Machine Perception Project: Dynamic Gesture Recognition

Mauro Luzzatto
maurol@student.ethz.ch

Dario Kneubuehler
dariok@student.ethz.ch

ABSTRACT

Methods relying on neural networks have demonstrated astounding results for solving various problems, ranging from image classification to speech recognition. However, extracting information from the temporal structure in video data remains a challenging task. Based on previous work done in the field by Pigou et al. [5] and Molchanov et al. [4] we aim at implementing a neural network, that is able of reaching performance comparable with the state of the art. Furthermore, we suggest some changes to the data augmentation used in previous work, that has the potential to further improve the performance.

1 INTRODUCTION

The goal of this machine perception project is to train and evaluate a deep learning model that is able to recognize individual gestures from a video clip. For every video snippet, one out of 20 Italian sign gesture should be predicted by the model. For this particular task, a customized version of the ChaLearn dataset is provided [2]. The ChaLearn dataset contains RGB, depth, segmentation mask and skeletal information for every video. Totally, 9661 video samples are available, out of which 5722 are used for training, 2174 for testing and 1765 for validation. The data is stored using the TFRecord format, where every clip contains 50-150 frames with a size of 80X80.

One of the major challenges of this project is to avoid using a too complex model which will overfit the training data and poorly generalize. Since there is a small number of training samples provided, the number of deep learning model parameters are likely to be much larger. Therefore, it is important to consider regularization techniques to reduce the problem of overfitting. Additionally, the convergence of the model is very sensitive to the chosen set hyperparameters. Small changes in batch-size, learning, etc. cause the model to not converge.

2 RELATED WORK

There is active research within the field of video content classification. Most closely related and relevant to this project is the work done by Pigou et al. [5]. They showed in their research that simple temporal feature pooling - mean- or max-pooling over several video frames - is not able to consider temporal aspects of a video. Therefore, a new method was presented by them, that uses deep architectures, bidirectional recurrence, and temporal convolutions to improve the temporal information extraction of a video significantly. The method was applied and tested on the same gesture recognition task using the ChaLearn dataset [5]. Similar work in the field of classification of dynamic hand gestures where a recurrent neural network (RNN) and a three-dimensional convolutional network (CNN) were combined, is presented by Molchanov et al. They used the previously mentioned approach to simultaneously detect and classify dynamic hand gestures from multi-modal data [4]. Finally, Du Tran et al. pointed out the accuracy advantage of three-dimensional over two-dimensional CNNs within the framework of residual learning. Additionally, it was shown in their work that factorizing the 3D convolutional filters into separate spatial and temporal components yields significant gains in accuracy [6]. Considering the related work, two model design choices seem to be the most promising to solve the task of dynamic gesture recognition. Fist, using CNNs with RNNs in combination to extract temporal features. Second, three-dimensional CNNs result in better features than two-dimensional CNNs.

3 METHOD

For our architecture, we decided to build a two-dimensional CNN in combination with a recurrent network and further expand this approach with a bidirectional RNN. In case this approach is not sufficient enough, we are going to further expand the CNN to a three-dimensional architecture. Additionally, we focus on various data augmentation techniques to tackle the problem of overfitting. Our main idea is to apply augmentation to a stronger extent on the entire video sequence and to a weaker extent on the single video frames. On the video level, we use data augmentation techniques as proposed by [5]. They proposed data augmentation techniques like rotation, cropping, and translation of the videos. We extended this approach to include a perspective transformation, that allows the augmentation to have a total of 8 degrees of freedom and extend the variety on the input data. A comparison between using only translation, rotation, and cropping, and a perspective transformation can be seen in figure 1. On a frame level, we applied the same augmentation techniques, but with smaller parameters. This essentially adds small amounts of noise to the input channels. Adding noise to the inputs is also recommended as a dataset augmentation strategy in [3] and [1].

Regarding the model architecture, we implemented two different CNN architectures. In CNN architecture one, after every convolutional layer, one max-pooling layer is applied. Filter numbers of in the following sequence C(16)–P –C(32) –C –P –C(64) –P –C(128) –P –C(256) –P –D(512) are chosen with a kernel size of 3x3. For the CNN architecture two, we implemented the architecture proposed by [5]. There, two convolutional layers are followed by one pooling layer. This resulted in a C(16) –C(16) –P –C(32) –C(32) –P –C(64) –C(64) –P –C(128) –C(128) –P –D(2048) –D(2048) sequence of layers. As previously mentioned, the outputs of the CNNs are either feed to a unidirectional or bidirectional layer with an LSTM size of 512. For the bidirectional case, the forward and backward outputs are added together. Regarding the activation of the layers, we consider a normal relu function and additionally a leaky relu function with an alpha value of 0.3. To avoid exploding gradients the range of the gradients is limited to a maximum value of 10, using gradient

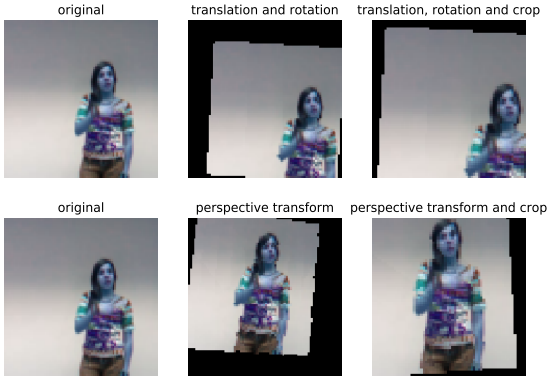

Figure 1: Data augmentation performed using translation, rotation and cropping (upper three pictures) compared to a perspective transformation and cropping (lower three images).

clipping. The optimization is done with an exponential decaying learning rate starting at 5e-4 and a decay rate of 0.97 every 500 steps. As optimizer ADAM was selected, which seems to be the most promising for this gesture recognition task [5]. Experimenting with the loss functions provided in the skeleton code shows, that the average loss of all logits is resulting in the best performance. This is then fed to a sequence to sequence loss. For the training, a batch size of 16 is chosen and depended on the chosen architecture epochs in the range of 25 to 40 are applied. Finally, additional regularization techniques like dropout in the range of 0.3 to 0.5 are considered and early stopping.

