Machine Learning Engineer Nanodegree

Capstone Project

Juan Andrés Ramírez September 26th, 2017

I. Definition

Project Overview

Hand gesture recognition is an important research issue because of its extensive applications in virtual reality, sign language recognition, and computer games [1]. Sensors used for hand gesture recognition include wearable sensors such a data gloves and external sensors such as video cameras and depth cameras [2]. Data gloves usually require extensive calibration and restrict natural hand movement [2]. Video-based based approaches addresses this issues but adds other problems, like the hand segmentation from background and occlusion [2]. There are several recent works that uses depth cameras as sensors, but this hardware doesn't have the availability that video cameras have today in people's homes.

Vision-based hand gesture recognition techniques can be divided into two groups: appearance-based approaches and 3D hand model-based approaches. Appearance-based approaches use image features of the hand and compare these parameters with the extracted image features from the input. 3D hand model-based approaches rely on a 3D kinematic hand model and try to estimate the hand parameters by comparison between the input images and possible 2D synthetic images, generated by the 3D hand model [3].

In this project I created an appearance-based classifier of the three rock-paper-scissors game hand gestures. The algorithm was trained with the SenseZ3D static hand gesture dataset [4, 5] combined with new images specially created for this project.

Problem Statement

The main objective of this work is to create an image classifier capable of detecting the three different hand gestures from the rock-paper-scissors game. This classifier should allow a human user to play this game with a computer using just a screen and a webcam as an interface.

The proposed solution obtains a classifier by applying *Transfer Learning* to ResNet50 [7] convolutional neural network. A new rock-paper-scissors dataset (based on SenseZ3D [3, 4] combined with new images) is created and used to adapt the weights of a fully connected layer that takes its inputs from ResNet50 pre-classification outputs. The resulting algorithm is evaluated as a classification task where the input are the images of people showing hand gestures from rock-paper-scissors game. The accuracy measure it was used to evaluate performance over the three different classes of the rock-paper-scissors game.

Metrics

The proposed evaluation metric is the classification accuracy over the test dataset. This measure is appropriate to the task because the classes have equal priority and the accuracy score shows overall classification performance.

II. Analysis

Data Exploration

The proposed classifier receives a webcam color image as input. The algorithm should work well on different light, postures and background conditions, so the training and testing datasets should reflect this noisy environments. For this purpose there's the need of using a lot of samples from different sources. The selected databases are: * A subset of the SenseZ3D static hand gesture dataset [4, 5], which contains 30 images of different hand gestures from each of 4 subjects in webcam similar situations. The proposed subset contains just the gestures of Rock-Paper-Scissors (G1, G2 and G5) * A specially made dataset with the webcam images of 7 subjects. This dataset contains 30 images of each gesture for each subject

The following table shows some samples from the considered datasets

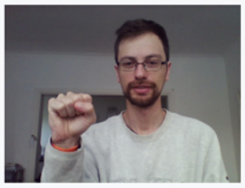
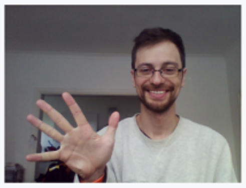
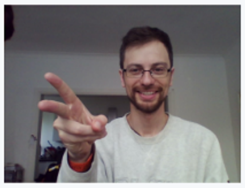
Dataset	Rock	Paper	Scissors
SenseZ3D			
Specially made			

Figure 1: dataset samples

Thus, considering both datasets, there were:

- 30 images per class, per subject
- 990 total images
- 330 images per class
- 11 different subjects

SenseZ3D Dataset

As mentioned before, the intended rock-paper-scissors classifier for the web should work well on a broad range of situations. However, the SenseZ3D dataset doesn't show too much variation in background, pose and illumination conditions of the images. The next figure shows this problem in the 30 images of subject 1 for gesture 1 (paper):

