* Experiment Modeling

In this section a description will be presented about the used dataset, the experiments that have been run, the validation of these tests and the comparison of the results with other literature.

1.1. ASL Dataset

The main focus of this research is finding a deep learning architecture, which can be run on a middle-end PC for sign language recognition. For this sake the data used is the popular ASL dataset. The ASL dataset stands as a foundational resource within the field of sign language recognition, serving as a pivotal repository for understanding and developing solutions in this domain, like in the transfer learning research [4].  
It is a collection of images created from video footage, containing 29 classes, ranging from A to Z and 3 additional ones: nothing, space and delete.

The classes are populated with 87000 images, all of them with size of 200 by 200.

The test set is made out of a portion of this, meaning 176 images from the whole.

For the training data augmentation is used on both the training and test sets. On the training the following techniques are used: horizontal flip, rotation up to 50 degrees, zooming up to 20%, width and height shift up to 20% and rescaling image values to between 0 and 1. On the test only the rescaling is applied.

1.2. Experiment structure

Since the main goal of the research is to find an architecture with minimal system requirements, the model used for transfer learning is the famous mobilent. This deep learning system was designed to be used on light hardware, like mobile phones, hence the choice for this paper.  
The experiments will make use of transfer learning from the pretrained weights, eliminating more and more of the last few layers with the aim to make the model lighter.

In this literature I ran 8 experiments sequentially eliminating the last couple convolutional layers from the model, and replacing them with a reshaping and dense layers to fit the ASL dataset.

Starting from the first convolutional layer to the eight the number of eliminated layers are from 4 to 34, and the exact informations can be found in the experiments.in file, where the first line indicates the last number of layers to be deleted from the model, the second line the shape of the data in that current layer, this is for the reshape layer to handle the tensor dimensions and the last layer is the last number of layers which are trained in the experiments with the new data. The weights of the layers which remain from the original model are then locked so they are not changed with the new dataset, having the pre learned values.

The training is done with an Adam classifier (training rate 1e^-3) and categorical cross entropy for a loss function.

The hardware used is an Nvidia GTX 1660 with 6GB VRAM running on windows 11 with 16 GB of RAM.

1.3. Validating results

Running the experiments means training the newly created deep learning architecture on the ASL datasets. From these runs I obtain an accuracy and loss score, while also running 6 single predictions to obtain an average running time in miliseconds. These metrics are than mainly compared to the YOLOv3 model [2], usage of representative frames [3] and simple transfer learning with the MobileNet model [3]. Since the main goal is achieving the quickest runtime, the minimum requirement is a less than 41 ms runtime, for the model to be able to be classified as runtime.

For the experiments the Tensorflow framework is used, and because of that the first prediction is not included in the average runtime, since Tensorflow uses this to setup the model for future predictions, hence skewing the average.

Also alongside the experiments, other programs were run, like in a real life scenario, so the comparisons will only be made with the average minimums of these runs.

* Case Study

For the first set of experiments I ran the MobilNet model with a small portion of the ASL dataset. In this architecture only the first convolutional layer is swapped and replaced with a Reshape and a Dense layer.

To achieve the least system requirements, the MobileNet is chosen since it is one of the most widely used light weight deep learning models, hence for image classification, the pretraining of it is more than sufficient.

After this initial experiment the loss (16.553715159256899), the accuracy (0.1523238646833523) and the average time (1.1954845123651235) is retrieved.

The first benchmark for this study is to check if the model precincts in real time. To validate this, I use the commonly accepted frame rate of 24 for movies and videos, meaning the code has to run under 41 milliseconds.

The average time was far off this mark, but the average minimums were around 300 ms, which is a good sign for prediction on a mid-end computer.

Comparing the time needed with other results, it comes shy off the mark, with the YOLOv3 [2] model achieving 0.9768 accuracy, while other methods, like finger segmentation [14], the usage of representative frames [3], other MobileNet models [4] and SVMs [13] are all achieving around 90-95 percent accuracy.