4 EXPERIMENTS AND RESULTS

Various experiments are conducted to determine the influence of the network parameters on the performance of the model. Table 1 shows the progress over multiple iterations of architectural changes. The architectures are labeled from 1 to 6, where 1 is the lowest score of the skeleton code and 6 corresponds to the final model with the highest score. First, by using the average loss over the logits of the RNN an improvement of 5.5% is achieved. However, overfitting is still a problem with the second architecture. The data augmentation that is implemented in architecture three randomly translates and rotates each video in the range of +/- 8 pixels, +/- 4 degrees respectively. Additionally, a random translation of +/- 1 pixel as well as a random rotation of +/- 1 degree is added to each frame in the video, acting as noise on the input dataset. These changes are able to increase the accuracy by an additional 7%. In the fourth iteration of architectural changes the unidirectional RNN is replaced by a bidirectional RNN and the relu activation function of the CNN is replaced by a leaky relu with a alpha parameter of 0.3. These changes boost the accuracy an additional 7.8%. In architecture five, the data augmentation is extended to include a random perspective projection of +/- 0.002 per video, and +/- 0.0001 per frame. This resulted in an accuracy of 83.2%. In the final architecture a number of parameters are fine-tuned. The dropout rate is increased from 0.3 to 0.4 in the CNN and to 0.35 in the RNN. The number of epochs is also increased to 40, resulting of a final validation accuracy of 87.1%.
All these results are achieved using the CNN architecture one. The second architecture is examined as well, however, it has a problem with convergence during training and did not yield comparable results.

Table 1: Validation Results

	Description	Epoch	Score
1	Skeleton code	25	61.7 %
2	Using average loss	25	67.2 %
3	Data Augmentation 1	25	74.2 %
4	Bidirectional LSTM, leaky relu activation for CNN	25	82.0 %
5	Data Augmentation 2	25	83.2 %
6	Increase dropout, more training	40	87.1 %

Table 2: Test score for different network architecture iterations. The accuracy score is calculated using 40% of the final test dataset provided on Kaggle. The epoch column stands for the number of epochs the model was trained for.

5 DISCUSSIONS

The measures that had the biggest impact on the performance were:

(1) The use of data augmentation techniques
(2) Using a bidirectional RNN and a leaky relu activation for the CNN
(3) Using dropout in both networks
(4) Using the average loss

Data augmentation proved to be very effective against overfitting. The additional incorporation of small amounts of noise helped with further generalizing the model. Adding noise to the inputs is also recommended as a dataset augmentation strategy in [3] and [1]. The use of a bidirectional RNN also had a large impact on the improvement of the performance. As suggested by [5] bidirectional RNNs outperform conventional RNNs, as they are able to process information in both temporal directions. Using dropout as a regularization technique also proved to be effective against overfitting. Adding two convolution layers in series as proposed in [5], resulted in the network not converging during training. It seemed like little architectural or parameter changes could have had a large impact on whether the network could have been successfully trained or not.

6 CONCLUSION AND FUTURE WORK

In this project, the authors implemented a deep learning model that is able to detect 20 different Italian sign gesture, with an accuracy of 87%. The final network uses 3 channel image inputs and processes them with a 5 layer convolutional neural network (CNN) followed

by a bidirectional recurrent neural network (RNN). Since the number of training samples is small compared to the complexity of the network, avoiding overfitting of the model was a major challenge. Data augmentation through the transformation of the input images, as well as adding a small amount of noise to the input data, helped to generate greater variety in the training set. This was additionally combined with early stopping and dropout in the CNN and the RNN to further reduce the impact of overfitting and improve the performance.

The current implementation could be further improved with adding more regularization measures, and the examination of additionally network architectures of the layers in the CNN and the bidirectional RNN. Especially, the CNN architecture could be further improved using three-dimensional CNNs instead of two-dimensional ones. Furthermore, determining the sensitivity of the various parameters on the overall result via a large grid search is expected to give more insight into the problem and help improve the accuracy. Finally, incorporating the additional channels of the dataset (depth, segmentation mask, and skeletal information) into the model is expected to boost the accuracy even further.

REFERENCES

[1] Chris Bishop, Christopher M Bishop, et al. 1995. *Neural networks for pattern recognition*. Oxford university press.

[2] Sergio Escalera, Xavier Baró, Jordi Gonzalez, Miguel A Bautista, Meysam Madadi, Miguel Reyes, Víctor Ponce-López, Hugo J Escalante, Jamie Shotton, and Isabelle Guyon. 2014. Chalearn looking at people challenge 2014: Dataset and results. In *Workshop at the European Conference on Computer Vision*. Springer, 459–473.

[3] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. 234 – 236 pages.

[4] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. 2016. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4207–4215.

[5] Lionel Pigou, Aäron Van Den Oord, Sander Dieleman, Mieke Van Herreweghe, and Joni Dambre. 2015. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision* (2015), 1–10.

[6] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. 2017. A Closer Look at Spatiotemporal Convolutions for Action Recognition. *arXiv preprint arXiv:1711.11248* (2017).