Andreas Alexnadrou(9503978) – Sotiris Zenios(8657785) (Group nr: 23 )

# I. ARCHITECTURE CHOICES

*Explain briefly the architectural choices for your frame, optical flow and two-stream networks. Use model.summary(). Discuss layers and hyperparameters. Make sure you discuss the type of fusion in the two-stream network.(two pages)*

## 1. Frame CNN architecture:

Architecture choice:

For the task of action recognition, a ResNet-18 architecture was chosen for the frame-based CNN. The reason we chose this architecture is because ResNet-18 is part of the residual family, and it has the ability to utilize skip connections to avoid vanishing gradient problem allowing for a deeper network that can learn robust features. The adaptation of ResNet-18 for our task involved modifying the final fully connected layer to output 12 classes, corresponding to the number of action classes in the Stanford 40 dataset. Also, ResNet-18 is pre-trained on ImageNet with 1000 classes that help our model to learn the new images easier. Furthermore, the reason for ResNet-18 choice instead of e.g. ResNet-50 is because this model is simpler, and the chances of overfitting are less.

Layers:

To give a deep dive in the layers of the ResNet-18, the input layer accepts images of size 224x244. The ResNet-18 is composed of a series of residual blocks that are connected with skip connections. Each block contains a convolutional layer (e.g., Conv2d-5) followed by batch normalization and Relu activation. Then a there is a second convolutional layer (e.g., Conv2d-8) that again is followed by batch normalization (e.g., BatchNorm2d-9). The network increases the number of filters while reducing the spatial dimension as it goes deeper. At the end of the network, an adaptive average pooling layer (AdaptiveAvgPool2d-67) reduces each feature map to a single value, essentially summarizing the salient features detected in each filter. The result is a 1D tensor of length equal to the number of filters in the final convolutional layer (512). Finally, the output layer transforms the 512 features into 12 output classes, corresponding to the dataset's actions.

Hyperparameters:

For the hyperparameters we used Adam optimizer due to its adaptive learning rate capabilities, which help converge fast and stable in comparison to other e.g. SGD. The learning rate used was 0.001 as a staring point, balancing the need for fast learning without overshooting minima in the loss landscape. As a loss function, we used CrossEntropyLoss was utilized as a loss function due to its effectiveness in multiclass classification. The epochs used was only 6 to prevent overfitting the model.

## 2. Optical flow CNN architecture:

Architecture choice:

For the analysis of optical flow data within the HMDB51 dataset, we adapted the pretrained ResNet-18 architecture to accept optical flow inputs. For the input of network is 8 optical flows with size 224x224. Optical flows are stacked together and represent the horizontal and vertical displacement of pixel intensity across consecutive frames. ResNet-18, pretrained on ImageNet, has learned a wide array of visual features. We used a pretrained model of ResNet-18 because even though it learned from static images, this can be transferable and beneficial for the task of motion recognition, as motion can be considered as a change in the appearance over time. Using ResNet-18 that wasn't pretrained had worse results.

Layers:

The model uses the same layers as the frame CNN architecture (ResNet-18) that I discussed before. The only differences is that the model take optical flow as an input (224x224x8) and the first convolutional layer (self.base_model.conv1) accepts this input. Another adaptation is the replacement of the original fully connected layer in ResNet-18 with an identity mapping (self.base_model.fc = nn.Identity()). The identity function is used to pass the output of the convolutional base directly to a new linear layer without any transformation. This allows for a more seamless integration with the newly added fully connected layer (self.fc = nn.Linear(num_ftrs, num_classes)), which is responsible for the final classification into the number of action classes in the dataset.

Hyperparameters:

The hyperparameters for the optical flow CNN are also similar to the frame CNN hyperparameters. We used Adam optimizer that resulted in better results than other methods, learning rate of 0.001 that made incremental improvements on the pretrained weights, and Cross – Entropy loss. The model was trained for 6 epochs , providing a balance between underfitting and overfitting for a model of this complexity.

## 3. Combination into two-stream network:

Architecture choice:

The frame model is based on a modified ResNet18 architecture, where the last fully connected layer is adapted to the specific number of classes in the target dataset and pre-trained weights are optionally loaded. The second stream, the flow model, processes motion cues captured by optical flow fields between frames. Finally, the outputs from both streams are fused using a fully connected layer, which concatenates the features extracted from both the spatial and temporal networks to make a final prediction (Late Fusion). The combined feature vector is then classified using a linear layer that maps the concatenated features to the predicted class labels.Averaging and Max Techniques were used but didn't produce better results.

