3D gesture classification with convolutional neural networks

Stefan Duffner, Samuel Berlemont, Grégoire Lefebvre, Christophe Garcia

3D GESTURE CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS

Stefan Duffner [(1)], Samuel Berlemont [(1,2)], Grégoire Lefebvre [(2)], Christophe Garcia [(1)]

(1) Université de Lyon, CNRS INSA-Lyon, LIRIS, UMR5205, F-69621, France
(2) Orange Labs, R&D, F-38240 Meylan, France

ABSTRACT

In this paper, we present an approach that classifies 3D gestures using jointly accelerometer and gyroscope signals from a mobile device. The proposed method is based on a convolutional neural network with a specific structure involving a combination of 1D convolution, averaging, and max-pooling operations. It directly classifies the fixed-length input matrix, composed of the normalised sensor data, as one of the gestures to be recognises. Experimental results on different datasets with varying training/testing configurations show that our method outperforms or is on par with current state-of-the-art methods for almost all data configurations.

Index Terms— 3D gesture recognition, convolutional neural network

1. INTRODUCTION

Nowadays, most portable devices such as mobile phones are equipped with inertial sensors like accelerometers and gyroscopes, so-called Micro-Electro-Mechanical (MEM) systems. These sensors measure respectively 3-dimensional linear acceleration and angular velocity and are widely used for entertainment applications, *e.g*. games, among others. The application that we consider here is the recognition of a set of 3D gestures performed by a user to execute commands on the device (see Fig. 1). However, 3D gesture classification based on MEM signals is very challenging due to three factors. First, dynamic variations may occur when users produce intense or phlegmatic gestures, slow or fast gestures. Secondly, semantic variations are possible with users performing several gestures from a large vocabulary with little training or tutorial help. Finally, volumetric variations are challenging from one user in a close world paradigm to multi-users in a open world paradigm (*e.g*. human ability, left or right-handed, on the move, in different contexts *etc*.). Classically, several processing steps are needed to deal with these variations: input data processing to reduce noise and enhance relevant information, data clustering to reduce dimensionality, and gesture model learning to build a strong classifier.

In this paper, we propose a novel gesture classification method based on a convolutional neural network (ConvNet) that operates on fixed-length, *i.e*. time-normalised, MEM data. As opposed to classical approaches, the proposed algorithm is able, with neither advanced pre-processing nor specific feature modelling or learning, to automatically extract and learn prominent features from the data as well as effectively classify them into one of the 14 gesture classes. With a thorough evaluation on different datasets, we show that the proposed approach outperforms or is on par with state-of-the-art methods that explicitly model temporal sequences (like Hidden Markov Models) and/or involve "hand-crafted" feature selection.

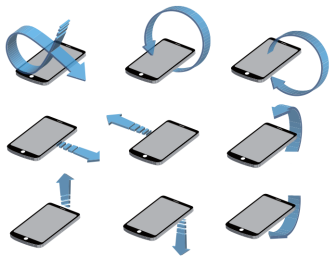

Fig. 1. Illustration of 9 gestures to be recognised.

2. RELATED WORK

In the recent literature, three main strategies exist to deal with 3D accelerometer-based gesture recognition: probabilistic temporal signal modelling, temporal warping or statistical machine learning.

The probabilistic approach has mainly been studied with discrete [1, 2, 3] and continuous HMMs [4]. For instance, Kela *et al*. [3] use discrete HMMs (dHMM) from gesture velocity profiles. The first step is the input data space clustering in order to build a feature vector codebook. The second one consists in creating a discrete HMM using the sequences of vector codebook indexes. A correct recognition rate of 96.1% is obtained with 5 HMM states and a codebook size of 8 from 8 gestures realised by 37 users. In order to use gesture data correlation in time, Pylvänäinen [4] proposes a system based on a continuous HMM (cHMM) achieving a recognition rate of 96.76% on a dataset with 20 samples for 10 gestures realised by 7 persons.

