

ChaLearn Gesture Challenge: Design and First Results

Isabelle Guyon
ChaLearn, California
isabelle@clopinet.com

Vassilis Athitsos and Pat Jangyodsuk
University of Texas at Arlington, Texas
{athitsos, pat.jangyodsuk}@uta.edu

Ben Hamner
Kaggle, California
ben@benhamner.com

Hugo Jair Escalante
INAOE, Mexico
hugojaire@inaoep.mx

Abstract

We organized a challenge on gesture recognition: <http://gesture.chalearn.org>. We made available a large database of 50,000 hand and arm gestures videorecorded with a KinectTM camera providing both RGB and depth images. We used the Kaggle platform to automate submissions and entry evaluation. The focus of the challenge is on “one-shot-learning”, which means training gesture classifiers from a single video clip example of each gesture. The data are split into subtasks, each using a small vocabulary of 8 to 12 gestures, related to a particular application domain: hand signals used by divers, finger codes to represent numerals, signals used by referees, marching signals to guide vehicles or aircrafts, etc. We limited the problem to single users for each task and to the recognition of short sequences of gestures punctuated by returning the hands to a resting position. This situation is encountered in computer interface applications, including robotics, education, and gaming. The challenge setting fosters progress in transfer learning by providing for training a large number of subtasks related to, but different from the tasks on which the competitors are tested.

1. Introduction

Challenges have been fostering progress in computer vision in recent years. Some of the notable challenges include the Visual Object Classes Challenges (VOC), sponsored by Pascal2, which have contributed to push the state-of-the-art. The initial effort concentrating on classifying images is now moving towards the analysis of video data. Several contests have been organized in activity recognition with applications in video surveillance (e.g. the CVPR 2011 DARPA sponsored VIRAT challenge). Our new challenge focuses on recognizing gestures from video data recorded with a Microsoft KinectTM camera, which provides both RGB im-

ages and a depth images obtained with an infrared sensor (Figure1). KinectTM, which has won a Guinness World Record of *fastest selling consumer device*, has revolutionized computer vision in the past few months by providing an affordable 3D camera. The applications, initially driven by the game industry, are rapidly diversifying and include video surveillance, computer interfaces, robot vision and control, and education.

We are organizing two rounds of the challenge in conjunction with the CVPR conference (Providence, Rhode Island, USA, June 2012) and the ICPR conference (Tsukuba, Japan, November 2012). The results of the first round, which just ended, are reported and discussed in this paper.

2. Problem setting and data

We are portraying a single user in front of a fixed camera, interacting with a computer by performing gestures to play a game, remotely control appliances or robots, or learn to perform gestures from an educational software. We have collected a large dataset of gestures using the Microsoft Software Development Kit (SKD) interfaced to Matlab, which includes:

- 50,000 gestures recorded with the KinectTM camera, including RGB and depth videos,
- with image sizes 240 x 320 pixels,
- at 10 frames per second,
- recorded by 20 different users,
- grouped in 500 batches of 100 gestures,
- each batch including 47 sequences of 1 to 5 gestures drawn from various small gesture vocabularies of 8 to 12 gestures,
- from over 30 different gesture vocabularies.

The data are available from <http://gesture.chalearn.org/data> in 2 formats: A lossy compressed AVI format (5 GB) and a quasi-lossless AVI format (30 GB). It presents various features of interest (Table 1).

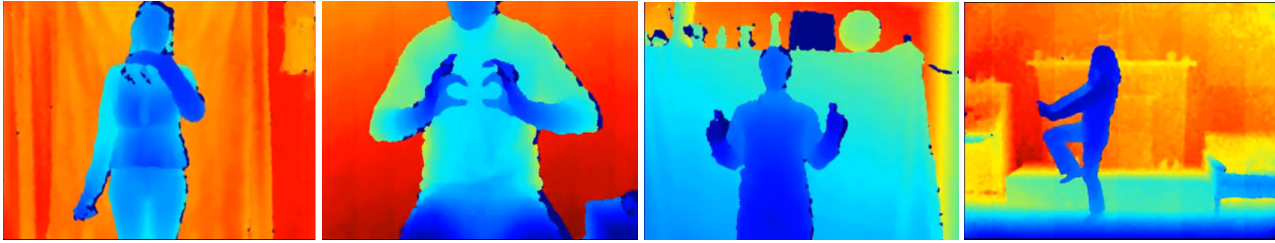


Figure 1. Color rendering of depth images from the gesture challenge database were recorded with a KinectTM camera.

