Methodology for Learning Multimodal Instructions in the Context of Human-Robot Interaction Using Machine Learning

Saith Rodríguez[1], Carlos A. Quintero[1](✉), Andrea K. Pérez[1], Eyberth Rojas[1], Oswaldo Peña[1], and Fernando De La Rosa[2]

[1] Universidad Santo Tomás, Bogotá D.C., Colombia
{saithrodriguez,carlosquinterop,andrea.perez, eyberthrojas,oswaldopena}@usantotomas.edu.co
[2] Universidad de los Andes, Bogotá D.C., Colombia
fde@uniandes.edu.co

Abstract. This work shows the design, implementation and evaluation of a human-robot interaction system where a robot is capable of learning multimodal instructions through gestures and voice issued by a human user. The learning procedure can be performed in two ways: an instruction learning phase, where the human aims at teaching one instruction to the robot by performing several repetitions and an instruction receiving phase where the robot reacts to the instructions given by the human and possibly asks for feedback from the user to strengthen the instruction's model.

1 Introduction

Every day more robots are immersed in daily-life tasks and disciplines such as home tasks, medical-care and industries, among others. Moreover, every day this human-robot interaction becomes more personal and accessible to everyone, from children to elders, from engineers to farmers. Therefore, it is paramount for robots to interact with people using types of communication familiar to anyone with whom it would interact. In other words, robots should understand and use natural and intuitive interactions with people.

To succeed in the aforementioned task, it is necessary to tackle the problem from multiple areas of expertise such as human-robot interaction, navigation, mapping, mechanics, object manipulation, among others. This article is focused in human-robot interaction, which includes sensor integration to capture data, artificial intelligence systems development, and implementation of mechanisms to interact with humans in a natural, reliable and simple manner. Specifically, we deal with the problem of one robot that learns how to interact with a human in a natural way by using inputs from varied nature, i.e., audio and video. The instruction learning process is achieved by several repetitions of the instruction

C. Brito-Loeza and A. Espinosa-Romero (Eds.): ISICS 2018, CCIS 820, pp. 41–55, 2018.
https://doi.org/10.1007/978-3-319-76261-6_4

performed by the human. Afterwards, the human can execute the instruction and the robot will understand and will react to it. The proposed model uses the input data to build two classifiers; one for the audio data and one for the video stream. Both classifiers consists of ensembles of one-class SVMs. The final decision of which instruction the input data belongs to, is taken based on each sample confidence.

This article is organized as follows, the first section presents the state-of-art in instructions learning in human-robot interaction. The following section presents the proposed methodology, first we show the system's overall architecture for learning multi-modal instructions in the human-robot interaction context; then, every module is explained in detail. The next section shows the experimental setup and the results obtained. Finally, the last section summarizes the conclusions of our work.

2 Related Work

Recently, there have been many research and application works around the problem of Human-Robot Interaction (HRI), especially, in proposals that aim at recognizing instructions issued by a human to a robot that makes the interaction as natural as possible to the human. The work in [1] show a proposal that uses gesture classification to orient a telepresence robot to direct its attention to specific targets using a Kinect. The gesture recognition is achieved by processing visual information that allows the system to track the human head as well as audio information from the robot's microphone to locate the source of sound. In their work, they use Localist Attractor Network for the gesture recognition task and Short-Term Memory to obtain the attention direction.

Other works also propose the application of machine learning techniques in the context of HRI to recognize only gestures by using the information of the human body motion by tracking its joints using a Kinect [2]. The authors propose three phases. In the initial phase, gesture classification is performed on a set of known human movements using an edited video stream. Each gesture consists of a set of frames where the Kinect detects the position of the 20 human joints for each frame. The second phase attempts to detect unknown movements by extending the techniques proposed in the first phase, detecting movements in an unedited video stream which also captures random gestures. In the third phase, a set of rules are applied in order to filter gestures that were classified into the known vocabulary but were not really intended to be instructions for the robot. Using this methodology, the authors build a proposal to classify gestures over unedited video streams.