The second approach is based on temporal warping from

a set of reference gestures [5, 6, 7]. Liu *et al*. [6] present a method using Dynamic Time Warping (DTW) from pre-processed signal data that gives gesture recognition and user identification rates of respectively 93.5% and 88%, outperforming in this study the HMM-based approach.

The third strategy is based on a specific classifier [8, 9, 10]. Hoffman *et al*. [8] propose a linear classifier and Adaboost, resulting in a recognition rate of 98% for 13 gestures performed by 17 participants. The study of Wu *et al*. [9] proposes to construct fixed-length feature vectors from the temporal input signal to be classified with Support Vector Machines (SVM). Each gesture is then segmented in time and statistical measures (mean, energy, entropy, standard deviation and correlation) are computed for each segment to form the final feature vectors. The resulting recognition rate is 95.21% for 12 gestures made by 10 individuals, outperforming in this study the DTW results. Finally, the recent study by Lefebvre *et al*. [10] proposes a method based on Bidirectional Long-Short-Term Memory Recurrent Neural Networks (BLSTM-RNN see [11]), which classifies sequences of raw MEM data with very good accuracy, outperforming classical HMM and DTW methods.

The algorithm proposed in this paper uses convolutional neural networks (ConvNet) [12, 13] and belongs to the last category. However, we do not partition the input signal into smaller time segments or sequentially process the data like with HMMs [1, 2, 4, 3], time warping based methods [5, 6, 7] or recurrent neural networks [10]. The model is rather trained and applied on the whole fixed-length gesture vectors, which avoids the error-prone step of segment length and boundary determination. Also, by using a ConvNet, features are automatically learnt from the raw (normalised) input signal. Thus, no "hand-crafted" feature design (such as for HMM codebooks or statistical descriptors) is required.

3. THE PROPOSED APPROACH

The proposed method is based on a ConvNet algorithm that jointly classifies the accelerometer and gyroscope data of a gesture as one of the N_G gestures to be recognised. The MEM data that correspond to a gesture are normalised to a fixed-size matrix (here: 45×6; 45 time steps and 6 inertial features) and then input to the network. The contribution of this paper is an effective ConvNet architecture that operates on *fixed-size* temporal MEM data, using 1D convolutions over time. The first layers of this architecture automatically learn to extract complex temporal patterns in the feature space, and the final layer then fuses the information from different sensors.

In the following we will describe the data normalisation procedure and then the proposed ConvNet classifier.

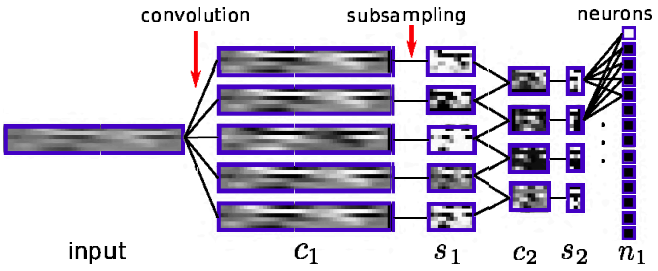

Fig. 2. Basic ConvNet (A1) with alternating convolution and subsampling layers, and visualisation of the classification of a gesture (white: high activation, black: low activation).

3.1. Pre-processing

Gesture data are preprocessed in three steps: amplitude scaling (normalisation), filtering and temporal scaling.

Gesture normalisation consists in dividing every sample of the gesture by the maximum norm over the samples. It provides a linear scaling of every sensor-axis data between -1 and $+1$, effectively keeping the accelerometer/gyroscope amplitude ratio. Then, a discrete low-pass filter is applied, giving a decreasing importance to past samples as well as reducing the influence of noise: $g_f(t+1) = (1-\beta)\cdot g(t+1) + \beta \cdot g_f(t)$ with g being the signal before filtering and g_f the signal after filtering, $\beta = 0.7$. In the last step (similar to [14]), temporal scaling is carried out by setting a common duration for every gesture. To this end, the gesture curvilinear length is approximated by summing the Euclidean distances between successive samples and dividing them into equal intervals to get the curvilinear coordinates of the new samples. Then, these new samples are computed using a linear interpolation of the existing samples, and they directly form the input to the ConvNet.