Easy aspects	Challenging aspects
Fixed camera Availability of depth data Within a batch: single user, homogeneous recording conditions, small vocabulary Gestures separated by returning to a resting position Gestures performed mostly by arms and hands Camera framing upper body (some exceptions)	Within a batch: only one labeled example of each gesture Skeleton tracking data not provided Between batches: variations in background, clothing, skin color, lighting, temperature, resolution Some errors or omissions in performing gestures Some users are less skilled than others Some parts of the body may be occluded

Table 1. Easy and challenging aspects of the data.

To get a sufficient spacial resolution, we framed only the upper body. Although Microsoft provided a full body skeleton tracker with the SDK used for data collection, it could not handle partial body occlusion and was not usable for our data. Rather, we provided manual annotations:

- all temporal segmentation for the devel01-20 batches into individual gestures;
- the position of the head, shoulders, elbows and hands for 400 frames sampled from the devel01-20 batches.

We also provided Matlab code to browse though the data and process it to create a sample submission. The data can also be viewed with most video viewers.

Goal of the challenge: One-shot-learning

The data are organized in batches: development batches devel01-480, validation batches valid01-20, and final evaluation batches final01-20. For the develXX batches, we provided all the labels. To evaluate the performances on “one-shot-learning” tasks, the validXX and finalXX batches, were provided with labels only for **one example of each gesture class** in each batch (training examples). The goal is to automatically predict the gesture labels for the remaining unlabeled gesture sequences (test examples).

Each batch includes 100 recorded gestures grouped in sequences of 1 to 5 gestures performed by the same user. The gestures are drawn from a small vocabulary of 8 to 12 unique gestures, which we call a “lexicon”. For instance a gesture vocabulary may consist of the signs to referee volleyball games or the signs to represent small animals in the sign language for the deaf. We selected lexicons from

nine categories corresponding to various settings or application domains; they include (1) body language gestures (like scratching your head, crossing your arms), (2) gesticulations performed to accompany speech, (3) illustrators (like Italian gestures), (4) emblems (like Indian Mudras), (5) signs (from sign languages for the deaf), (6) signals (like referee signals, diving signals, or marshalling signals to guide machinery or vehicle), (7) actions (like drinking or writing), (8) pantomimes (gestures made to mimic actions), and (9) dance postures. During the challenge, we do not disclose the identity of the lexicons and of the users. They will be revealed (after user anonymization) at the end of the challenge.

3. Protocol and evaluation

The challenge (round 1) consisted of two main components: a development phase (Dec 7, 2011 to Apr 6, 2012) and a final evaluation phase (Apr 7 to Apr 10, 2012):

During the development phase (lasting approximately 4 months), the participants were asked to build a learning system capable of learning from a single training example a gesture classification problem. To that end, they received the development data to train and self-evaluate their systems. To monitor their progress they could use the validation data for which the test labels were withheld. The prediction results on validation data could be submitted online to get immediate feed-back. A real-time leaderboard showed to the participants their current standing based on their validation set predictions.

During the final evaluation phase (lasting four days), the participants performed similar tasks as those of the valida-

tion data on new final evaluation data revealed at the end of the development phase. The participants had only a few days to train their systems and upload their predictions. Prior to the end of the development phase, and BEFORE the final evaluation data was revealed, a software vault was made available so the participants could upload executable code for their best learning system, which they then used to train their models and make predictions on the final evaluation test data. This allowed the competition organizers to check their results and ensure the fairness of the competition. Note that participation was NOT conditioned on submitting code or disclosing methods. If any of the top ranking participants had opted not to submit their learning system for verification, an alternative verification method would have been offered. However, all top ranking participants submitted their code in round 1.

