Writeup

# EDA

EDA was carried out to learn some important aspect of the provided data and plan different strategies/choices regarding training parameters. Different choices, reasons and their pros/cons were explained here visually.

# Model building

**Starting parameters**

Before getting into model building, a few experiments were conducted to decide on the “starting” parameters regarding batch size, image indexes and cropping resize. These parameters then were used with baseline models and might be adjusted as needed in later, more complex models.

**Model building**

Two different type of models were experimented: Conv3D and CNN + LSTM. In each type, the first models are usually extremely basic, trained on few epochs, used smaller resolution and without image augmentation. Depends on the result, next steps will be taken accordingly (increase number of epochs, increase model complexity, adjust learning rate, different image size and augmentation,…). Details in the table below. Plots of loss/accuracy per epoch can be found in the notebook.

Finally, pretrained models were tested against handmade models.

**Evaluation metrics**

Validation accuracy was the primary metrics for model performance since the class distribution was fairly balance. While training accuracy and train/val loss were not the main metrics, they were also taken into consideration since they provided insights on what was happening during training process and what steps to do next to improve performance.

**Detail experiments**

**(\*) Below are highest train accuracy and validation accuracy recorded during training, not necessary on the final epoch.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment** | **Model** | **Briefly describe the experiment** | **Result, conclusion and to do** |
| 0 | Conv3D | Preliminary test for “base” generator configuration | Decide on starting parameters:  Batch size: 16  Image indexes: every 3rd frame  Image cropping/resize: 64x64 |
| **Conv3D models** | | | |
| **Model 1** | Conv3D | Baseline Conv3D model | Train acc: 1.00 | Val acc: 0.79  Training loss/accuracy converged very fast while validation loss failed to go down.  The learning rate might be too large.  To do: reduce learning rate |
| **Model 2** | Conv3D | Reduced learning rate to 1e-4 | Train acc: 0.9367 | Val acc: 0.77  Both overfitting and somewhat underfitting  To do: add more layers and train for more epoch. |
| **Model 3** | Conv3D | Added a dense layer with 64 neurons for classification and ran for more epochs (20) | Train acc: 0.8296 | Val acc: 0.77  Seems to reduce overfitting.  Underfitting issue remains. Loss and accuracy have not yet converged.  To do: Train for more epochs. |
| **Model 3.1** | Conv3D | Train model 3 for 30 epochs | Train acc: 0.8959 | Val acc: 0.75  Training accuracy has been improved but not validation.  To do: try to reduce overfitting with random image augmentation |
| **Model 3.2** | Conv3D | Follow up model 3.1, apply different cropping/resizing technique in the generator. | Train acc: 0.8000 | Val acc: 0.86  Validation accuracy here are improving alongside with training accuracy. Looks like they have not converged yet.  To do: train for more epochs |
| **Model 3.3** | Conv3D | Follow up model 3.2, train for 50 epochs | Train acc: 0.9412 | Val acc: 0.84  Training for more epochs no longer improves performance.  To do: Add more Conv3D layers for feature extraction. |
| **Model 4** | Conv3D | Add more conv3d layers | Train acc: 0.9578 | Val acc: 0.87  Validation loss and accuracy have improved. Overfitting remains an issue.  To do: try image augmentation |
| **Model 4.1** | Conv3D | Attempt to fix overfitting by applying random augmentation | Train acc: 0.9427 | Val acc: 0.90  Both validation loss and accuracy have improved even further with random augmentation. Training accuracy doesn't seem to converge yet.  To do: train for more epochs. |
| **Model 4.2** | Conv3D | Train for 100 epochs. | Train acc: 0.9759 | Val acc: 0.91  Training for more epoch doesn't seem to improve further.  To do: apply image augmentation + use higher resolution images. |
| **Model 4.3** | Conv3D | Apply random augmentation, increase image resolution | Train acc: 0.8808 | Val acc: 0.90  Training on higher resolution images doesn't improve validation accuracy by much, but seems to eliminate overfitting issue.  Both loss and accuracy improved in a very similar manner at each epoch.  This model and generator parameter manage to achieve the lowest validation loss so far. |
| **CNN+LSTM models** | | | |
| **Model 1** | CNN+LSTM | Baseline CNN+LSTM models | Train acc: 1.0000 | Val acc: 0.88  Clear sign of overfitting. Validation loss/accuracy plateaued out.  To do: reduce learning rate |
| **Model 2** | CNN+LSTM | Reduce learning rate to 1e-3 | Train acc: 0.9985 | Val acc: 0.89  Overfitting still remains.  To do: apply image augmentation |
| **Model 3** | CNN+LSTM | Apply augmentation to help with overfitting. Also increase number of epochs. | Train acc: 0.9864 | Val acc: 0.89  Applying random augmentation does not help much. Suspect that the model is too simple, hence unable to learn enough features to generalize to the validation set.  To do: increase number of filters in CNN layers. |
| **Model 4** | CNN+LSTM | Increase the number of filters in the CNN layers and neurons in classification layers. | Train acc: 1.0000 | Val acc: 0.95  Increase model complexity helped increasing validation accuracy (95%), highest so far. |
| **Model 4.1** | CNN+LSTM | Enable image augmentation for model 4. | Train acc: 0.9985 | Val acc: 0.91  Enable image augmentation does not seem to help with improving loss/accuracy. |
| **Transfer learning models** | | | |
| **Model 1** | Mobilenet + LSTM | Train classification layers + finetune CNN layers | Train acc: 1.0000 | Val acc: 0.84  Without retraining the feature extraction layers, performance were not as good as expected. |
| **Model 1.1** | Mobilenet+ LSTM | Retrain all CNN layers | Train acc: 1.0000 | Val acc: 1.00  Stellar performance, outperforms any handmade model.  To do: attempt to reduce model size without compromising performance. |
| **Model 2.** | Mobilenet+ LSTM | Reduce the number of neurons in the ConvLSTM2D layer and the classification layer to reduce the model size | Train acc: 1.0000 | Val acc: 1.00  Perfect performance even with reduced complexity  Passed all test cases which were unseen during training/validation. (See section 7.3 Test data in the notebook) |

# Final model recommendation

**Primary recommendation**

Considering performance and model size, **Retrained Mobilenet + LSTM** is recommended. Reasons:

- Highest validation accuracy (100%).

