Hand Gesture Recognition with 3D Convolutional Neural Networks

Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz
NVIDIA, Santa Clara, California, USA

Abstract

Touchless hand gesture recognition systems are becoming important in automotive user interfaces as they improve safety and comfort. Various computer vision algorithms have employed color and depth cameras for hand gesture recognition, but robust classification of gestures from different subjects performed under widely varying lighting conditions is still challenging. We propose an algorithm for drivers' hand gesture recognition from challenging depth and intensity data using 3D convolutional neural networks. Our solution combines information from multiple spatial scales for the final prediction. It also employs spatio-temporal data augmentation for more effective training and to reduce potential overfitting. Our method achieves a correct classification rate of 77.5% on the VIVA challenge dataset.

1. INTRODUCTION

Hand gesture recognition is important for designing touchless interfaces in cars. Such interfaces allow drivers to focus on driving while interacting with other controls, *e.g.*, audio and air conditioning, and thus improve drivers' safety and comfort. In the last decade, many vision-based dynamic hand gesture recognition algorithms were introduced [11, 16]. To recognize gestures, different features such as hand-crafted spatio-temporal descriptors [23] and articulated models [9], were used. As gesture classifiers, hidden Markov models [20], conditional random fields [24] and support vector machines (SVM) [4] have been widely used. However, robust classification of gestures under widely varying lighting conditions, and from different subjects is still a challenging problem [25, 1, 15].

To improve classification accuracy, gesture recognition methods with multi-modal sensors were introduced [14, 21, 12, 5, 13]. Neverova et al. successfully combined RGBD data from the hand region with upper-body skeletal motion data using convolutional neural networks (CNNs) for recognizing 20 Italian sign language gestures [13]. However, their technique was intended for gestures performed indoors only. Ohn-Bar and Trivedi evaluated various hand-crafted spatio-temporal features and classifiers for in-car hand-gesture recognition with RGBD data [14]. They reported the best performance with a combination of histogram of gradient (HOG) features and an SVM classifier. Molchanov et al. [12] fused information of hand gestures from depth, color and radar sensors and jointly trained a convolutional neural network with it. They demonstrated successful classification results for widely varying lighting conditions, which motivated our work.

Recently, classification with deep convolutional neural networks has been successful in various recognition challenges [2, 8, 10, 18]. Multi-column deep CNNs that employ multiple parallel networks have been shown to improve recognition rates of single networks by 30-80% for various image classification tasks [3]. Similarly, for large scale video classification, Karpathy et al. [7] observed the best results on combining CNNs trained with two separate streams of the original and spatially cropped video frames.

Several authors have emphasized the importance of using many diverse training examples for CNNs [8, 17, 19]. They have proposed data augmentation strategies to prevent CNNs from overfitting when training with datasets containig limited diversity. Krizhevsky et al. [8] employed translation, horizontal flipping and RGB jittering of the training and testing images for classifying them into 1000 categories. Simonyan and Zisserman [19] employed similar spatial augmentation on each video frame to train CNNs for video-based human activity recognition. However, these data augmentation methods were limited to spatial variations only. To add variations to video sequences containing dynamic motion, Pigou et al. [17] temporally translated the video frames in addition to applying spatial transformations.

In this paper, we introduce a hand gesture recognition system that utilizes depth and intensity channels with 3D convolutional neural networks. Motivated by Molchanov et al. [12], we interleave the two channels to build normalized spatio-temporal volumes, and train two separate subnetworks with these volumes. To reduce potential overfitting and improve generalization of the gesture classifier, we propose an effective spatio-temporal data augmentation method to deform the input volumes of hand gestures. The

augmentation method also incorporates existing spatial augmentation techniques [8]. This work bears similarities to the multi-sensor approach of Molchanov et al. [12], but differs in the the use of two separate sub-networks and data augmentation.

We demonstrate that our system, with two sub-networks, that employs spatio-temporal data augmentation for training, outperforms both a single CNN and the baseline feature-based algorithm [14] on the VIVA challenge's dataset.