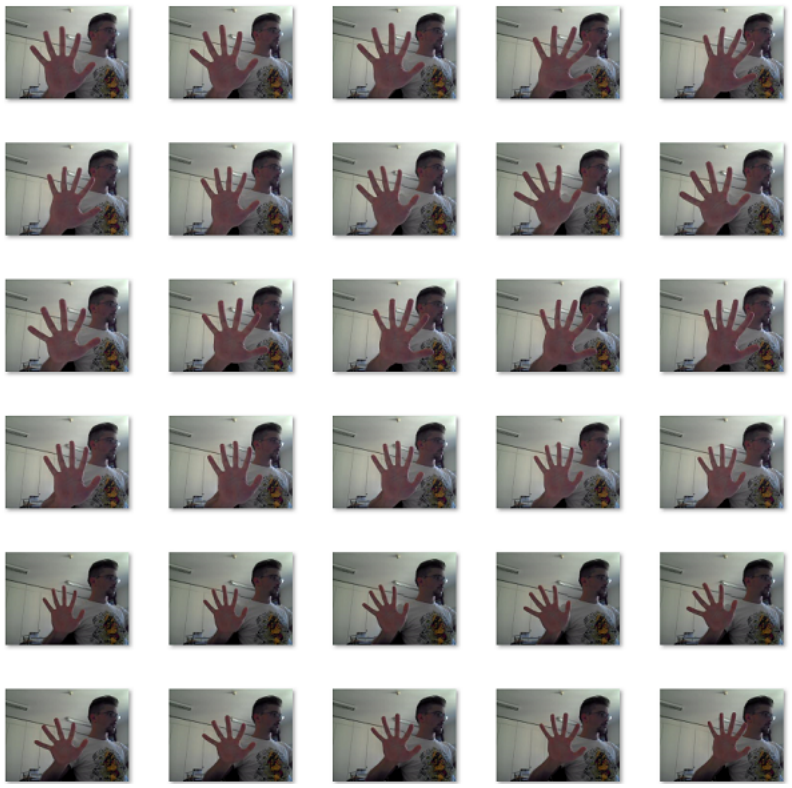


Figure 2: sensez3d paper images

Specially Made Dataset

This dataset was created to provide the algorithm the possibility to learn from a wide range of conditions. There were used three different locations when capturing each gesture subset. The next figure shows the variations in pose, background and illumination for the paper images of one subject

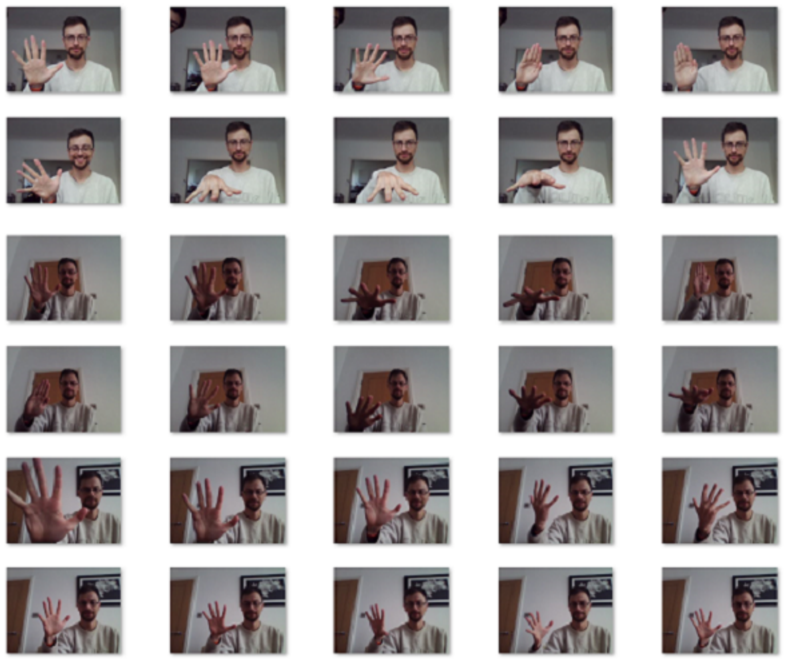


Figure 3: RPS dataset paper images

From the images above we may observe the following:

- Given a static hand gesture (paper), different postures/orientations generates very different images
- There are several degrees of freedom for a hand to express a static gesture

This facts are reflected on this dataset

Exploratory Visualization

The main difficulties of this problem were already mentioned. These include:

- Pose variation
- Illumination variation
- Background variation

This variations are shown in the figure above (refer to *Specially Made Dataset*).