The submission and evaluation of the challenge entries was via the Kaggle platform: <http://www.kaggle.com/c/GestureChallenge>. The official rules are provided on the website of the challenge.

Metric of evaluation

For each unlabeled video, the participants were instructed to provide an ordered list of labels R corresponding to the recognized gestures. We compared this list to the corresponding list of truth labels T *i.e.* the prescribed list of gestures that the user had to play during data collection. We computed the Levenshtein distance $L(R, T)$, that is the minimum number of edit operations (substitution, insertion, or deletion) that one has to perform to go from R to T (or vice versa). The Levenshtein distance is also known as “edit distance”. For example: $L([124], [32]) = 2$.

The overall score is the sum of the Levenshtein distances for all the lines of the result file compared to the corresponding lines in the truth value file, divided by the total number of gestures in the truth value file. This score is analogous to an error rate. For simplicity, in what follows, we call it error rate. However, it can exceed one.

Public score means the score that appears on the leaderboard during the development period and is based on the validation data. Private score means the score that was computed on the final evaluation data released at the end of the development period, which was not revealed until the challenge was over. The private score was used to rank the participants and determine the prizes.

4. Results

The first round of the challenge attracted 153 players regrouped in 54 teams who made a total of 573 entries. This is a good level of participation for a computer vision challenge requiring very specialized skills. For comparison, the popular Pascal2 VOC challenges attracted in 2011 between and 1 and 19 participants.

Team	Public score (validation set)	Private score (final set)
alfnie	0.1426	0.0996
Pennect	0.1797	0.1652
OneMillionMonkeys	0.2697	0.1685
immortals	0.2543	0.1846
Zonga	0.2714	0.2303
Balazs Godeny	0.2637	0.2314
SkyNet	0.2825	0.2330
xiaozhuwudi	0.2930	0.2564

Table 2. Results of round 1.

The results of the top 8 ranking participants were checked by the organizers by reproducing their results using the code that they provided BEFORE they had access to the final evaluation data. All of them passed successfully the verification process. These results are shown in Table 2.

Statistics on the results

We show in Figure 2 the distribution of results. Figure 2a represents the histogram of performance of all the entries made on validation data. The distribution is widely spread, which indicates that the tasks of the challenge separated well the participants: they were challenging enough to require some effort to achieve good results, yet they were doable. The two top ranking participants and several others made no entry on the validation set until the very last days of the challenge, possibly in an attempt to avoid that other competitors would put additional effort to try to beat them. Their entries ended up cutting the error rate by almost a factor of two, narrowing down considerably the gap to human performance (which is under 2% error).

Figure 2b shows the correlation between the validation set error and the final evaluation data error. We purposely made the final evaluation data slightly easier so the entrants would not eventually feel frustrated. Hence we expected that the entry points would be under the diagonal. This is confirmed for the top ranking participants. However, some lower ranking participants have entry points over the diagonal, which indicates a possible overfitting of the validation data.

Fact sheets

We asked the participants to fill out a survey about the methods employed. Twenty groups replied, among the 22 top ranking participants. We briefly summarize the results. Detailed results may be found at <http://www.causality.inf.ethz.ch/Gesture/FactSheetSummaryRound1.pdf>.

Preprocessing and data representation: Most of the participants employed image enhancement and filtering techniques, in majority denoising or outlier removal and