2. METHOD

We use a convolutional neural network classifier for dynamic hand gesture recognition. Sec. 2.1, briefly describes the VIVA challenge's hand gesture dataset used in this paper. Sec. 2.2 to 2.4 describe the preprocessing steps needed for our model, the details of the classifier and the training pipeline for the two sub-networks (Fig. 1). Finally, we introduce a spatio-temporal data augmentation method in Sec. 2.5, and show how it is combined with spatial transformations.

2.1. Dataset

The VIVA challenge was organized to evaluate and advance the state-of-the-art in multi-modal dynamic hand gesture recognition under challenging conditions (with variable lighting and multiple subjects). The VIVA challenge's dataset contains 885 intensity and depth video sequences of 19 different dynamic hand gestures performed by 8 subjects inside a vehicle [14]. Both channels were recorded with the Microsoft Kinect device and have a resolution of 115×250 pixels. The dataset was collected under varying illumination conditions. The gestures were performed either with the right hand by subjects in the driver's seat or with the left hand by subjects in the front passenger's seat. The hand gestures involve hand and/or finger motion.

2.2. Preprocessing

Each hand gesture sequence in the VIVA dataset has a different duration. To normalize the temporal lengths of the gestures, we first re-sampled each gesture sequence to 32 frames using nearest neighbor interpolation (NNI) by dropping or repeating frames [12]. We also spatially down sampled the original intensity and depth images by a factor of 2 to 57×125 pixels. We computed gradients from the intensity channel using the Sobel operator of size 3×3 pixels. Gradients helped to improve robustness to the different illumination conditions present in the dataset. We normalized each channel of a particular gesture's video sequence to be of zero mean and unit variance. This helped our gesture classifier converge faster. The final inputs to the gesture classifier were $57 \times 125 \times 32$ sized columns containing interleaved image gradient and depth frames.

2.3. Classifier

Our convolutional neural network classifier consisted of two sub-networks (Fig. 1): *a high-resolution network* (HRN) and *low-resolution network* (LRN), with network parameters $\mathcal{W}_H$ and $\mathcal{W}_L$, respectively. Each network, with parameters $\mathcal{W}$, produced class-membership probabilities ($P(C|x, \mathcal{W})$ for classes C given the gesture's observation x. We multipled the class-membership probabilities from the two networks element-wise to compute the final class-membership probabilities for the gesture classifier:

$$P(C|x) = P(C|x, \mathcal{W}_L) * P(C|x, \mathcal{W}_H). \quad (1)$$

We predicted the class label $c^* = \arg\max P(C|x)$. The networks contained more than 1.2 million trainable parameters.

The high-resolution network consisted of four 3D convolution layers, each of which was followed by the max-pooling operator. Fig. 1 shows the sizes of the convolution kernels, volumes at each layer, and the pooling operators. We input the output of the fourth 3D convolutional layer to two fully-connected layers (FCLs) with 512 and 256 neurons, respectively. The output of this high-resolution network was a softmax layer, which produced class-membership probabilities ($P(C|x, \mathcal{W}_H)$) for the 19 gesture classes.

We input a spatially down sampled (via NNI) gesture volume of $28 \times 62 \times 32$ interleaved depth and image gradient values to the low-resolution network. Similar to the HRN, LRN also comprised of a number of 3D convolutional layers, each followed by a max-pooling layer, two FCLs, and an output softmax layer that estimated the class-membership probability $P(C|x, \mathcal{W}_L)$ values (Fig. 1).

All the layers in the networks, except for the softmax layers, had rectified linear unit (ReLU) activation functions:

$$f(z) = \max(0, z). \quad (2)$$

We computed the output of the softmax layers as:

$$P(C|x, \mathcal{W}) = \frac{\exp(z_C)}{\sum_q \exp(z_q)}, \quad (3)$$

where z_q was the output of the neuron q.