As the proposed solution uses ResNet50 features extracted from images, it would be interesting to look how this features will enable the classifier to discriminate between classes. This could be done by projecting sample's features into PCA axes and plotting the labeled points. This kind of analysis may help us to decrease the dimensionality of the problem as we could reduce the number of features. . A first approach to this analysis is to evaluate how much variance is explained by the most important components of the PCA transformation from training data. The next figure shows just this for the most important components:

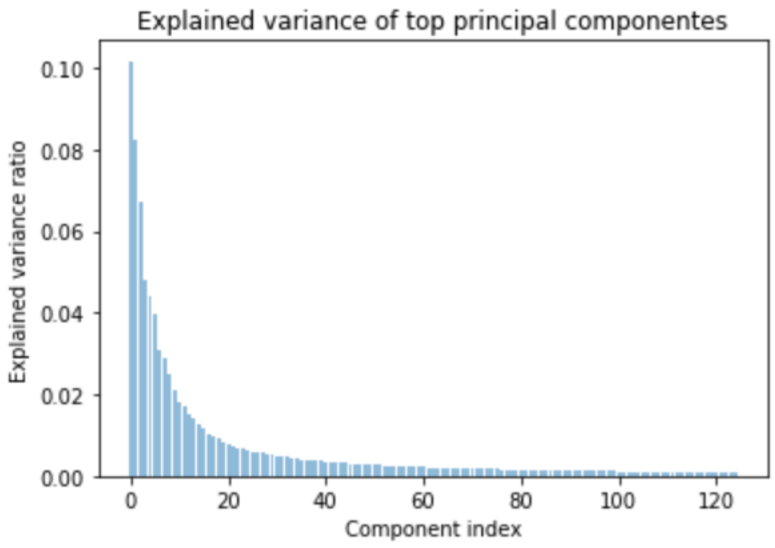


Figure 4: Normalized principal components explained variance

The first 100 components of the 2048 total number of features sums 83% of the total variance. This shows that there is a great chance we can reduce dimensionality in a significant way. However, this doesn't say anything about how this new subset of features may help us to discriminate between classes. The reduction of features will need testing and further study and are postponed for future work.

Algorithms and Techniques

Classification Model

Deep convolutional neural networks have led to a series of breakthroughs for image classification [7]. Several results have been shown that this networks may outperform humans in object recognition tasks over images. This type of neural networks replace large layers of fully connected units by filtering layers that apply convolutions to their inputs. The use of several layers provides the network with great representation capacity.

The difficulty of designing a good model from scratch (computational complexity and the requirement of a huge amount of samples) justifies the utilization of *Transfer Learning* methods.

ResNet50, a convolutional deep neural network based on residual learning, is one of the better models available (considering performance on ImageNet) for the task of object recognition in images [7]. For this reason it was selected to be part of the proposed model. This model uses *residual layers*. This kind of layer differs from other architectures because provides input information to the output of the layer. The next figure shows the basic building block of ResNet50:

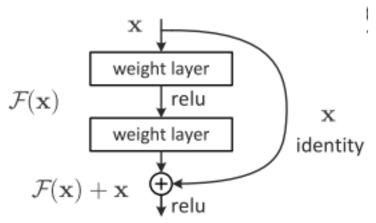


Figure 5: The basic building block of ResNet50

This building block is serially repeated many times. ResNet50 is much deeper than VGG architectures (VGG16 and VGG19) and the model size is actually substantially smaller due to the usage of global average pooling rather than fully-connected layers.

The intended classifier is a fully connected neural network layer which receives ResNet50 features from images. The whole algorithm receives a color image of size 224x224 which must be preprocessed with the methods proposed in [7]. As an output the classifier returns a vector where each of its three components is the predicted probability for each class. The features from ResNet50 are taken after reducing dimensionality with an average pooling layer, as suggested by the authors [7].

The average pooling layer takes the average result from certain dimension of a vector. Dropout layers were also tested (droput layers randomly inhibits signals on training and are commonly used to prevent overfitting) but wasn't considered in the final model.

The final total number of features is 2048. As the authors use a final fully-connected neural network layer, the same it is used in this project. This layer ends on a *softmax* activation step to get probability outputs. The classification layer has 6147 weights to provide an output of three components.

Training

Several algorithms and configurations were tested. The one that threw better results was RMSProp with Keras default parameters

Data Augmentation

Data Augmentation it is a common technique to better exploit the available dataset and also used by the authors of ResNet50. For the purposes of this project, a different configuration was applied:

- Rotation up to 40 degrees (to consider just standard hand positions)
- Shifting up to 10% (not too much shifting to prevent the hands to be cropped)
- Zoom range of 0.1 (not too much zoom variation to prevent the hands to be cropped)
- Horizontal flipping to prevent right-handed bias
- Filling by nearest method, to hide rotation information to the algorithm

The following figure shows the samples generated from a single image:

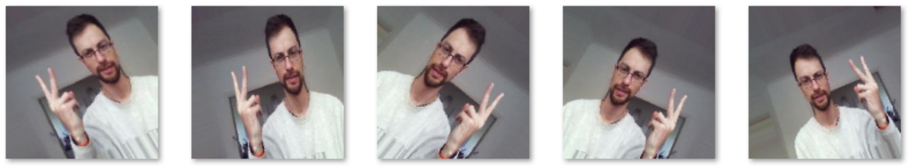


Figure 6: Augmentation examples

Benchmark

A general hand gesture classifier model may be used as a benchmark for this project. This type of classifiers usually work with more than three classes, but they are the closest benchmarking models found in literature.