3.2. Classification

We propose a specific convolutional neural network architecture to classify the normalised MEM data of fixed size 45×6 as one of the 14 gestures by activating one of the 14 output neurons. The overall network architecture is illustrated in Fig. 2. It comprises five layers (excluding the input layer): alternating convolution and subsampling layers c_1, s_1, c_2, and s_2 for low-level feature extraction, and a neuron layer n performing the final classification. Each layer contains a certain number of maps or neurons that we varied in our experiments (see Section 4). All parameters are trained jointly producing a classifier with tightly integrated feature extraction, thus avoiding prior *"hand-crafted"* feature design.

Classical ConvNets [12, 13] have been used successfully for computer vision tasks, where the input is a 2D image or image sequence and the convolution filters extract simple 2D features (like edges) in the first layer(s) and more complex features (like corners) in the subsequents layer(s). Recently, ConvNets have also been applied to speech recognition problems [15], where for example, the convolutions are performed

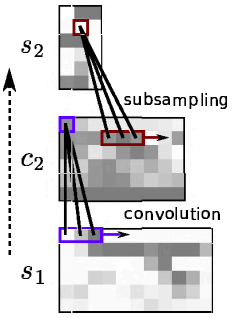

Fig. 3. Operation of 1D convolution and subsampling (max-pooling) layers (here: c_2 and s_2).

over different frequency bands. In our case, the input feature matrix contains 3D data from different sensors over time. 2D convolutions in this feature space make less sense, and we experimentally show that, for our application, convolution maps with 1D convolutions over time give the best results.

The operation of one convolution map and a subsequent subsampling map is illustrated in Fig. 3. The output of a convolution map i is computed as:

$$c_i(x,y) = \sum_{\substack{k\in\{1..u_i\}\\ l\in\{1..v_i\}}} w_{i,k,l} I(x+k, y+l) + b_i \ , \quad (1)$$

where I is the map of the previous layer connected to c_i (can be the input map or s_1); u_i and v_i define the size of the convolution kernel, w_i are the trainable weights that form the kernel (shared over all positions x, y), and b_i is the bias. If several input maps are connected to one convolution map, the output is simply the sum of both convolutions.

Two types of subsampling maps follow the convolution layers. In s_1 we use *averaging* maps:

$$s_1(x,y) = \phi\Big(w_{s1} \sum_{\substack{k\in\{1..p_1\}\\ l\in\{1..q_1\}}} c_1(xp_1+k, yq_1+l) + b_{s1}\Big) \ , \quad (2)$$

where w_{s1} and b_{s1} are the trainable weight and bias, p_1 and q_1 define the size of the subsampling kernel, and the activation function $\phi(x) = \tanh(x)$.

The subsampling layer s_2 contains *max-pooling* maps:

$$s_2(x,y) = \phi\left(\max_{\substack{k\in\{1..p_2\}\\ l\in\{1..q_2\}}} c_2(xp_2+k, yq_2+l)\right) \ , \quad (3)$$

which simply output the maximum for each non-overlapping region, *i.e*. there are no trainable parameters.

The final neuron layer contains one neuron for each gesture class to be recognised. It computes:

$$n(i) = \sum_{k,l} w_{n,k,l} s_2(k,l) + b_n \ , \quad (4)$$

where $w_{n,k,l}$ and b_n are the trainable weights and bias. Here the neurons $n(i)$ are fully connected to every neuron of s_2.