2.4. Training

The process of training a CNN involves the optimization of the network's parameters $\mathcal{W}$ to minimize a cost function for the dataset $\mathcal{D}$. We selected negative log-likelihood as the cost function:

$$\mathcal{L}(\mathcal{W}, \mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{i=0}^{|\mathcal{D}|} \log\left(P(C^{(i)}|x^{(i)}, \mathcal{W})\right). \quad (4)$$

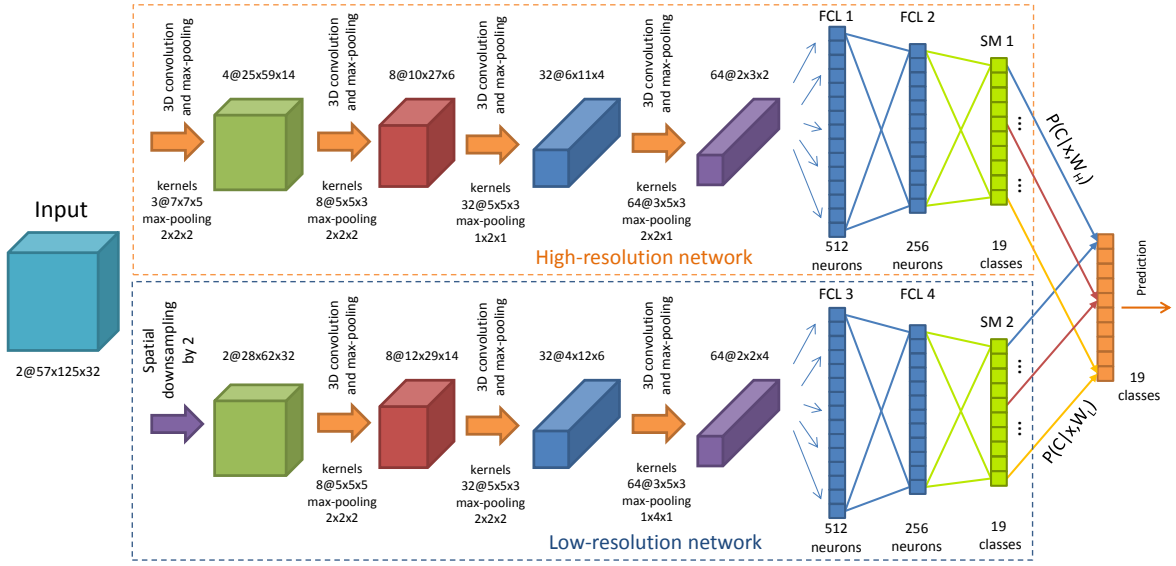

Figure 1: **Overview of our CNN classifier** We used a CNN-based classifier for hand gesture recognition. The inputs to the classifier were $57 \times 125 \times 32$ sized volumes of image gradient and depth values. The classifier consisted of two sub-networks: a high-resolution network (HRN) and a low-resolution network (LRN). The outputs of the sub-networks were class-membership probabilities $P(C|x, \mathcal{W}_H)$ and $P(C|x, \mathcal{W}_L)$, respectively. The two networks were fused by multiplying their respective class-membership probabilities element-wise.

We performed optimization via stochastic gradient descent with mini-batches of 40 and 20 training samples for the LRN and the HRN, respectively. We updated the network's parameters, $w \in \mathcal{W}$ with the *Nesterov accelerated gradient (NAG)* [22] at every iteration i as:

$$\nabla w_i = \left\langle \frac{\delta \mathcal{L}}{\delta(w_{i-1})} \right\rangle_{batch}, \tag{5a}$$

$$v_i = \mu v_{i-1} - \lambda \nabla w_i, \tag{5b}$$

$$w_i = w_{i-1} + \mu v_i - \lambda \nabla w_i, \tag{5c}$$

where λ was the learning rate, μ was the momentum coefficient, ∇w_i was the value of gradient of the cost function with respect to the parameter w_i averaged across the mini-batch. We set the momentum to 0.9. We observed that NAG converged faster than gradient descent with only momentum.

We initialized the weights of the 3D convolution layers with random samples from a uniform distribution between $[-W_b, W_b]$, where $W_b = \sqrt{6/(n_i + n_o)}$, and n_i and n_o were the number of input and output neurons, respectively. We initialized the weights of the fully-connected hidden layers and the softmax layer with random samples from a normal distribution $\mathcal{N}(0, 0.01)$. The biases for all layers, except for the softmax layer, were initialized with a value of 1 in order to have a non-zero partial derivative. For the softmax layers biases were set to 0.