In the work done in [6], researchers used a very small training set (images from 3 out of 19 individuals) and a very large testing dataset. They claimed to have achieved 85.8% accuracy on images with complex background when classifying them into 12 different hand postures. This work seems to be a little bit old and it is not possible to get its source code. Also the dataset used in their study contains images of smaller size, which makes them not appropriate to use with the proposed model. However, as most of the recent work is based on range cameras, this research appears to be a good reference and starting point for this project.

III. Methodology

Data Preprocessing

Operation

When in operation, to classify images, the proposed method need all images to be pre-processed by the Operation Pre-Process scheme, defined by the steps below:

1. Rescale to 224x224
2. Execute the preprocessing method defined in [7] and available in Keras
3. Get features from ResNet50: Execute ResNet50 algorithm and extract the outputs of the layer that precedes the classification layer

Training/Testing

For training and testing, the following pre-processing steps were applied to Training, Evaluation and Testing datasets:

1. Make augmentation. The datasets were augmented off-line by a factor of k (different values of k were used). This means that for each image, k new images were generated applying random transformations. The transformations parameters were described in *Data Augmentation*. The resulting images were scaled to 224x224
2. Apply the Operation Pre-Process to all images

Implementation

The final implementation was made on Keras, using TensorFlow backend. The Keras implementation of ResNet50 was used to extract the features from the images. The code with the bests results can be found at the project's notebook.

Data Partition

Before pre-process, data was divided into training, validation and testing subsets. The validation subset was made with all the images taken from one of the subjects from the *Specially Made* dataset. The testing subset was created from all the images of another subject from the same dataset. This partition enabled us to measure how the classifier generalizes with new subjects.

The first experiments consisted in training using only the specially made dataset, augmented using a factor of k = 10. Thus, the dataset had a total of 630 original images, augmented to 6300. The training subset consisted of 450 (71%) original images, augmented to 4500. Test and validation subsets had 90 (14%) original images each (900 augmented). There were poor results with this data configuration. There was a quick tendency of increasing the validation error during training. Also, the number of trainable weights (6147) was high in comparison with the number of training examples.

The next experiments added the SenseZ3D dataset and changed the augmentation factor to k = 5 (*see Refinement*). This new dataset configuration had a total of 990 original images, augmented to 4950. The training subset consisted of

810 (82%) original images, augmented to 4500. Test and validation subsets had 90 (9%) original images each (900 augmented). This configuration doubled the number of original training images and improved the classifier's performance. However in this case, the number of testing and validation images is below the usual standards. It was made this way because of the lack of data available for training. The usual costs of using small testing/validation subsets are the uncertainty of the generalization measure (we can't be so certain about our measured generalization power) and the high deviation of results from the same model architecture (same training experiments with different initializations may deliver different results). Nevertheless, as new subjects were used on the validation/test subsets, generalization measure could be good enough for our purposes (note that the need of using more data is one of the main conclusions of this project).

Training

The model was feed with the preprocessed images as described on previous sections.

RMSProp was used to train the proposed model.The best results were obtained with the default parameters (lr=0.001, rho=0.9, epsilon=1e-08, decay=0.0). However, it was difficult to test different parameters because of the high variation of results (small test subset as explained before).

Training was made considering a batch size of 400 and 200 epochs. The training curves are shown in the following figures:

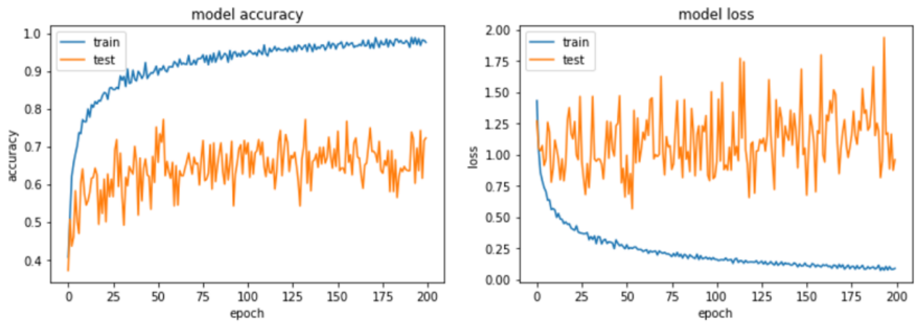


Figure 7: training curves

Testing

The resulting model was evaluated using the preprocessed and augmented test dataset. The accuracy score was calculated and also a confusion matrix was generated to compare individual class performances.