The output of the neurons $n(i)$ goes through a softmax activation function that ensures that all values are in $[0,1]$:

$$o(i) = \frac{exp(n(i))}{\sum_{k=1}^{N_G} exp(n(k))} \quad (5)$$

with $N_G = 14$. The final output $o(i)$ is supposed to be 1 for the neuron representing the desired class and 0 for the others. To train the network, the standard online error back-propagation algorithm is used, minimising the energy function $E = \sum_{j=1}^{N}\sum_{i=1}^{N_G} \frac{1}{2}(o_j(i) - t_j(i))^2$, with $t_j \in \{0,1\}$ being the desired output value of example j, and N the number of training examples.

Having trained the neural network, a new gesture is classified as gesture g by simply propagating the input pattern forward and computing: $g = \mathrm{argmax}_i\, o(i)$.

4. EXPERIMENTAL RESULTS

4.1. Datasets & Testing Protocols

We collected two datasets (on an Android Nexus S Samsung smartphone), DB1 and DB2, comprising 14 different gesture classes. Our experiments have been carried out with four different configurations over these datasets, (protocols) P1-P4, which are oriented towards different application scenarios.

The first dataset DB1 corresponds to a single user recording 40 samples for every gesture class. DB2 contains samples from 22 different users, with five samples per class for each user. The 14 symbolic gestures comprise two families: linear gestures (e.g. north, south, east and west flicks, and up, down, pick and throw gestures) and curvilinear gestures (e.g. alpha, heart, letter N, letter Z, clockwise and counter-clockwise).

From DB1 and DB2, four configurations were created. The first protocol, P1, is based on DB1. Considering a single user, it is oriented towards user personalisation applications, with five samples used for training and 16 for testing. The other three protocols are based on DB2. P2 is the multi-user case, testing the ability of a classifier to absorb the variability of different users performing the same gesture. From the five samples available for each class and user, three are used for training (924 in total) and two for testing (616 in total). P3 is a more realistic open-world scenario, where every user cannot be represented in the training set and it is up to the system to generalise to new users without the need of any additional personalisation. Its training set is composed of all the samples from 17 users, and the five remaining users' samples form the test set. Finally, P4 is the most challenging case, in terms of the generalisation capacity of the classifiers: the five samples from one user form the training set, while the other 1470 from unknown users are used for testing.

Every protocol is repeated 10 times in order to get meaningful and representative results.

	P1	P2	P3	P4
c_2 : 3x1	$\mathbf{0.965} \pm 0.015$	$\mathbf{0.937} \pm 0.010$	$\mathbf{0.915} \pm 0.011$	$\mathbf{0.735} \pm 0.043$
c_2 : 1x3	0.956 ± 0.012	0.927 ± 0.016	0.891 ± 0.026	0.691 ± 0.059
c_2 : 3x3	$\mathbf{0.965} \pm 0.012$	0.935 ± 0.012	0.905 ± 0.020	0.723 ± 0.043
s_2 : avg	0.959 ± 0.015	0.936 ± 0.008	$\mathbf{0.915} \pm 0.020$	0.684 ± 0.064

Table 1. Recognition rates for different kernels (for a 5-5-4-4-14 architecture). Kernel sizes for c_2 are: 3×1 (temporal convolutions), 1×3 (feature convolutions), and 3×3 (temporal+feature convolutions). The last row shows the results with the average operation for s_2 instead of max-pooling.

	P1	P2	P3	P4
A1	0.965 ± 0.015	0.937 ± 0.010	0.915 ± 0.011	0.735 ± 0.043
A2	$\mathbf{0.983} \pm 0.006$	0.953 ± 0.006	0.933 ± 0.011	0.752 ± 0.031
A3	0.979 ± 0.005	$\mathbf{0.985} \pm 0.009$	$\mathbf{0.934} \pm 0.015$	$\mathbf{0.787} \pm 0.034$

Table 2. Recognition rates for different network architectures (A1-A3) of increasing complexity.