We trained the LRN and the HRN separately and merged them only during the forward propagation stage employed for decision making. We applied weight decay to all convolution layers. After processing each mini-batch we subtracted 0.5% from the network weights. We observed that regularization with weight decay usually led to better generalization for gestures from different subjects. We also applied drop-out (with $p = 0.5$) to the outputs of the fully-connected hidden layers [6]. During drop-out, the outputs were randomly (with $p = 0.5$) set to 0, and were consequently not used in the back-propagation step of that training iteration. For the forward propagation stage, the weights of the layer following the dropped layer were multiplied by 2 to compensate for the effect of drop-out.

For tuning the learning rate, we first initialized the rate to 0.005 and reduced it by a factor of 2 if the cost function did not improve by more than 10% in the preceding 40 epochs. We terminated network training after the learning rate had decayed at least 4 times or if the number of epochs had exceeded 300. Since the dataset is small, we did not reserve data from any subjects to construct a validation set. Instead, we selected the network configuration that resulted in the smallest error on the training set.

2.5. Spatio-temporal Data augmentation

The VIVA challenge dataset contains less than 750 gestures for training, which are not enough to prevent overfitting. To avoid overfitting we performed offline and online spatio-temporal data augmentation. Note that we did not augment the test data.

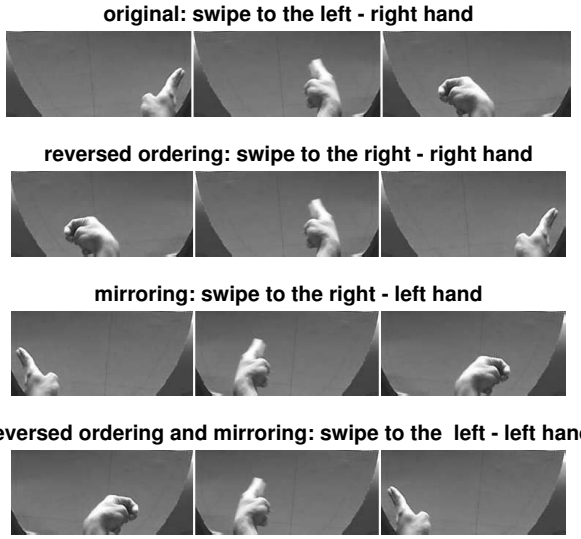

Figure 2: **Offline data augmentation** We used three transformations on the original data: *reversed ordering* of frames only, *mirroring* of frames only, and *reversed ordering and mirroring* of frames.

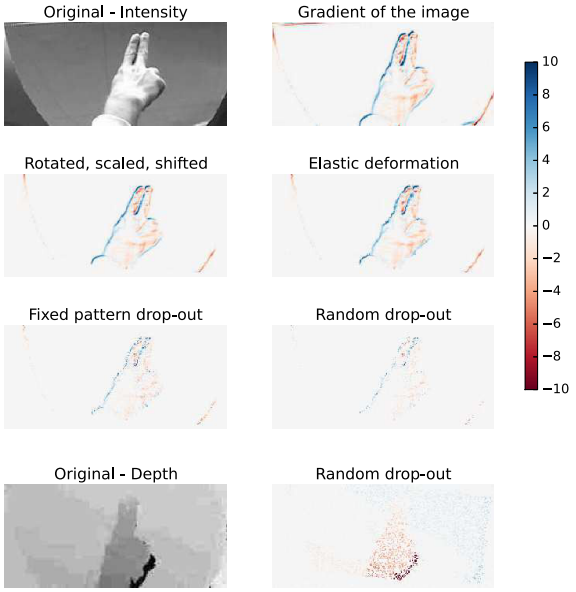

Figure 3: **Online data augmentation** The top three rows show augmentation of the intensity channel. The last shows augmentation of the depth channel.

Offline data augmentation (Fig. 2) was motivated by existing methods [8, 19], and is comprised of three operations: reverse ordering of frames, horizontal mirroring, and applying both operations together. With these operations we generated additional samples for training. For example, applying both operations transforms the original gesture *swipe to the left* with the *right hand* to a new gesture: *swipe to the left* with the *right hand*.