Refinement

As stated before, it was difficult to refine parameters because of the lack of data and the consequent variance of the accuracy for different initializations (~5%).

Several combinations of number of epochs and batch size were studied without significant conclusions.

Two values of the dataset augmentation factor k were tested using the optimal model. The value of 10 revealed to be not better than k factor value of 5. As there weren't too much samples for this project, data augmentation sounded like a very good option to improve results. However, adding more than 5 augmentation images wasn't making the results any better. This can be explained by the deepness of the ResNet50. This model was trained on a huge augmented dataset and as it is shown on the literature, most of the last layers of a deep neural network works over high-level object representations. Thus it is probable that most of the features we are using to feed the proposed model are already robust to rotation and scaling transformations. The final code uses k = 5 as it proved to work well and it is also more efficient than any bigger value.

IV. Results

Model Evaluation and Validation

The selected model achieved a mean accuracy of 78.1% over the test dataset (standard deviation of 1.82).

The model's accuracy was measured on several experiments, using the same conditions but different initial weights and data sorting. Results are resumed in the following table:

Test ID	Accuracy Score[%]
1	78.2
2	79.1
3	80.2
4	77.8
5	75.3

This results show that it is possible to obtain a model that generalizes well. However, the testing dataset contains just 1 subject, so it is important to consider more subjects in the future work to be able to predict better the generalization power of the obtained model.

As announced before, there is also high variance in the result just by changing initialization weights and the order of the samples input. This is another reason to believe in adding more data to the test subset to help us to be more confident about generalization.

The model from test number 1 (closest to the mean) was used to generate the following non-normalized confusion matrix:

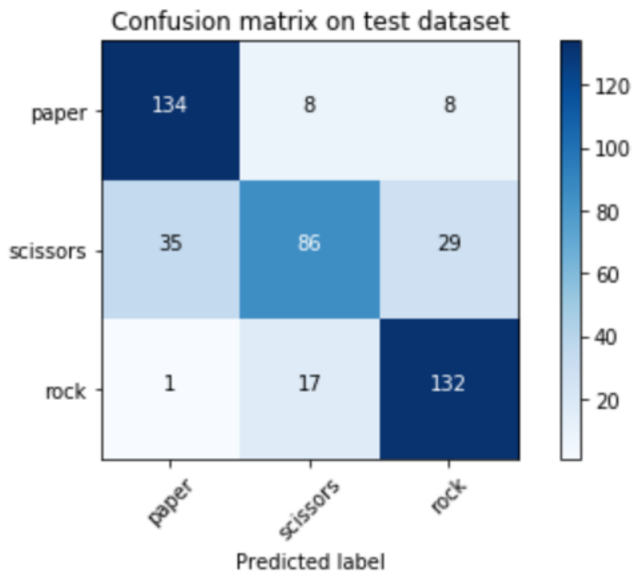


Figure 8: Confusion matrix

The confusion matrix shows that the proposed classifier performs better on Rock and Paper images than Scissors. The images of this last category are often misclassified.

The results are good, far better than a random classifier. However there's still work to do in order to be able to perform on a website.

Justification

The benchmarking model reported 85.8[%] in the static hand gesture classification task. This result is better than the obtained by the proposed classifier (78.1[%]). However there is plenty of evidence that the proposed architecture could do better if the dataset cardinality is increased. One of this evidences is the fact that the weights of the model are more numerous than the total quantity of training samples, so the improvement in generalization is very probable if more data is added.

The proposed evaluation dataset is also more complex than the one of the benchmarking model, but the last one solved a classification problem of 12 classes. It is not possible to compare the scores directly.

The obtained score doesn't reach the benchmarking threshold and neither seems to be reliable enough for using in production environment. However, it is very probable that the use of more data could change this result.