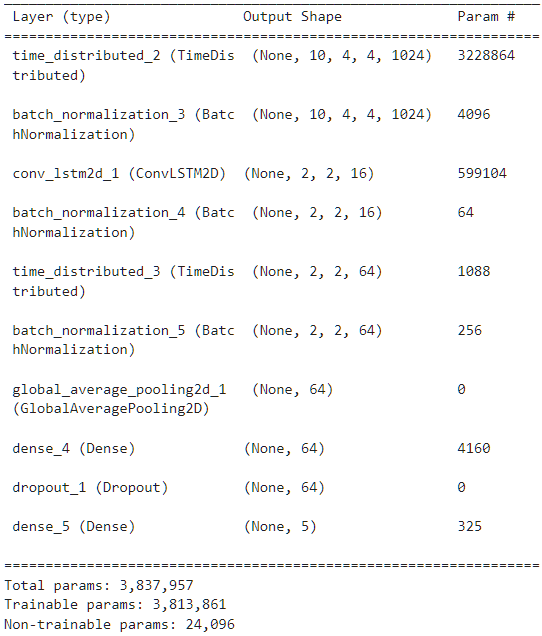
- Relatively low parameters for such performance.

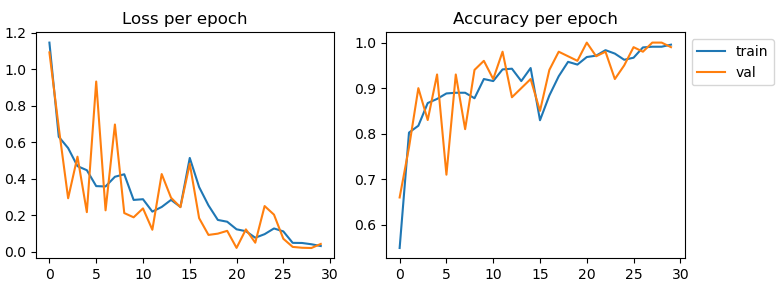
- Mobilenet was designed to be low-latency, low-power, parameterized to work under the resource constraints.

- Passed all test cases which were unseen during training/validation. (See section 7.3 Test data in the notebook)

- From a user experience perspective, TV equipped with gesture recognition tend to be higher end. At such price range, this feature has to be as reliable as possible, or else it will end up just a marketing gimmick that no one actually uses.

Model architecture and performance:





**Alternative recommendation**

Alternative recommendation: **Handmade CNN+LSTM model**. Reason:

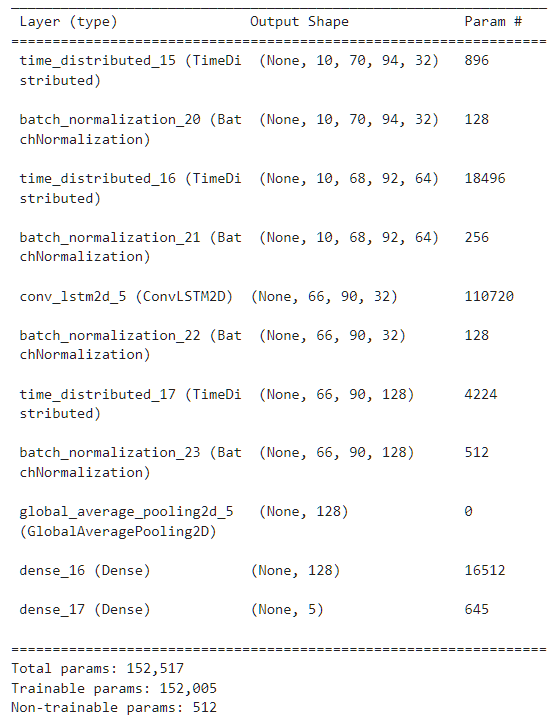
- Second highest validation accuracy (95%)

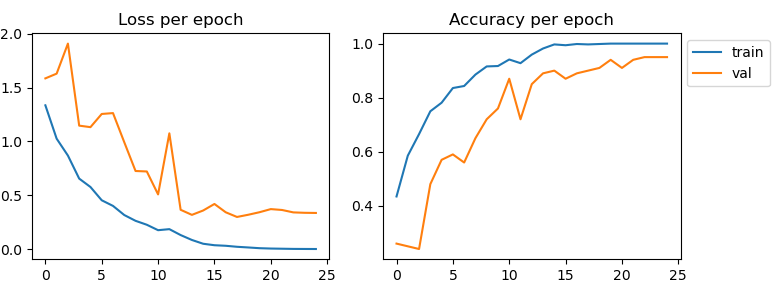
- Extremely small number of parameters (only about 100,000)

- Performance and hardware requirement fitted for mid-low range TV.

- May not work as well on unseen test data.

Model architecture and performance:





End of summary. Thank you.