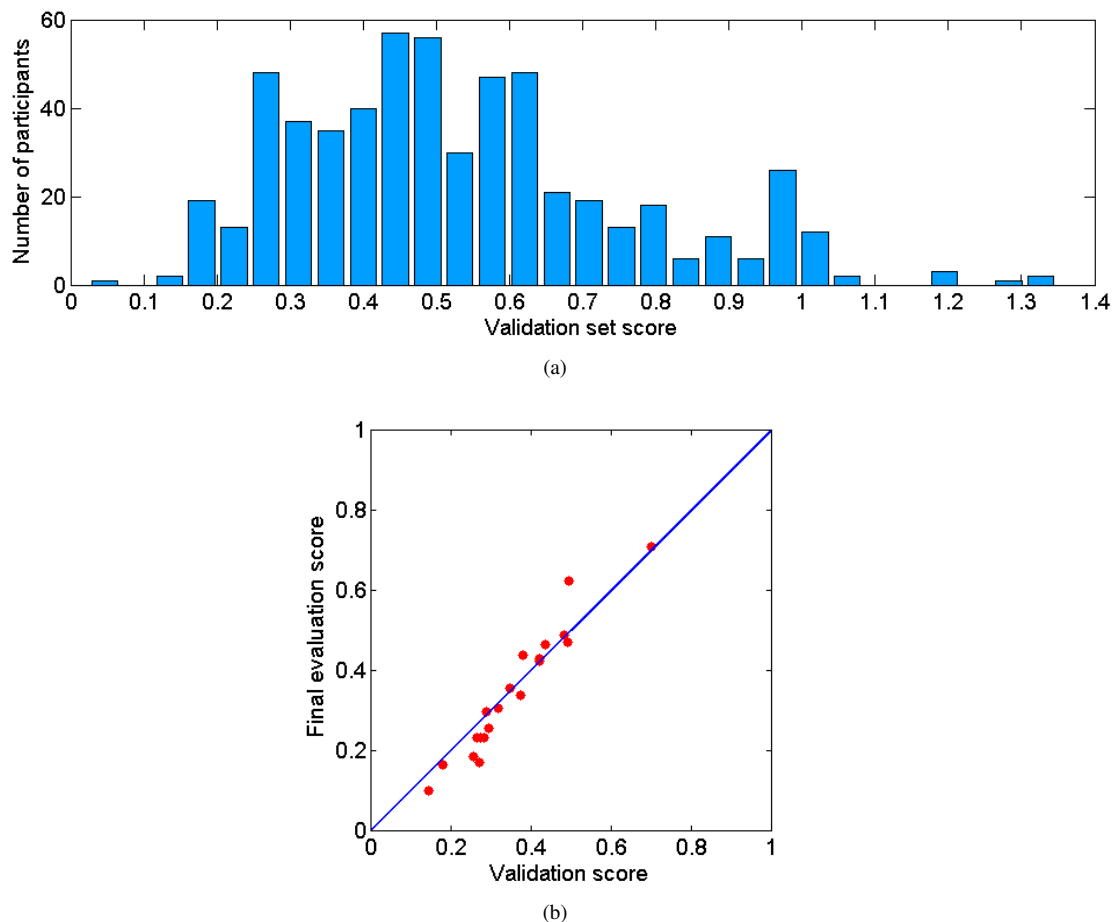


Figure 2. Result distribution. (a) Histogram of performance on validation data. (b) Correlation between validation results and final evaluation results.

background removal. Some reduced the image resolution for faster processing. Notably, some of the top ranking participants did not do any such low level preprocessing. The majority of the top ranking participants used HOG/HOF features and/or ad-hoc hand crafted features, edge/corner detectors or SIFT/STIP features. Very few resorted to using body parts or trained features. Most participants used the depth image only, but about one third used both RGB and depth images. Interestingly, the second place winner used the RGB image only. About one third of the participants did no dimensionality reduction at all (including the second and third place winners) and one third resorted to feature selection. Other popular techniques included linear transforms (such as PCA, see *e.g.* [3]) and clustering.

Recognition: For temporal segmentation, most participants used candidate cuts based on similarities with the resting position or on amount of motion. All the top ranking participants used recognition-based segmentation techniques (in which recognition and segmentation are integrated). As gesture representation, all highest ranking par-

ticipants used a variable length sequence of feature vectors (sometimes in combination with other representations). Consistent with the data representation, the highest ranking participants used Hidden Markov Models (HMM), Conditional Random Fields (CRF) or other similar graphical models, see *e.g.* [4]. Some highly ranked, but not top ranking, participants used a bag-of-word representation or image templates, including motion energy or motion history representations [1]. The corresponding classifiers were usually nearest neighbors (using as metric the Euclidean distance or correlation). One participant used a linear SVM, see *e.g.* [3]. Many participants made use of the development data to either learn features or gesture representations in the spirit of “transfer learning”, see *e.g.* [8]. However, none of the 4 top ranking ones resorted to transfer learning.