V. Conclusion

Free-Form Visualization

Some results over random images from the testing subject are displayed below:

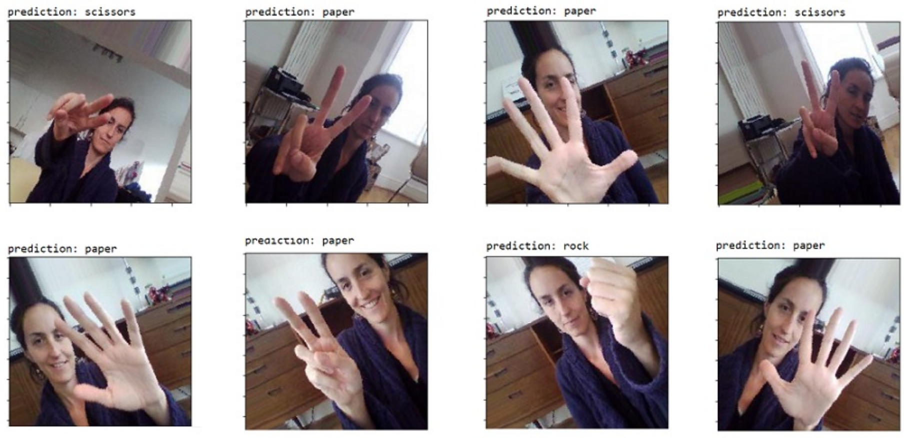


Figure 9: Some predictions over testing subset

The above figure shows the proposed classifier in action. There's still work to do to improve scissors classification.

Reflection

An automatic classifier of rock-paper-scissor hand gesture images was created. The goal was to create an algorithm able to perform in a game that allows the user to play rock-paper-scissors with just the screen and a webcam as interface.

The proposed algorithm receives color images of any size, and after preprocessing the input (resizing, rescaling), it takes features from images using the ResNet50 network implementation from Keras. This features are used to feed a fully connected neural network layer which classifies the input into three different categories: Rock, Paper, Scissors.

A new database was created in order to make training and testing of the classification layer. Two different subjects were used to generate the testing and validation subsets, without mixing their images with the training set. Also a dataset from a more general hand gesture study was used to increase the number of samples used for training. The dataset was also augmented to generate more images.

A simple fully connected layer was trained using RMSProp algorithm. Accuracy scores were calculated from several training experiments and the confusion matrix of the selected model was generated.

The proposed algorithm still needs some performance improvements to reach benchmarking standards. Also needs to have better accuracy to be able to be used on a production environment. Nevertheless, there is the intuition that it could be possible to achieve better results by making a larger dataset.

It is very interesting to note that to create this classifier it wasn't needed any special knowledge, like digital image processing or hand anatomy studies. Just the correct images were fed to a general purpose model and good results were obtained.

It was also very interesting to note that, from certain point, adding data augmentation wasn't helping anymore. This could be due to the transformation invariance of ResNet50 features.

Improvement

The need of more examples has been extensively remarked along this report. There are also other improvements opportunities:

- **Feature Selection**: ResNet50 was trained to detect 1000 different objects and it is probable that there are a lot of features that aren't relevant to our classification task. Thus, a method like PCA could help us to search for the most relevant features and reduce the complexity of the classifier.
- **Fine tuning**: With more samples and a more significant test dataset, the fine tuning of this methodology could improve results. It would be important to test different values for the learning rate.
- **Train another model**: It could be interesting to try another model from scratch. To make this, it would be necessary to have more samples to be able to train a deep network

---

References

- [1] Ren, Zhou, et al. "Robust hand gesture recognition with kinect sensor." Proceedings of the 19th ACM international conference on Multimedia. ACM, 2011.

- [2] Suarez, Jesus, and Robin R. Murphy. "Hand gesture recognition with depth images: A review." Ro-man, 2012 IEEE. IEEE, 2012.
- [3] Chen, Qing, Nicolas D. Georganas, and Emil M. Petriu. "Real-time vision-based hand gesture recognition using haar-like features." Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007. IEEE. IEEE, 2007.
- [4] Minto, L., and P. Zanuttigh. "Exploiting silhouette descriptors and synthetic data for hand gesture recognition." (2015).
- [5] Memo, Alvise, and Pietro Zanuttigh. "Head-mounted gesture controlled interface for human-computer interaction." Multimedia Tools and Applications (2016): 1-27.
- [6] Triesch, Jochen, and Christoph Von Der Malsburg. "A system for person-independent hand posture recognition against complex backgrounds." IEEE Transactions on Pattern Analysis and Machine Intelligence 23.12 (2001): 1449-1453.
- [7] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.