4.2. Results

For all evaluations, we measure the overall recognition rate, *i.e*. the number of correctly recognised gestures divided by the total number of gestures in the test set. The rates for ConvNets are averaged over 30 runs with random weight initialisation. The ConvNets have been trained for 5000 iterations, where early stopping is performed with a separate validation set composed of 10% random samples from the training set.

4.2.1. Different network architectures

In the first set of experiments, we study the influence of the network architecture, *i.e*. kernel sizes and number of maps, on the recognition performance. Due to limited space, we only present the most important results concerning the choice of these different hyper-parameters.

The input is a matrix of 45 time steps times 6 features. Kernels in c_1 are of size 2×1, *i.e*. very simple 1D temporal filters, followed by 4×1 averaging kernels in s_1. The latter basically reduce the impact of temporal shifts of the input signal. For c_2, we evaluated the performance of a 5-5-4-4-14 ConvNet (*c.f*. Fig. 2) using kernel sizes: 3×1 (temporal convolution), 1×3 (feature convolution), and 3×3 (temporal + feature convolutions). Table 1 shows the resulting recognition rates (and standard deviations). Temporal convolutions produce the best results on all datasets. Layer s_4 contains max-pooling maps with window size 4×1, obtaining temporal invariance of more complex features over a certain time frame. The last row in Table 1 shows that replacing these maps with averaging maps (as in s_1) decreases the performance.

We further varied the number of maps in each layer. Table 2 shows the results for three different architectures of increasing complexity. A1 corresponds to the 5-5-4-4-14 ConvNet used in the previous experiments. A2 has a 4-4-14-14-14 architecture. The 14 maps in $c2$ are composed of 8 maps resulting from 2 convolutions over each of the 4 maps in s_2, and 6 maps with all combinations of pairs in s_2. Finally, A3 has the same connections scheme as A2 but with more maps, that is 10-10-65-65-14. A3 performs best on almost all datasets except P1 where rates are slightly below the ones of A2.

	P1	P2	P3	P4
DTW	0.997 ± 0.004	0.940 ± 0.002	0.917 ± 0.001	0.781 ± 0.016
cHMM	$\mathbf{0.999} \pm 0.002$	0.858 ± 0.007	0.828 ± 0.001	0.756 ± 0.023
SVM	0.961 ± 0.009	0.949 ± 0.008	0.913 ± 0.002	0.674 ± 0.037
FDB+SVM	0.964 ± 0.019	0.954 ± 0.006	0.924 ± 0.013	0.693 ± 0.041
BLSTM	0.868 ± 0.007	0.956 ± 0.005	0.926 ± 0.029	-
ConvNet	0.979 ± 0.005	$\mathbf{0.958} \pm 0.009$	$\mathbf{0.954} \pm 0.015$	$\mathbf{0.787} \pm 0.034$

Table 3. Recognition rates for the different protocols P1-P4 and for different methods.

4.2.2. Comparison with state-of-the-art methods

Finally, we compared the results of the best ConvNet, A3, to state-of-the-art methods applied to the same datasets. Table 3 summarises the results. The DTW method [16] uses variable-sized signals that are filtered, normalised, vectorised, and then classified by a k-Nearest-Neighbour classifier ($k = 5$). cHMM [10] uses a left-to-right model composed of 12 states and forward jumps limited to three states with filtered, normalised, variable-sized input signals. The method using SVMs works with fixed-size vectors ($45 \times 6 = 270$) of accelerometer and gyroscope data and a linear kernel. The FDB+SVM method ([9]) applies SVMs on 19-dimensional feature descriptors (mean, energy etc.) for 9 time segments and 6 signal dimensions, *i.e*. feature vectors of 1026 dimensions. Finally, the BLSTM method [10] works with the raw input signal and uses a 100-neuron LSTM. The proposed ConvNet outperforms all existing methods or is on par. For P1, the results are slightly worse than with the DTW method. This is probably due to the very small training set in this configuration (only five examples for each gesture).