Implementation: Most participants claimed that the algorithmic complexity of their methods was linear in image size, number of frames per video, and number of training examples. The median execution time on the 20 batches of the final evaluation set was 2.5 hours, which is very rea-

sonable and close to real time performance. However, there were a few outliers and it took up to 50 hours for the slowest code. Most participants, including the three top ranking ones, claimed that their methods were simple and easy to implement. Several top ranking participants claimed that they had made novel contributions. Most participants developed their own code and may be willing to share it when it matures. Matlab was the most popular platform and was used by the 2 top ranking participants. “C” and derived languages were often used and sometimes in combination with Matlab, in particular making use of OpenCV libraries [2]. A few participants used Java. The majority of the participants developed under the Windows operating system (65%), the rest divided about evenly between MacOS and Linux. Two third of the systems required less than 2GB of memory and all less than 8 GB. Parallelism was usually exploited via the use of multi-processor machines.

Development effort: Two-third of the survey respondents spent more than 2 man-weeks of development effort. This is a lot of effort in comparison to the computer time generally estimated to a few hours of a few days. Half of the participants estimated that the challenge duration was sufficient to achieve their goals. One third declared that they will spend more time during the second round.

Algorithms employed

We do not have details on the algorithms employed by most of the participants. Understandably, since there is still a second round, many preferred not to publish at an early stage. For the unpublished methods, we summarize the descriptions provided in the fact sheets. Interestingly, all top ranking methods are based on techniques making no explicit detection and tracking of humans or individual body parts.

The winning team (alfnie) used a novel technique called “Motion Signature analyses”, inspired by the neural mechanisms underlying information processing in the visual system. This is an unpublished method using a sliding window to perform simultaneously recognition and temporal segmentation, based solely on depth images.

The second best ranked participants (team Pennect) did not publish their method yet. From the fact sheets we only know that it is an HMM-style method using HOG/HOF features with a temporal segmentation based on candidate cuts. Only RGB images were used. The methods of the two best ranking participants are quite fast. They claim a linear complexity in image size, number of frames, and number of training examples.

The third best ranked team (OneMillionMonkeys) did not publish either, but they provided a high level description indicating that the system uses a HMM in which a state is created for each frame of the gesture exemplars. The state machine includes skips and self-loops to allow for variation

in the speed of the gesture execution. The most likely sequence of gestures is determined by a Viterbi search. Comparisons between frames are based on the edges detected in each frame. Edges are associated with several attributes including the X/Y coordinates, their orientation, their sharpness, their depth and location in an area of change. In matching one frame against another, they find the nearest neighbor in the second frame for every edge point in the first frame, and calculate the joint probability of all the nearest neighbors using a simple Gaussian model. The system works exclusively from the depth images. The system is one of the slowest proposed. Its processing speed is linear in number of training examples but quadratic in image size and number of frames per video.

The fourth best ranked entrant Manavender Malgireddy (immortals) published in these proceedings a paper called “Detecting and Localizing Activities/Gestures in Video Sequences” [7]. They detect and localize activities from HOG/HOF features in unconstrained real-life video sequences, a more complex problem than that of the challenge. To obtain real-life data, they used video clips from the Human Motion Database (HMDB). The detection and localization paradigm was adapted from the speech recognition community, where a keyword model is used for detecting key phrases in speech. The method learns models for activities-of-interest and creates a network of these models to detect keywords. According to the paper, the approach out-performed all the current state-of-the-art classifiers when tested on publicly available datasets such as KTH and HMDB.

The fifth best ranked entrant Yui Man Lui (Zonga) published his results in these proceedings [5] a paper entitled “A Least Squares Regression Framework on Manifolds and its Application to Gesture Recognition” [5]. The method is derived from his CVPR 2010 paper [6]. It characterizes action videos as elements on a product manifold through High-Order Singular Value Decomposition (HOSVD) and employs geodesic distance for classification. HOSVD on a query gesture video factors it into three sub-regression models on different manifolds. Least-square RBF regression on these manifolds provide a distance measure and a nearest neighbor result is returned. For challenge data, the method gets good results using only depth images. The authors mention that they “synthesize training examples” from the single original training instance in the challenge dataset. The method also produces good (though not the best) results on other public datasets. The method requires temporal segmentation of gestures to be performed as pre-processing, which may limit performance.