In [3,4], a proposal is presented to create an automatic gesture classifier capable of discriminate between known gestures and signals with no meaning. In this work, a Kinect is also used to acquire the input data and a set of 4 gestures with 300 samples each is considered. Additionally, a set of random gestures is also acquired and included in the classifier's accuracy calculation during the validation process. In [5], an audio classification system is proposed using SVMs and RBF-NN. In this work, the authors propose a feature extraction phase for the audio data using Mel-frequency Cepstral Coefficients (MFCCs). Such coefficients have shown to be a key step in the construction of classification systems that use audio inputs [6] since they accurately represent the acoustic phenomena by highlighting low frequencies.

In [7] the authors introduce a classification system with 5 different labels by analyzing visual features. This work is the base for [8,9], where the instructions given by the human are captured, not only using visual information, but also audio features. These works propose a first attempt to combine both data sources (audio and visual) to classify the 5 labels (news, advertisement, sport, serial and movies). Like in the previous works, the authors also use the MFCCs to extract audio features. For the gesture data they propose the use of color histograms for video segmentation. The information obtained from both sources is used to construct a different classifier, i.e., one for audio and one for video, using Support Vector Machines. By weighted sum of each individual classifier, the output is combined into one single answer. Many authors have proposed the use of individual classifiers, mostly for audio and video classification and the general idea is to create several classifiers and combine them in different ways to improve the general system's performance [10].

One of the most interesting applications of HRI is that of collaboration between humans and robots to execute specific tasks. However, several factors need to be taken into account to successfully accomplish such interaction. Specifically, the robot and humans goals may not be completely aligned when long-term interactions are established. The authors in [11] claim that online learning is of paramount importance to alleviate this situation. However, it is not yet widely used in most human-robot interactions for two main reasons: On one hand, typical learning algorithms are incapable of learning in time frames required to interact effectively with humans. On the other hand, there is a random exploration property present in most online learning algorithms that may cause inappropriate results when a robot interacts with humans. In their work, they propose the use of effective communication strategies to accelerate the learning process during the interaction.

3 Proposed Methodology

In the following sections we show the proposed methodology to design, develop and test a HRI system capable of learning instructions issued by a human using multimodal inputs, i.e., audio and gesture signals.

3.1 Solution Architecture

Figure 1 shows our proposed architecture. The process starts by using the Kinect to capture both the audio signal from the human voice and the gestures from the video stream. The overall design consist of 5 main modules, namely Motion

Detection, Voice Activity Detection, Feature Extraction (skeleton features), Feature Extraction (MFCC) and Multimodal Classifier. The first two modules allow the detection and identification of possible instructions given by the human. These modules allow our system to discriminate between small movements and audio that may be considered noise, from actual instructions issued by the human. Afterwards, a feature extraction process is performed over the filtered data in order to extract and keep the important information required for the classification task. Finally, a multimodal classifier provides an answer that is transmitted to the user through an action performed by the robot. This validation is made in order to improve the learning model while the interaction runs.

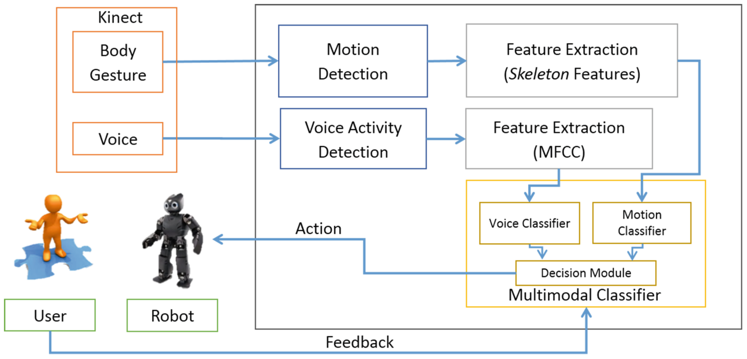


Fig. 1. Overall system architecture

The following sections show a detailed description of each module within the overall architecture.