Most studies don’t dwell into real time recognition of Sign Language symbols in this topic hence here three studies are used for comparison. The Representative Frame approach [3] realizes a run time of 110 ms to 180 ms, the YOLOv3 model [2], while it isn’t discussed what they mean exactly, achieves real-time performance.

The approach of Neethu, Suguna and Sathus [7] completes in around 0.333 s.

From the mentioned studies, achieving a run time of 100-200 ms for these experiments would result in a sufficient approach, if achieving the 41ms might not be realizable.

* Related Work

Github link for detailed description about the experiments, and their exact parameters:

<https://github.com/Arnold-21/ResearchProject>

3.1. Experiment results

The experiments were run by swapping the convolutional layers of the MobileNet model from the first to the eight. The replacement was a Reshaping and a Dense layer, while also retraining the last 25-30 layers of the existing model.

Out of these 8 experiments the most two important metrics were testing accuracy and run time. Looking at the minimums of these values there are two cantidades worth comparing.

The first run, where the first 5 convolutional layers, meaning up to the 17th layer, was swapped and the last 30 layers were trained, achieved an accuracy of 96.55% with a run time of 949 ms and an average minimum between 40 and 50 ms.

The second run with the first 6 convolutional layers swapped, up to the 22th layer, the last 30 layers trained, realized an accuracy of 96.55% with a slightly worse loss, but a better run time of 753 ms and an average minimum between 35 and 45 ms.

3.2. Comparison of the results

These experiments can be best compared with the three methodologies already mentioned in the Case Study section.

The YOLOv3 model is another efficient deep learning architecture, making it more than useful in achieving real time performance on sign language detection problems. The study of Mujahid, Awan and co. [2]. Their approach is about teaching a model from scratch on the Mindst dataset, achieving good results. Their model can recognize and bound the hand gestures with an accuracy of 98% in real-time.

The difference in this method compared to theirs lies mainly in the data and model, both of them being simpler in complexity and grandness. While the comparison is not accurate enough to yield any concrete result, since the YOLOv3 is much simpler than the MobileNet, achieving an average time of 753 ms with the average minimums also hitting our real-time threshold, shows that this method can be considered an efficient one on a more difficult dataset.

To better show the results of this approach the best comparison can be made with the works of Wanga, Hua and Jina [4]. Their approach consists of using the most of the pretrained MobilNet weight values, and utilizing transfer learning at the end. The architecture was tried out on three different datasets. Out of these their approach achieved an accuracy of 98% on a similar dataset to ASL, showing the potential of this approach. Given that they don’t go into much detail about timewise performance, no comparison can be made in that part.

The third paper is the representative frame approach [3]. In this work, although MobilenNet wasn’t used, the architecture was a linear CNN, roughly half the size of MobileNet in terms of parameters. They managed to get an accuracy of 91% on a similar approach. The method consists of two parts: the extraction of representative frames from video and the recognition of these images. In their approach the real-time performance was 70 ms and 40 ms respectively, realizing an overall of 110 ms.

In comparison we can conclude that the proposed approach in this paper has promising results. In comparison to a more defined and specific transfer learning approach [4] the accuracy difference on a similar dataset is realistically around 2-3%. And with respect to time needed, we can see the difference in size between these two MobileNet models, which are already brought down in dimensions, and the likes of YOLOv3 [2] or custom models [4]. The time difference is vast in around 550-600 ms, although running in isolation the models produce promising results, which can run as low 35-50 ms or more realistically around 150-200 ms.Not being perfect, the two models found achieve a reasonable performance, in near real-time.

3.3. Areas for improvement

While the test data provided by the ASL dataset is sufficient, the pictures are very similar to the training part. While data augmentation is used, having more variety would give a more real picture to recognition accuracy.

With according the time readings, running the predictions in a more controlled environment, where not many other programs are fighting for computing time, would give a clearer picture about the true values of this metric.

