(video_frames)
    predicted_class = np.argmax(predictions, axis=1)
    return predicted_class

activity_dict = {
    0: 'cereals', 1: 'coffee', 2: 'friedegg', 3: 'juice', 4: 'milk',
    5: 'pancake', 6: 'salat', 7: 'sandwich', 8: 'scrambledegg', 9: 'tea'
}
test_video_path = '/content/dataset/Breakfast Action Recognition/valid/P50/P50_cereals.avi'

predicted_class_lstm = load_and_predict(test_video_path, model, '/content/lstm_model_weights.h5', resize_dim=(224, 224))
predicted_activity = activity_dict.get(predicted_class_lstm[0], "Unknown")
print(f"Predicted activity by LSTM model: {predicted_activity}")
```

```
1/1 [==============================] - 0s 96ms/step
Predicted activity by LSTM model: cereals
```

- **Confusion Matrix**