3.2 Inputs Description

One of the main important contributions within our proposal is the input's multimodal nature, i.e., it comes from two different sources for the same instruction. One of them is the video stream to detect gestures and the other one is audio data to detect a voice instruction. For this purpose, we have integrated the Microsoft Kinect sensor which provides us with both data sources.

Table 1 shows the sampled frequency and format of each input captured using the Kinect.

3.3 Gestures Detection and Voice Activity

The main target of the gestures and voice activity detection modules is to effectively detect and extract the pieces of information, provided by the Kinect, where

Table 1. Sampling frequency and format of each input captured using the Kinect

Input	Format	Frequency
Body gesture	(x, y, z) datapoints that indicate the spatial coordinate (in meters) of certain human body joints	30 fps
Audio	Data array from the 24 bit ADC	16 kHz

presumably a human has issued an instruction, either spoken or by using a gesture. In both cases, the modules are implemented through a finite state machine especially designed to follow the behavior of each type of input signal: for the audio input, the signal energy is computed and processed, while for the gesture signal a change of coordinates is performed in order to obtain an estimated "amount of movement" from the user in certain periods of time. For the audio signal, if the calculated energy does not exceed a threshold value, it is considered noise. Similarly, if the calculated "amount of movement" is not high enough, it is not considered an actual gesture.

3.4 Feature Extraction for Gesture Instructions

This module is in charge of extracting useful information from the data provided by the Kinect in the form of skeleton joint coordinates. The main idea is to build a set of fixed-size distinctive features that may be used to classify between different body gestures issued by different users.

The first step consists of performing a sub-sampling procedure on the data in order to match the number of samples between gestures, since it may vary according to the user and the specific body movement. The general procedure is shown in Fig. 2 for two different body gestures, one with n_1 samples and another with $n_2 < n_1$ samples. Both gestures are sub-sampled to match a desired number of m samples.

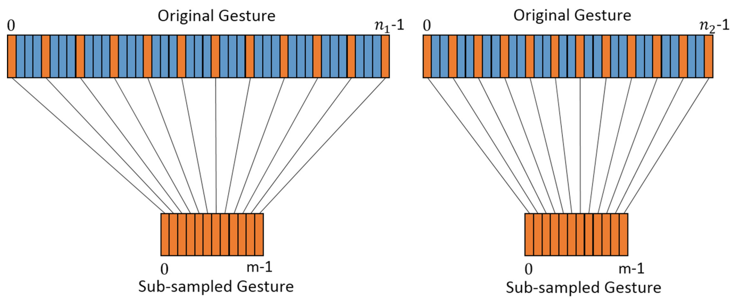


Fig. 2. Sub-sampling procedure applied to body gesture data for two gestures with a different amount of original sample numbers.

Initially, the new sampling frequency is computed using the original number of samples and the desired number of samples as shown in Eq. (1).

$$f_m = \frac{n}{m} \tag{1}$$

where
f_m is the new sampling frequency
n is the number of samples from the original gestsure
m is the desired number of samples for the sub-sampled gesture

The new dataset of size m is obtained by extracting the original samples as shown in Eq. (2) for $i = 0, 1, ..., m - 1$.

$$\hat{S}_i = S_{i * f_m} \tag{2}$$

where:
$\hat{S}_i$ is the i-th sub-sampled gesture sample
S_i is i-th original gesture sample
f_m is the new sampling frequency

This sub-sampling process guarantees that every gesture performed by the user is represented with m samples that are in turn composed of 9 skeleton (x, y, z) coordinate joints. However, these gesture representations are static coordinates that do not represent the motion of the gesture itself. To alleviate this situation we have proposed to create vectors that represent the gesture motion among time using the available joints coordinates. The procedure consists on calculating origin-centered vectors for each joint where the vector tail is located in the current sample and its head is in the next corresponding joint sample. A diagram of the general procedure is shown in Fig. 3 for two consecutive samples.

The amount of features for the body gesture input corresponds to the $(m - 1)$ joint motion vectors times the 9 skeleton joints times the 3 (x, y, z) spatial coordinates for a total of $27(m - 1)$ features.