To further prevent overfitting and to increase the generalization performance of the classifier, we also augmented the data online during training (Fig. 3). During online augmentation, we trained each epoch with different training samples. While the back propagation stage of an epoch was being performed, we concurrently generated the training samples for the next epoch by transforming 50% randomly selected training gestures.

Online data augmentation included spatial and temporal transformations. Spatial augmentation was composed of: *(a) affine transformations*: rotation ($\pm 10^o$), scaling ($\pm 30\%$), translation (± 4 pixels along the x axis, and ± 8 along the y axis), (b) *spatial elastic deformation* [18] with pixel displacement $\alpha = 6$ and standard deviation of the smoothing Gaussian kernel $\sigma = 10$ pixels, (c) *fixed pattern drop-out*: setting $p = 50\%$ randomly selected spatial locations to zero across all frames, (d) *random drop-out*: randomly setting $p = 50\%$ of the pixels in the entire volume of the gesture to zero.

The transformation parameters for (a) and (b) were fixed for a particular gesture and for the two sensor channels for an epoch. The transformation parameters for (a) were drawn from a uniform distribution. During each epoch new transformation parameters for (a)-(d) were drawn from their corresponding distributions. We applied the transformations (c) and (d) independently to the depth and intensity channels. We observed that the CNN learned temporal smoothing filters in the first convolutional layer when we only used *random drop-out* (c) without the *fixed pattern drop-out* (d). When the *fixed pattern drop-out* (d) was applied, we did not observe this since the network was then forced to learn from gestures with missing data in frames at fixed locations.

We additionally transformed the gestures temporally by displacing frames within the input temporal volume (Fig. 5). Temporal augmentation consisted of three stages: (a) scaling the duration of a sequence by $\pm 20\%$, (b) temporally translating them by ± 4 frames, and (c) applying elastic deformation to the temporal domain.

While spatial elastic deformation has been employed previously [18], we extended it to the temporal domain as follows. Temporal elastic deformation (TED) shrinks and stretches a video without altering its length. Fig. 4 shows how temporal deformation is applied to a video sequence. The key characteristic of TED is determined by its principal point $g = [n, m]$, which is randomly sampled from the distributions:

$$n \in \mathcal{N}(M, 4), \quad m \in \mathcal{N}\big(n, 4 * (1 - |n - M|/M)\big), \quad (6)$$

where $\mathcal{N}(a, b)$ is a normal distribution with mean a and

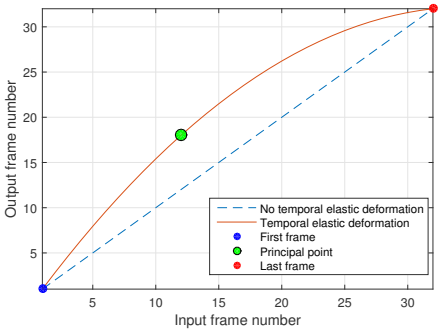

Figure 4: **Temporal Elastic Deformation** TED temporally warps video sequences while maintaining the order of its frames and the size of its volume. Its key characteristic is the principal point. After the deformation, the frames are re-mapped according to the temporal elastic deformation curve.

standard deviation b, and M is half of the gesture's temporal length. The elastic deformation function $\chi(g)$ is approximated as a polynomial of order 2 that fits the positions of three points: the first frame $[1, 1]$, the principal point $g = [n, m]$ and the last frame $[T, T]$. Finally, the frames within the gesture's volume are interpolated using the function $\chi(g)$ via nearest-neighbor interpolation. The parameters of spatial and temporal data augmentation were randomly sampled from the corresponding distributions as described above. After each epoch, all the spatial and temporal transformations were re-applied to the gesture in the training dataset.