Layers:

The model uses the networks mentioned in 1) and 2) that we discussed before. We integrate newly added fully connected layer (self.frame_model.fc = nn.Linear(num_ftrs_frame, num_classes)) and a fully

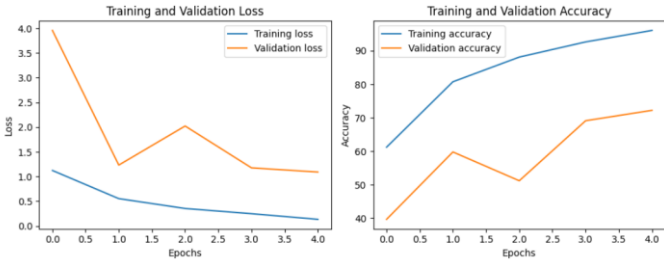
connected layer after those 2, that essentially fuses them (self.fc_fusion = nn.Linear(num_classes * 2, num_classes)).

Hyperparameters:

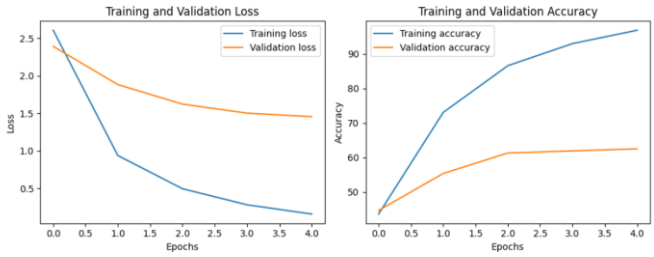
The hyperparameters for the optical flow CNN are also similar to the frame CNN hyperparameters. We used Adam optimizer that resulted in better results than other methods, learning rate of 0.01 that made incremental improvements on the pretrained weights, and Cross – Entropy loss. The model was trained for 20 epochs, as it needed more time to learn the combined features of the frame and optical flow images.

## II. Results

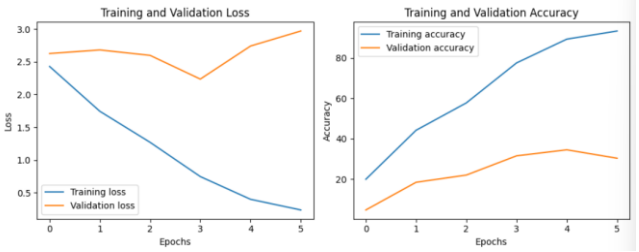
### A. Stanford 40 Frames



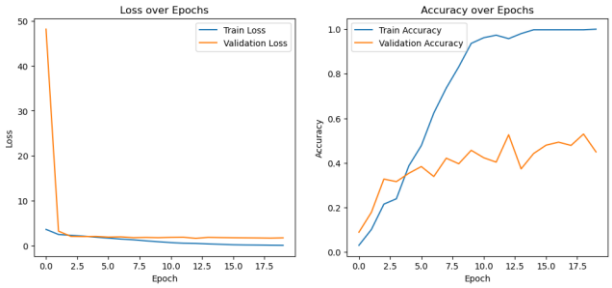

### B. HMDB51 Frames




### C. HMDB51 Optical flow




### D. HMDB51 Two-stream




### E. Result table

*Fill in the following table. Use torchsummary to obtain the number of parameters.*

TABLE I. SUMMARY OF YOUR RESULTS PER MODEL

| Dataset | Model specifications | | | |
|---|---|---|---|---|
| | Model | Top-1 accuracy (train / test) | Top-1 loss train | Model parameters |
| Stanford 40 | Frames | 0.960 / 0.70 | 0.133 | 11,182,668 |
| HMDB51 | Frames | 0. 968/0.56 | 0.157 | 11,182,668 |
| HMDB51 | Optical flow | 0.933/0.25 | 0.2368 | 11,198,348 |
| HMDB51 | Two-stream | 0.99/0.24 | 0.08 | 22,381,316 |

## III. Discussion of Results

*Discuss the performance of your models, and compare the results between models. Explicitly discuss the training procedure, and risk of overfitting. (one column)*