3.5 Feature Extraction for Voice Instructions

This module will receive audio data captured by the Kinect's microphone. The idea is to obtain a set of features from such data stream useful to classify between instructions given orally by a human.

For this purpose, we used the approach followed by [12], where the signal is split in smaller audio windows where the coefficients of the Discrete Fourier Transform (DFT) are calculated for each one, in order to estimate the power spectral density. This density is then filtered using a set of overlapped triangular filters to highlight the importance of low frequencies imitating the behavior of the human ear. Afterwards, the signal is scaled to a logarithmic scale and the Discrete Cosine Transform (DCT) is applied in order to eliminate the dependency of adjacent bands. Finally, the signal is split into several windows where the MFCCs are calculated for each one and then averaged for all windows. A summary of the whole procedure is shown in Fig. 4.

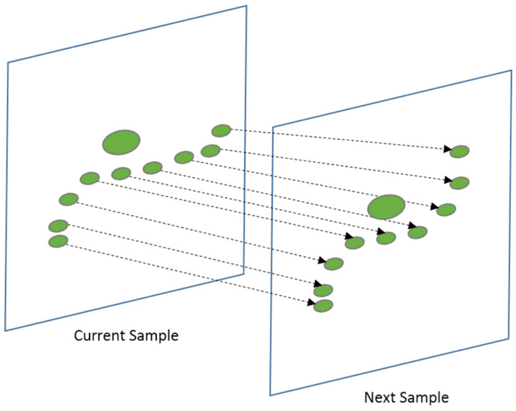


Fig. 3. Transformation from joint coordinates to joint motion vectors using the current and the next samples

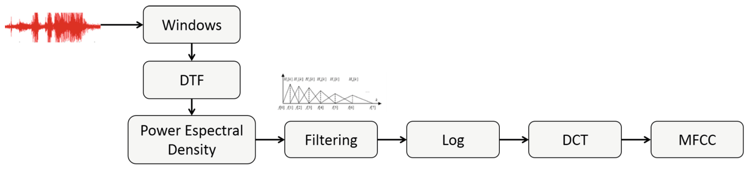


Fig. 4. Procedure for the extraction of MFCCs

3.6 Classifier

The classifier is the module responsible for receiving the processed data from the feature extraction modules and providing an answer of which instruction, if any, has been issued by the human user.

We have proposed the architecture shown in Fig. 5 for the multiclass multimodal classification task. A specific classifier is created based on the nature of the input data, i.e., one classification for the audio input and another for the gesture motion input. A decision output module takes the outputs of both classifiers and provides the general classifier output.

This module is capable of performing two different functions depending on the phase of interaction between the human and the robot, namely the **instruction learning phase** and the **instruction receiving phase**. On the first phase, the human attempts to teach one instruction to the robot by means of a training procedure, while on the latter, the human user issues instructions that the robot attempts to understand and execute. The steps performed by the classifier in each interaction phase are described in Fig. 6.

During the *instruction learning phase*, the human presents to the robot a set of repetitions of the specific gesture and voice command that determines one instruction. The classification module receives such multimodal inputs and uses them to build, through a training process, a model capable of accurately classify such instruction when presented again. In this phase, the classifier

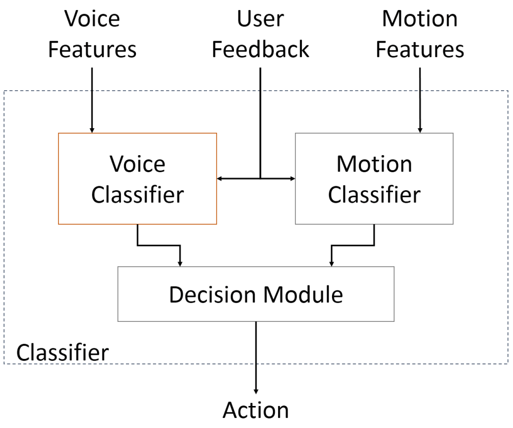


Fig. 5. Proposed architecture for the automatic classifier

performs a training function (Fig. 6(a)). During the *instruction receiving phase*, the human issues an instruction which is processed by the classifier to determine the user's intention. According to its result, the robot provides an action and then receives feedback from the user in order to enlarge the training database which in turn further strengthen the existing classification model. During this phase, the module acts mostly as a classifier (Fig. 6(b)).