5. CONCLUSIONS

We presented a 3D gesture recognition approach based on a convolutional neural network. The proposed algorithm classifies normalised fixed-length matrices of accelerometer and gyroscope data using a specific type of architecture involving a cascade of 1D temporal convolution maps and averaging as well as max-pooling maps. Using different datasets and training/testing configurations including 14 different gestures, we studied the influence of various important hyper-parameters (kernel sizes, network size) on the overall performance, and showed for almost all the cases that the recognition rates are superior to those of state-of-the-art methods.

6. REFERENCES

[1] F. G. Hofmann, P. Heyer, and G. Hommel, "Velocity profile based recognition of dynamic gestures with discrete Hidden Markov Models," in *Gesture and Sign Language in Human-Computer Interaction*, vol. 1371 of *Lecture Notes in Computer Science*, pp. 81–95. 1998.

[2] S. Kallio, J. Kela, and J. Mantyjarvi, "Online gesture recognition system for mobile interaction," in *Systems, Man and Cybernetics*, 2003, vol. 3, pp. 2070 – 2076 vol.3.

[3] J. Kela, P. Korpipää, J. Mäntyjärvi, S. Kallio, G. Savino, L. Jozzo, and D. Marca, "Accelerometer-based gesture control for a design environment," *Personal and Ubiquitous Computing*, vol. 10, no. 5, pp. 285–299, July 2006.

[4] T. Pylvänäinen, "Accelerometer Based Gesture Recognition Using Continuous HMMs Pattern Recognition and Image Analysis," vol. 3522 of *Lecture Notes in Computer Science*, chapter 77, pp. 413–430. Berlin, Heidelberg, 2005.

[5] D. H. Wilson and A. Wilson, "Gesture recognition using the xwand," Tech. Rep. CMU-RI-TR-04-57, Robotics Institute, April 2004.

[6] Jiayang Liu, Zhen Wang, Lin Zhong, J. Wickramasuriya, and V. Vasudevan, "uWave: Accelerometer-based personalized gesture recognition and its applications," in *International Conference on Pervasive Computing and Communications*, 2009, pp. 1 –9.

[7] A. Akl and S. Valaee, "Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing," in *Proceedings of the Internation Conference on Acoustics, Speech, and Signal Processing*, 2010.

[8] M. Hoffman, P. Varcholik, and J. J. LaViola, "Breaking the status quo: Improving 3D gesture recognition with spatially convenient input devices," in *Virtual Reality Conference (VR)*, 2010, pp. 59 –66.

[9] Jiahui Wu, Gang Pan, Daqing Zhang, Guande Qi, and Shijian Li, "Gesture recognition with a 3-D accelerometer," 2009, Ubiquitous Intelligence and Computing, pp. 25–38.

[10] G. Lefebvre, S. Berlemont, F. Mamalet, and C. Garcia, "BLSTM-RNN based 3D gesture classification," in *Proceedings of the International Conference on Artificial Neural Networks*, 2013, pp. 381–388.

[11] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *IEEE Transactions on Neural Networks*, , no. 18, pp. 5–6, 2005.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, November 1998.

[13] Y. LeCun, K. Kavukvuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proceedings of the International Symposium on Circuits and Systems (ISCAS'10)*. 2010, IEEE.

[14] S.-J. Cho, E. Choi, W.-C. Bang, J. Yang, J. Sohn, D. Y. Kim, Y.-B. Lee, and S. Kim, "Two-stage recognition of raw acceleration signals for 3-D gesture-understanding cell phones," in *10th International Workshop on Frontiers in Handwriting Recognition*, 2006.

[15] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Proceedings of the Internation Conference on Acoustics, Speech, and Signal Processing*, 2012, pp. 4277–4280.

[16] E. Petit, "Grasp: Moteur de reconnaissance de gestes," 2010, Dépôt logiciel, France Télécom, IDDNFR.001.030023.000.S.P.2010.000.31500.