The Stanford 40 Frames model achieved a high training accuracy of 96% but suffered a significant drop to 70% in test accuracy, showing a potential overfitting.For this reason, the choice tasks of data augmentation and cyclical learning rate were used to improve the results. During transfer learning in the 2nd task , the HMDB51 Frames model remained at the same training accuracy at 96.8% but a lower test accuracy of 56%, suggesting similar generalization challenges. Again, we applied data augmentation to improve this performance. The HMDB51 Optical Flow model, which focuses on temporal features by capturing the flow with cv2' function v2.calcOpticalFlowFarneback, showed some overfitting with a test accuracy of only 25%, but despite the a decent training accuracy of 93.3%. The test accuracy though can be described as sufficient as we only capture movements in the flow images saved. This indicates that while it captures movement well, but doesn't effectively integrate this with spatial information.

The Two Stream model, combines spatial and temporal data and therefore needed significantly more epochs than the other models to be trained.It achieved a highest training accuracy of 99%. However, it also exhibited severe overfitting, evidenced by a test accuracy of just 24% and validation accuracy of 54% , when it was expected to exceed the performance of model 2 as it was using spatial information from optical flow model. Its training might be too narrowly focused on the training dataset specifics, leading to poor performance in general test data. Simpler initial models would probably benefit and reduce the risk of overfitting, as the dataset samples differ greatly from each other, making our models based on the ResNet18 structure to adapt specifically to the features of each individual sample.In addition, mid fusion techniques would be promising to improve the performance.

## IV. Link to Model Weights

*Make sure your link is accessible* form.ning@uu.nl *and* r.w.poppe@uu.nl

model weights

## V. CHOICE TASKS

*Mention choice tasks, and include the results (when applicable) or a motivation how you implemented the choice tasks. (one page)*

***CHOICE 1: Do data augmentation for Stanford-40:***

For this choice task data augmentation was implemented on the Stanford40 dataset. For this we used 3 techniques. First, we used Random Horizontal Flips in order to mirror images on the vertical axis. Since the actions in the dataset, such as jumping or drinking, can occur in either horizontal orientation, this augmentation helps the model become invariant to the direction of the action. Secondly, we use Random Rotation by rotating the images by a small angle to be able to capture the same action in different angles Finaly, Color Jitter was used to modify the brightness, contrast, saturation and hue of the images to add robustness against changes in lighting and color variations, that may occur. The use of data augmentation techniques led to an increase in training loss (0.3040) and a decrease in training accuracy (90.16%), which is expected as the model now trains on a more varied and challenging set of images. However, this complexity helps in reducing the model's overfitting to the training data, which is evident from the increase in test accuracy from 70% to 75% despite a slight drop in validation accuracy (66.73%).This shows that the model has benefited from data augmentation in a way that the model generalizes more and is more prepared to handle real-world images.

***CHOICE 2: Do data augmentation for HMDB51:***

For this task effective data augmentation techniques were used to handle motion aspects in the images. We used 3 techniques to do that. First, we use Random Resized Crop for scale between 80% and 100% allows the model to focus on slightly different regions, simulating a form of zoom and shift that could occur during video capture. Also, Random Horizontal Flips techniques were used to that is relevant for actions that are horizontally symmetrical or where the direction of action is not crucial to its classification, such as walking or running, that introduces invariance to directionality in the training data. Furthermore, small rotations (between -10 and +10 degrees) introduce variations in the camera angle, a common occurrence in real-world scenarios, helping the model to generalize better across different filming conditions. Using these techniques ,we get a training loss (0.66) and accuracy (79.76%) dropped, which is typical when a model is exposed to a more challenging and varied set of training data. However, this strategy improved the validation and test accuracies from 30.36% to 36.31% and 25% to 29% respectively, showing the efficacy of these augmentation techniques in enhancing the model's ability to generalize.