Gesture and Voice Classifiers. As shown in Fig. 5, the general classifier is integrated by two independent classifiers; one for the audio data and one for the motion data. Each classifier is built independently using Support Vector Machines (SVMs), as these have shown high success in classification problems for audio and motion data [13]. In Fig. 7 we show our proposed architecture for each separate classifier using a special class of classifiers known as *One-class SVM*.

The One-class SVMs are learning machines capable of discriminate whether a given input sample belongs to one class or not [14]. Similar to other SVM-based algorithms, they rely on the kernel trick, a mathematical concept that allows the algorithms to create non-linear classification boundaries. In this work we use the widely used Gaussian and Polynomial kernels that have shown improved performance for general applications. Unlike binary classifiers, the One-class SVMs do not require training samples from other classes other than the one they are built for in their training process. Using a set of One-class SVMs, i.e., one for each class, as shown in Fig. 7, it is possible to build complete multiclass classifiers using one classifier per instruction. This architecture has the advantage that each new instruction provided by the user is independent from the others and does not require retraining the whole classifier, but only the one in charge of the new class.

The training procedure of the One-class SVMs is performed following a typical methodology for building statistical learning models [15]. The complete

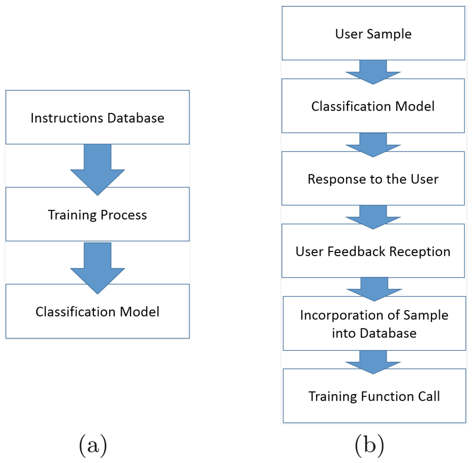


Fig. 6. Functions performed by the classifier (a) During the training function, the database for each instruction is built while the human interacts with the robot and is used in a training process that finally allows the creation of a classification model. (b) During the classification function, a user provides a sample to the robot which passes through the classification model. Depending on the classifier's output, the robot provides a reaction and asks (or not) for feedback to identify the real data label that is finally incorporated into the dataset to retrain the classifier.

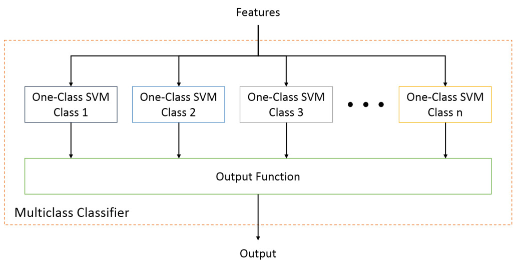


Fig. 7. General architecture for the multiclass classifiers for audio and motion data. The input data is passed through all one-class SVMs and then, the answer is merged into one single output.

dataset is split into two disjoint sets; one for the training process and other for testing the generalization capabilities of the classifier. In this case, the training data for each One-class SVM is made of samples that belong only to one instruction. However, the testing set for each One-class SVM classifier is composed of samples from the own class, but also of data samples from other classes.

In order to obtain classifiers with high generalization capabilities, the values of the kernel parameter and regularization parameter C are chosen such that the empirical generalization error is minimized [15]. For this, a model selection methodology is applied creating a grid search of the discretized parameter space.

In each parameter combination, a model is created using the training set and tested using the testing set. The chosen model is the one that shows the lowest classification error (i.e., the fraction of missclassified samples) on the testing dataset.