3. RESULTS

We evaluated the performance of our dynamic hand gesture recognition system using leave-one-subject-out cross-validation on the VIVA challenge's dataset [14]. We used data from one of the 8 subjects for testing and trained the classifier with data from the 7 remaining subjects; we repeated this process for each of the 8 subjects and averaged the accuracy. Fig. 6 shows the performance of the LRN during training. We applied various forms of regularization to the network in order to prevent overfitting even after a large number of training epochs. Data augmentation and drop-out were key components to successful generalization of the classifier.

The correct classification rates for our gesture recognition system are listed in Table 1. We compared our classifier to the baseline method proposed by Ohn-Bar and Trivedi [14], which employs HOG+HOG^2 features. Both the low and high resolution convolutional neural networks that we proposed, outperformed Ohn-Bar and Trivedi's method by 9.8% and 5.5%, respectively. Furthermore, the final classifier that combined the outputs of LRN and HRN outperformed the baseline method by 13.0%. Moreover, 52% of the final classifier's errors were associated with the second most probable class. The results indicate that our CNN-based classifier for in-car dynamic hand gesture recognition considerably outperforms approaches that employ hand-crafted features.

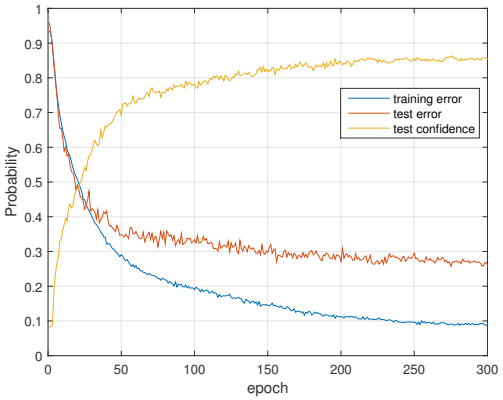

Figure 6: **LRN Training** The curves show average values for leave-one-subject-out cross-validation. Observe that the network does not overfit even after a very large number of epochs.

Method	HOG [14]	LRN	HRN	LRN+HRN
Mean	64.5%	74.4%	70.0%	**77.5**$\%$
Std	16.9%	8.9%	7.8%	7.9%

Table 1: **The classification results for leave-one-subject-out cross-validation** Both the LRN and the HRN outperformed the HOG-based approach [14]. Our final classifier, which combined the LRN and the HRN resulted in the best performance.

CNNs have been shown to be effective at combining data from different sensors [12]. We compared the performance of early and late fusion of sensors in the CNN architecture. In Table 2, we present the correct classification rates for the LRN trained with different input modalities. Observe that, individually, the depth data (accuracy = 65%) performed better than the intensity data (accuracy = 57%). On element-wise late multiplying the class-membership probabilities of the two LRNs trained individually with the two modalities, the accuracy improved to 70.4%. However, the highest accuracy of 74.4% was obtained when the LRN was trained with interleaved depth and image gradient frames as input.

In Table 3, we present the correct classification rates for the low resolution network with different forms of data augmentation. We observed that the training error increased on enabling data augmentation. However, the test error decreased. This demonstrates that the proposed data aug-

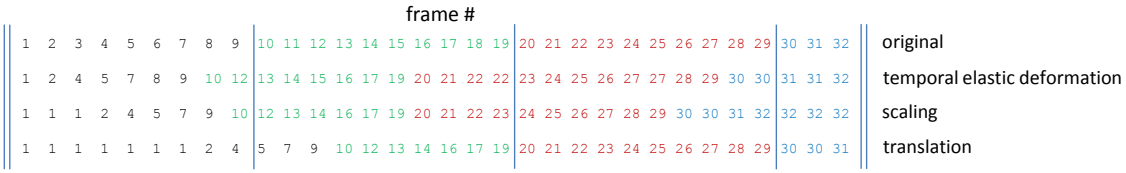

Figure 5: **Examples of temporal data augmentation**. The numbers in each row correspond to the frame IDs in the original sequence (top row). Note how the order and the occurrence of frames is changed by incrementally applying temporal elastic deformation (TED) (2nd row), TED + scaling (3rd row) and TED + scaling + translation (4th row) to the original input.