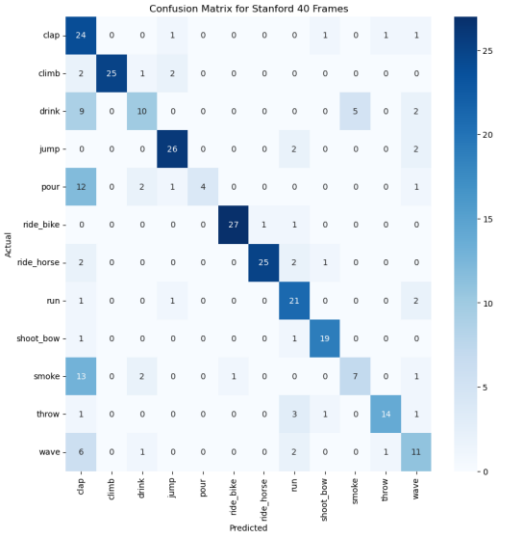
***CHOICE 3: Create a Cyclical Learning rate schedule: 5 [Compare a single model with and without the cyclical learning rate scheduler in your report. You're free to use it for your other models as well.]***

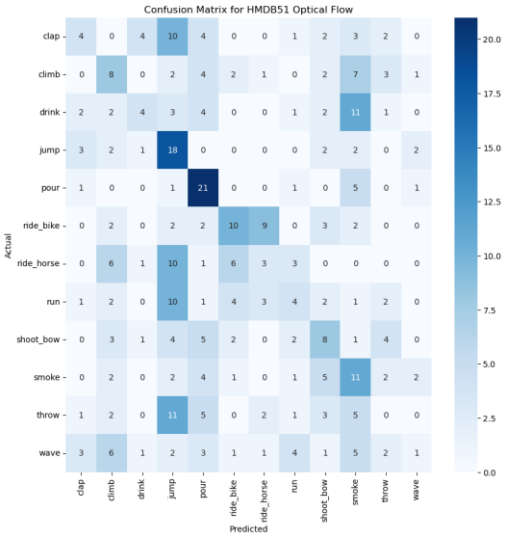
For this choice task, for the Stanford40 model, we implemented triagular2 cyclical learning rate where the learning rate drops after each cycle. The learning rate starts at 0.0001 and reaches the upper boundary that is set to 0.01. The number of iterations required to move the learning rate from the base to the max is set to the half length of the training loader, that means that we conduct 1 cycle every epoch. The implementation of a cyclical learning rate schedule for 5 epochs had a notably positive impact on the model's performance. The training accuracy improved from 96.07% to 97.62%, indicating that the model could better fit the training data without drastic overfitting, as evidenced by the substantial improvement in validation accuracy from 72.21% to 79.71%. The validation loss significantly decreased from 1.09 to 0.63, reflecting an enhanced model robustness and its ability to generalize to new, unseen data. By changing the learning rate in a cyclical way, the model could explore a broader range of solutions, avoiding poor local minima and potentially finding better convergence points.

***CHOICE 8: Present and discuss a confusion matrix for your (1) Stanford 40 Frames and (2) HMDB51 Optical flow models: 5***

Starting with the confusion matrix for Stanford 40 Frames, that scored 70% test accuracy, we see several insights. Based on the 70% accuracy, most of the classes were mostly well classified although we can see some confusions happening in several classes. For example, the class “drink” was usually confused with class smoke, which may be due to similarities in hand-to-mouth gestures in both actions. Similarly, misclassification between “clapping” and “wave” also occurred because both actions involve hand movements that are similar, and the wave classification seemed to be really bad at predicting between the two. Classes like “jimp”, “run”, “shoot_bow” had really good classifications and were well recognized by the model that showcase that these actions had a distinctive feature that the model could capture.

In this context, HMDB51 optical flow model faced more challenges, as reflected in the confusion matrix. Due to the 25% accuracy on the test set the model seemed to have a lot of difficulties recognizing movements in the videos based on several frames. Actions with motion like "pouring" and "riding a bike" were recognized with moderate success. The model's ability to capture the flow of these motions suggests that optical flow information was utilized effectively. "Riding a horse," however, was confused with several other actions, such as "riding a bike" and "running." This confusion could be attributed to similar motion patterns in these activities, indicating a need for the model to learn more distinctive motion features for "riding a horse". It is obvious that a lot of classes were confused with random classes indicating that more optical flows for these classes would help the model to recognize the movements more accurately. Also, maybe a more precise capturing of the optical flow data that could capture the movements of the classes in the specific videos more accurately could also help the classification between the classes.