Finally, our methodology includes a grid refinement step where we begin with a coarse grid and end with a recursively finer grid. The recursion is achieved by creating new parameter values the half closer than the previous grid size, until a desired generalization error is achieved or a maximum number of iterations is reached.

Classification Confidence. One additional parameter from our proposal is the calculation of a classification confidence value δ for each classifier for an input sample. This value provides the classifier's certainty about its output and is closely related to the concept of classification margin for each input data point. The general idea is that the higher the margin, the higher its classification certainty. Finally, we propose a margin normalization that will allow us to compare the margins between different classifiers for one single input sample.

The confidence value of the j-th classifier when presented the input sample x is as shown in Eq. (3)

$$\delta_j(x) = \frac{m_j(x)}{\hat{m_j}} \tag{3}$$

where $m_j(x)$ is the margin of the input sample x and $\hat{m_j}$ is the average margin of the j-th classifier in the training dataset.

This confidence value may be understood as follows:

- If $\delta_j(x) \approx 1$, it means that the margin is similar to the average margins for the classification set.
- If $\delta_j(x) > 1$, the margin is better than the average, i.e., there is a high certainty on the classifier's output.
- If $\delta_j(x) < 1$, the margin is lower than the average and hence there is a high uncertainty regarding the classified sample.
- If $\delta_j(x) < 0$, the input data point does not belong to the class and its magnitude represents the certainty of not belonging to the class, i.e., the higher the margin's magnitude, the lower uncertainty of the classified sample.

Multiclass Output Function. The j-th classifier outputs two values when presented an input sample x: (1) its classification label $y_j(x)$, whose value is $y_j(x) = 1$ if it labels the input sample as belonging to its class or $y_j(x) = 0$ otherwise and (2) its classification confidence $\delta_j(x)$. The last module within the classifier is the output function that receives the outputs of all One-class SVMs and merge them into one single answer (classification label and confidence). This module implements the functions shown in Eqs. (4) and (5) in the pressence of N classifiers.

$$\hat{y}(x) = \begin{cases} y_j \text{ if } \sum_{i=0}^{N} y_i(x) = 0 \\ y_k \text{ if } \sum_{i=0}^{N} y_i(x) \neq 0 \end{cases} \tag{4}$$

$$\hat{\delta}(x) = \begin{cases} 0 & \text{if } \sum_{i=0}^{N} y_i = 0 \\ \max_{i=1:N} \delta_i(x) & \text{if } \sum_{i=0}^{N} y_i \neq 0 \end{cases} \tag{5}$$

where $j = \arg\min_{i=1:N} \|\delta_i(x)\|$ and $k = \arg\max_{i=1:N} \delta_i(x)$.

The general idea is to assign to the input data point x the label of the class "closer" to the classification boundary, when all the classifiers claim that it does not belong to any class. In the opposite case, where one or more classifiers assign the input sample as belonging to their own class, the assigned label is the one given by the classifier with highest confidence.

It is noteworthy that this architecture is feasible in the context of HRI, specifically in our proposed framework, since each classifier is anticipated to have high bias due to the low amount of data available for training, compared to other applications and the complexity of the problem at hand.

3.7 Decision Module

Both classifiers, the one related to the audio data and the one that processes the gesture actions, finally produce a label and a confidence value $(\hat{y}_a(x), \hat{\delta}_a(x))$ and $(\hat{y}_v(x), \hat{\delta}_v(x)))$ respectively. There is a final module that receives the answer provided by the two classifiers and makes the final decision regarding the class of the input sample. The *decision module* implements the function in Eq. (6) to produce the final classification output $\hat{y}_g(x)$.

$$\hat{y}_g(x) = \begin{cases} \hat{y}_v(x) & \text{if } \hat{y}_v(x) = \hat{y}_a(x) \\ \hat{y}_v(x) & \text{if } \Big((\hat{y}_v(x) \neq \hat{y}_a(x)) \wedge ((\hat{\delta}_v(x) - \hat{\delta}_a(x)) \geq l)\Big) \\ \hat{y}_a(x) & \text{if } \Big((\hat{y}_v(x) \neq \hat{y}_a(x)) \wedge ((\hat{\delta}_v(x) - \hat{\delta}_a(x)) \leq -l)\Big) \\ 0 & \text{if } \Big((\hat{y}_v(x) \neq \hat{y}_a(x)) \wedge (|\hat{\delta}_v(x)) - \hat{\delta}_a(x))| < l)\Big) \end{cases} \tag{6}$$