LRN_{RGB}	LRN_{D}	LRN_{RGB}+LRN_{D}	LRN_{RGBD}
57.0%	65.0%	70.4%	74.4%

Table 2: **The accuracy of LRN with different inputs** Leave-one-subject-out cross-validation results of LRN trained with different inputs. The best results were obtained when we interleaved the intensity and depth frames and input them to the CNN.

	None	Offline	Online	Both
Train	99.9%	99.8%	93.0%	91.1%
Test	48.3%	56.2%	59.1%	74.4%

Table 3: **Classification results with data augmentation** The average correct classification rates for LRN with different forms of data augmentation. When we employed both forms of data augmentation the accuracy for the test set increased considerably.

mentation method successfully reduced over-fitting and improved generalization of the gesture classifier. Additionally, we observed that image gradients increased the final correct classification rate by 1.1%, and spatial and temporal elastic deformations applied to the training data increased it by 1.2% and 1.72%, respectively.

Table 4 shows the confusion matrix of our proposed final classifier. Our classifier often confused between the *Swipe* and *Scroll* gestures performed along the same direction. Many gestures were mis-classified as the *Swipe down* gesture. The *Rotate CW/CCW* gestures were difficult for the classifier. The classifier also had difficulties with distinguishing between the *Swipe +* and the *Swipe X* gestures. The *Tap-3* gesture produced 38% of the miss-classifications.

The classifier's less confident decisions can be rejected by setting an empirical threshold. This helps to increase the correct classification rate, but at the cost of a greater number of missed gestures. Fig. 7 demonstrates this tradeoff for our gesture classifier at various confidence threshold values.

It took 1.83 and 2.5 hours to train the LRN and the HRN, respectively, on an NVIDIA Tesla K20m GPU for a single fold of the leave-one-subject-out cross-validation. To propagate a gesture forward though the LRN it took $20ms$ on

Class	1. Swipe R	2. Swipe L	3. Swipe D	4. Swipe U	5. Swipe V	6. Swipe X	7. Swipe +	8. Scroll R	9. Scroll L	10. Scroll D	11. Scroll U	12. Tap-1	13. Tap-3	14. Pinch	15. Expand	16. Rotate CCW	17. Rotate CW	18. Open	19. Close
1.	**77**							12						2	2		7		
2.		**74**	2		7				7				2	4	2	2			
3.			**66**							9			9	2		4	4	2	4
4.				**90**							2	4				2	2		
5.		12			**80**							2	6						
6.			2		5	**70**	9					5	2	2			5		
7.	2	2				21	**62**									2	11		
8.	9							**91**											
9.		8							**90**						2				
10.			11							**82**		7							
11.											**96**	4							
12.			7									**82**	7		4				
13.	8		10	2		2						6	**64**	2	2	2	2		
14.			19									2		**75**		2			2
15.			4										4		**74**	6	10	2	
16.			11	7			2						2	2	4	**65**	7		
17.			4	10								2	2	2		8	**70**		2
18.				2														**88**	10
19.			2											4				10	**84**

Table 4: **The confusion matrix for our proposed gesture classifier.** Abbreviations: L - left, R - right, D - down, U - up, CW/CCW - clock/counter-clock wise.

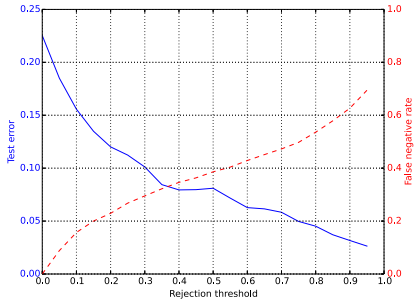

Figure 7: **Accuracy versus False negative rate** The classification error for the test set and the false negative rate as a function of the rejection threshold. On increasing the rejection threshold, the accuracy of the classifier increases at the cost of increased falsely rejected gestures.

the GPU and $78ms$ on an Intel Core i5-3210m CPU. For the HRN, it took $68ms$ on the GPU and $196ms$ on the CPU. On dividing by the number of frames in each gesture (32)

we get 400 FPS for LRN, 160 FPS for HRN and 110 FPS for their combination. The decision making time for our classifier is nearly half of that of the baseline HOG+HOG^2 features-based method (50 FPS) [14].