***CHOICE 5: Systematically investigate how your results change when using other frames than the middle frame in your HMDB51 frame model:10***

***Last Frame:***

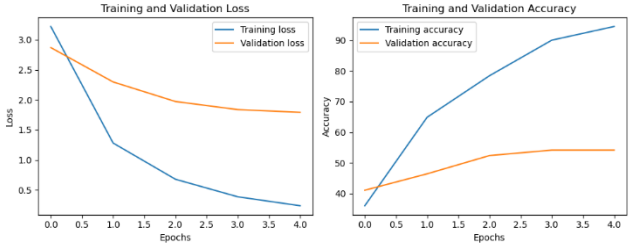



*Figure 1 Last Frame*

***First Frame:***

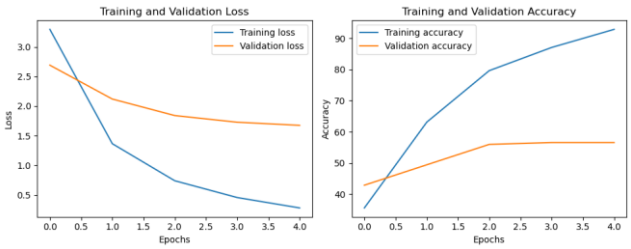



*Figure 2 First Frame*

***Quarter Frame:***

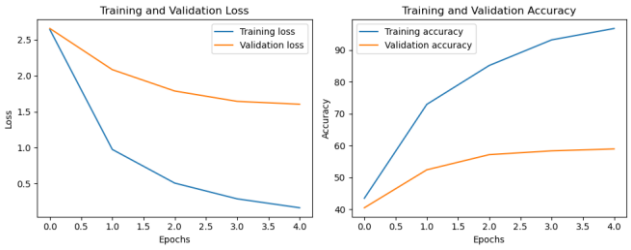



*Figure 3 Quarter Frame*

***3rd Quarter Frame:***

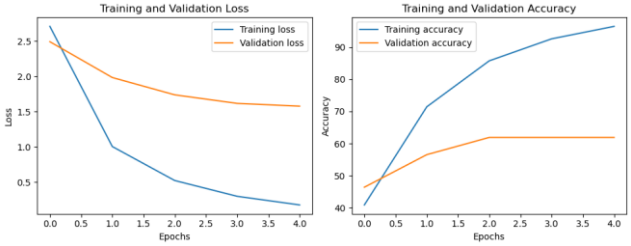



*Figure 4 3rd Quarter Frame*

***Middle Frame:***

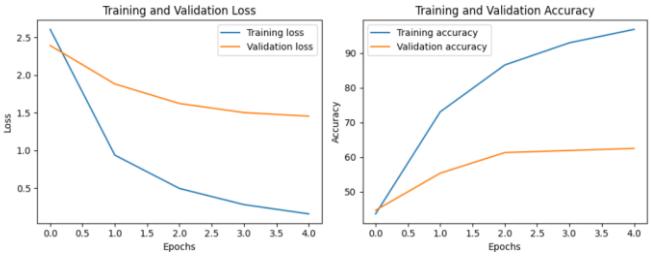



*Figure 5 Middle Frame*

*Table 2 Different Frames for Transfer Learning*

| Frame | Top-1 accuracy Train | Top-1 accuracy Test |
|---|---|---|
| First | 0.92 | 0.47 |
| 1st Quarter | 0.96 | 0.47 |
| Middle | 0.96 | 0.56 |
| 3rd Quarter | 0.96 | 0.49 |
| Last | 0.94 | 0.50 |

Based on the testing of different frames from videos in the HMDB51 dataset for training a pretrained CNN, the

findings show variations in model performance depending on the frame selection. The middle frame of the videos consistently outperformed other frames with a training loss of 0.157, a training accuracy of 96%, and a validation accuracy of 61%.

In contrast, the last frame presented the most challenges, showing a higher training loss of 0.239 and a lower validation accuracy of 54.17%, potentially due to less relevant content or motion blur towards the video's end. The first frame also underperformed with a training loss of 0.281 and a validation accuracy of about 56.55%, indicating initial frames might not fully represent the video content.

However, frames from the 1st and 3rd quarters showed improved outcomes, with the 3rd quarter frame achieving a validation accuracy close to that of the middle frame at approximately 61.90%, and the 1st quarter frame also performing reasonably well. The results show the importance of frame selection in video processing tasks, suggesting that frames capturing substantial action (typically around the middle of the video) are more effective for training models.