The decision module output can be understood as follows:

– If both classifiers agree on the class, then such output is the final decision
– If the classifiers disagree on their outputs, and the difference in their confidences is greater than certain threshold value, then the overall output is the one with largest confidence
– If the classifiers disagree on their outputs and the difference in their confidence *do not* exceed certain threshold value, then the overall classifier output is "don't know", meaning that the instruction has not been recognized.

3.8 Human-Robot Interaction

The last step in our methodology is the actual interaction between the robot and the human. The robot will execute an action according to the output given by the classifier module and hence two possibilities arise: (1) The robot may be highly certain of its answer or (2) the robot has high uncertainty about it. In both cases, the system may use the current sample to further strengthen the classifier's model (i.e., by including it into the training process and retraining the one-class SVM). In the first case, the label assigned to the input sample is the one assigned by the robot (since it is highly certain of its answer). However, in the latter, a feedback from the user is required. This feedback is provided in the form of *approved* or *not approved* signals, meaning that the robot's action is the one expected by the human or not, respectively. If the robot's output was not the one expected by the human, the input sample is not included into the training process (since its true label is unknown).

4 Experiments and Results

We have prepared an experimental setup to test our proposed methodology using a Kinect sensor and a DARwIn-OP robot. Figure 8 shows a human user interacting with the robot in 3 possible steps. Initially, the user performs several repetitions of the instruction that needs to be learned. Afterwards, the user shows to the robot the action that needs to be done when such instruction is issued. Finally, the user issues the instruction and waits for the robot's action.

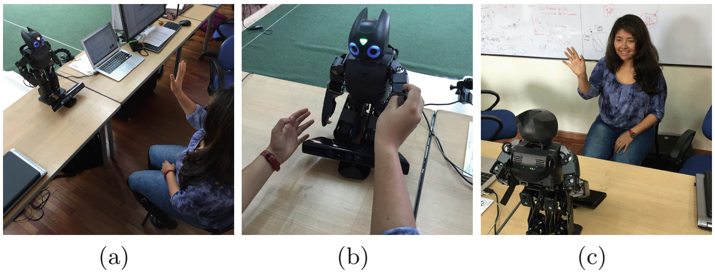


Fig. 8. Experimental setup for the human-robot interaction test: (a) The human performs several repetitions of the instruction to the robot. (b) The human shows the robot the expected answer (c) The human issues the instruction

For the experiment we have asked 4 different users to interact with the robot to teach the system 5 different instructions, namely: **Hello**, **Head**, **Right**, **Left** and **Amen**. Every instruction is composed of a voice command and a body gesture performed by the human. In the *instruction learning phase* the user performs 25 repetitions of each class and these data are used by the robot to build a model for each one. Afterwards, the user starts interacting with the robot 100 times for each instruction, for a total of 500 interactions. During each interaction, the robot has the opportunity of asking for feedback from the user and hence strengthen the classifier's model. In Fig. 9 we show the results of the interaction between the human and the robot for two users when performing the 100 interactions for each class.

In Fig. 9(a) and (b) we present an accumulated count on the number of errors made by the learning system as the interactions pass. In the plots, we show the