The sixth best ranked entrant (Balazs Godeny) calls his unpublished method “Thumbnail Dynamic Time Warping” (DTW). He provided a summary indicating that DTW was performed on very heavily subsampled depth images:

16x12 pixel thumbnails from each frame. DTW distance between vocabulary videos and test video segments were calculated. For frame-by-frame distance he took the squared pixel differences summed for all pixels, while allowing a 1-pixel offset of the two frames in both dimensions. This classifier was good enough for those cases where there were large scale differences between vocabulary videos. For small scale differences he took snapshots from the video where the image looked stable (not much change compared to the next frame). He calculated the histogram of oriented gradients (HOG) from the edges on such snapshots and compared these snapshot HOGs of the test segments with those of the vocabulary videos. Temporal segmentation was done based on the minimum difference of a frame from any starting or ending frame of vocabulary videos, plus the speed of change at a given frame.

The seventh ranked team (SkyNet) describe their unpublished method as extracting features from depth videos (RGB videos were not used), building a classifier, time-matching frames with a variation of Dynamic Time Warping (DTW), and then using the classifier to score matched frame pairs.

The eighth ranked team (xiazhuwudi) published a paper in these proceedings entitled “One Shot Learning Gesture Recognition from RGBD Images” [9]. The authors introduce the Extended-MHI, i.e., a variation of the well known Motion History Image (MHI) feature extraction approach [1], and provide some experimental results by contrasting it against MHI and other two feature extraction methods (GEI and INV). The two modalities (RGB and depth) are fused with multi-view spectral embedding. Nearest neighbor classification is performed with a correlation coefficient as similarity measure.

5. Conclusion

The final evaluation performance for the first round of the gesture challenge is around 10% error, still far from human performance, which is below 2% error. However, the progress made during round 1, starting at a baseline performance of 60% error indicates that the objective of attaining or surpassing human performance could possibly be reached in the second round. There is room for improvement particularly because the top two ranking participants used only one modality (the first one depth only and the second one RGB only) and because many gestures are recognizable only when details of hand posture are used, yet none of the methods disclosed made use of such information. The most efficient techniques so far have used sequences of features processed by graphical models of the HMM/CRF family, similar to techniques used in speech recognition. No use of skeleton extraction or body part detection was made. Rather, orientation and ad hoc features were extracted. It is possible that progress will also be made in feature ex-

traction by making better use of the development data for transfer learning.

Acknowledgements

This challenge is organized by ChaLearn <http://chalearn.org> whose directors are gratefully acknowledged. The submission website is hosted by Kaggle <http://kaggle.com> whom we thank for wonderful support. This effort was initiated by the DARPA Deep Learning program and is supported by the US National Science Foundation (NSF) under grants ECCS 1128436 and ECCS 1128296, and the EU Pascal2 network of excellence. Our sponsors include Microsoft (Kinect for Xbox 360) and Texas Instrument who donated prizes. We are very grateful to Alex Kipman and Laura Massey at Microsoft and to Branislav Kisanin at Texas Instrument who made this possible. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies and sponsors. We also thank the committee members and participants of the CVPR 2011 gesture recognition workshop and the Pascal2 reviewers who made valuable suggestions.

References

- [1] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(3):257–267, Mar. 2001.
- [2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
- [3] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. New York: Springer-Verlag, 2001.
- [4] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [5] Y. M. Lui. A least squares regression framework on manifolds and its application to gesture recognition. In *CVPR2012 workshop on gesture recognition*, 2012.
- [6] Y. M. Lui, J. R. Beveridge, and M. Kirby. Action classification on product manifolds. In *CVPR2010*, pages 833–839, 2010.
- [7] M. Malgouyres. Detecting and localizing activities/gestures in video sequences. In *CVPR2012 workshop on gesture recognition*, 2012.
- [8] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.
- [9] D. Wu, F. Zhu, and L. Shao. One shot learning gesture recognition from rgb-d images. In *CVPR2012 workshop on gesture recognition*, 2012.