4. CONCLUSIONS

We developed an effective method for dynamic hand gesture recognition with 3D convolutional neural networks. The proposed classifier uses a fused motion volume of normalized depth and image gradient values, and utilizes spatio-temporal data augmentation to avoid overfitting.

By means of extensive evaluation, we demonstrated that the combination of low and high resolution sub-networks improves classification accuracy considerably. We further demonstrated that the proposed data augmentation technique plays an important role in achieving superior performance. For the challenging VIVA dataset, our proposed system achieved a classification rate of 77.5%. Our future work will include more adaptive selection of the optimal hyperparameters of the CNNs, and investigating robust classifiers that can classify higher level dynamic gestures including activities and motion contexts.

References

[1] F. Althoff, R. Lindl, and L. Walchshäusl. Robust multimodal hand-and head gesture recognition for controlling automotive infotainment systems. In *VDI-Tagung: Der Fahrer im 21. Jahrhundert*, 2005. 1

[2] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *International Joint Conference on Artificial Intelligence*, pages 1237–1242, 2011. 1

[3] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR*, pages 3642–3649, 2012. 1

[4] N. Dardas and N. D. Georganas. Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. *IEEE Transactions on Instrumentation and Measurement*, 60(11):3592–3607, 2011. 1

[5] S. Escalera, X. Bar, J. Gonzlez, M. A. Bautista, M. Madadi, M. Reyes, V. Ponce, H. J. Escalante, J. Shotton, and I. Guyon. Chalearn looking at people challenge 2014: Dataset and results. In *ECCVW*, 2014. 1

[6] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*, 2012. 3

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, pages 1725–1732, 2014. 1

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105. 2012. 1, 2, 4

[9] J. J. LaViola Jr. An introduction to 3D gestural interfaces. In *SIGGRAPH Course*, 2014. 1

[10] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 1

[11] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Systems, Man, and Cybernetics*, 37:311–324, 2007. 1

[12] P. Molchanov, S. Gupta, K. Kim, and K. Pulli. Multi-sensor System for Driver's Hand-gesture Recognition. In *AFGR*, 2015. 1, 2, 5

[13] N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout. Multi-scale deep learning for gesture detection and localization. In *ECCVW*, 2014. 1

[14] E. Ohn-Bar and M. Trivedi. Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. *IEEE Trans. on Intelligent Transportation Systems*, 15(6):1–10, 2014. 1, 2, 5, 7

[15] F. Parada-Loira, E. Gonzalez-Agulla, and J. Alba-Castro. Hand gestures to control infotainment equipment in cars. In *IEEE Intelligent Vehicles Symposium*, pages 1–6, 2014. 1

[16] V. I. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *PAMI*, 19:677–695, 1997. 1

[17] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen. Sign language recognition using convolutional neural networks. In *ECCVW*, 2014. 1

[18] P. Y. Simard, D. Steinkraus, and J. C. Platt. J.c.: Best practices for convolutional neural networks applied to visual document analysis. In *Int. Conference on Document Analysis and Recognition*, pages 958–963, 2003. 1, 4

[19] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014. 1, 4

[20] T. Starner, A. Pentland, and J. Weaver. Real-time american sign language recognition using desk and wearable computer based video. *PAMI*, 20(12):1371–1375, 1998. 1

[21] J. Suarez and R. R. Murphy. Hand gesture recognition with depth images: A review. In *ROMAN*, pages 411–417, 2012. 1

[22] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, pages 1139–1147, 2013. 3

[23] P. Trindade, J. Lobo, and J. Barreto. Hand gesture recognition using color and depth images enhanced with hand angular pose data. In *IEEE Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 71–76, 2012. 1

[24] S. B. Wang, A. Quattoni, L. Morency, D. Demirdjian, and T. Darrell. Hidden conditional random fields for gesture recognition. In *CVPR*, pages 1521–1527, 2006. 1

[25] M. Zobl, R. Nieschulz, M. Geiger, M. Lang, and G. Rigoll. Gesture components for natural interaction with in-car devices. In *Gesture-Based Communication in Human-Computer Interaction*, pages 448–459. Springer, 2004. 1