* References

1. Oyebade K. Oyedotun, Adnan Khashman (2016). Deep learning in vision-based static hand gesture recognition. Neural Computing and Applications, 28(12), 3941–3951. p:3-9, doi:10.1007/s00521-016-2294-8

2. Abdullah Mujahid, Mazhar Javed Awan, Awais Yasin, Mazin Abed Mohammed, Robertas Damaševicius, Rytis Maskeliunas, Karrar Hameed Abdulkareem (2021). Real-Time Hand Gesture Recognition Based on Deep Learning YOLOv3 Model. Applied Sciences, 11(9), 4164. p:5-12, doi:10.3390/app11094164

3. Vijay John, Ali Boyali, Seiichi Mita, Masayuki Imanishi, Norio Sanma (2016). Deep Learning-Based Fast Hand Gesture Recognition Using Representative Frames. 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA). p:2-8, doi:10.1109/dicta.2016.7797030

4. Fei Wanga, Ronglin Hua, Ying Jina (2021). Research on gesture image recognition method based on transfer learning. Procedia Computer Science, 187, 140–145. p:3-6, doi:10.1016/j.procs.2021.04.044

5. Adithya V. & Rajesh R. (2020). A Deep Convolutional Neural Network Approach for Static Hand Gesture Recognition. Procedia Computer Science, 171, 2353–2361. p:3-8, doi:10.1016/j.procs.2020.04.255

6. Mehmet Akif OZDEMIR, Deniz Hande KISA, Onan GUREN, Aytug ONAN, Aydin AKAN (2020). EMG based Hand Gesture Recognition using Deep Learning. 2020 Medical Technologies Congress (TIPTEKNO). p:2-4, doi:10.1109/tiptekno50054.2020.9299264

7. P. S. Neethu, R. Suguna, Divya Sathish (2020). An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. Soft Computing. p:3-8, doi:10.1007/s00500-020-04860-5

8. Manasi Agrawal, Rutuja Ainapure, Shrushti Agrawal, Simran Bhosale, Sharmishta Desai (2020). Models for Hand Gesture Recognition using Deep Learning. 2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA). p:2-5, doi:10.1109/iccca49541.2020.9250846

9. Sakshi Sharma, Sukhwinder Singh (2021). Vision-based hand gesture recognition using deep learning for the interpretation of sign language. Expert Systems with Applications, 182, 115657. p:4-11, doi:10.1016/j.eswa.2021.115657

10. Oyebade K. Oyedotun, Adnan Khashman (2016). Deep learning in vision-based static hand gesture recognition. Neural Computing and Applications, 28(12), 3941–3951. p:2-9, doi:10.1007/s00521-016-2294-8

11. Priyanka Parvathy, Kamalraj Subramaniam, G. K. D. Prasanna Venkatesan, P. Karthikaikumar, Justin Varghese, T. Jayasankar (2020). Development of hand gesture recognition system using machine learning. Journal of Ambient Intelligence and Humanized Computing, 12(6), 6793–6800. p:3-7, doi:10.1007/s12652-020-02314-2

12. Tayyip Ozcan, Alper Basturk (2019). Transfer learning-based convolutional neural networks with heuristic optimization for hand gesture recognition. Neural Computing and Applications. p:3-14, doi:10.1007/s00521-019-04427-y

13. Bi-Xiao Wu, Chen-Guang Yang, Jun-Pei Zhong (2021). Research on Transfer Learning of Vision-based Gesture Recognition. International Journal of Automation and Computing, 18(3), 422–431. p:3-8, doi:10.1007/s11633-020-1273-9

14. Anjali R. Patil, S. Subbaaraman (2022). Performance analysis of static hand gesture recognition approaches using artificial neural network, support vector machine and two stream based transfer learning approach. International Journal of Information Technology, 14, 3781-3792. p:2-9, doi:10.1007/s41870-021-00831-7

15. Anuja P. Parameshwaran1, Heta P. Desai1, Rajshekhar Sunderraman, Michael Weeks (2019). Transfer Learning for Classifying Single Hand Gestures on Comprehensive Bharatanatyam Mudra Dataset. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). p:1-2, doi:10.1109/cvprw.2019.00074