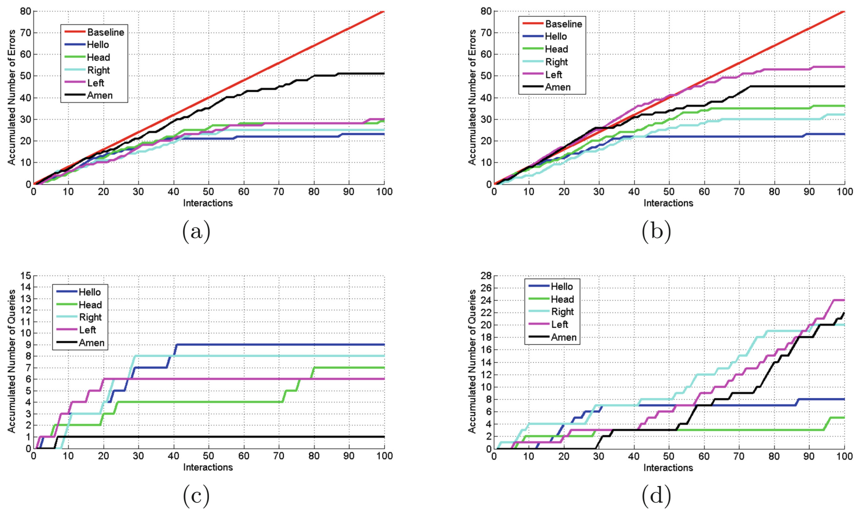
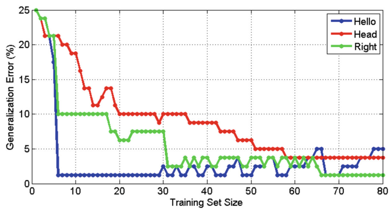


Fig. 9. (a) Accumulated number of errors during the human-robot interaction for user 1. (b) Accumulated number of errors during the human-robot interaction for user 2. (c) Accumulated number of queries during the human-robot interaction for user 1. (d) Accumulated number of queries during the human-robot interaction for user 2.

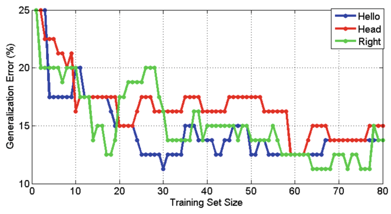
behavior of a Baseline classifier which randomly (uniformly distributed) chooses one of the 5 classes regardless of the input sample. It is noteworthy that as the interaction progresses, for all classes, the accumulated number of errors moves away from the classifier baseline behavior, showing that our system is, in fact, learning throughout the interaction with the user for all classes. We can also see that, for all classes, the number of accumulated errors, for both users, flattened after a certain number of iterations, meaning that the number of data samples included in the training process during the interaction is in fact improving each classifier model.

In Fig. 9(c) and (d), we also show the accumulated number of queries made by the robot during its interaction with the human. Recall from Sect. 3.8 that the robot may ask for feedback to the human only when its output confidence is not high enough. The results show that, for most classes, the robot requests for queries mostly at the beginning of the interaction and that these queries tend to decrease during the interaction. Note that this is an expected result since the models for each One-class SVM become more accurate when new samples are included into the training process and hence the confidence values of each classifier tend to increase. Nevertheless, for some classes, of certain users, it is possible that more queries are required before their confidence is high enough to avoid asking for user feedback.

Finally, we show in Fig. 10 an approximation to the generalization error for three out of the five classes for both classifiers, motion and voice, separately, as the training set grows.



(a)



(b)

Fig. 10. (a) Generalization error for the classes “Hello”, “Head” and “Right” for the gesture classifier. (b) Generalization error for the classes “Hello”, “Head” and “Right” for the audio classifier.

For both cases, it is possible to note that the generalization error decreases as the training dataset size increases. However, the gesture classifier shows lower generalization error than the audio-based classifier.

5 Conclusions

We have proposed an instruction learning system capable of using multimodal inputs in the context of human-robot interaction that makes it possible for a human with no knowledge or experience in programming, robotics and/or engineering to control or receive help from a robot in a natural way.

Our proposed architecture for the learning system is based on the ensemble of several One-class SVMs to create a multiclass classifier, one for each input mode. This architecture allows the complete system to improve its model every time a new data sample is available without retraining the complete multiclass model, but only the one-class classifier corresponding to the class of the input data point. We believe this is an important contribution since it allows the incorporation of the training process as the interaction with the user progresses in a moderate computational time.

Finally, we have proposed a possible feedback query from the robot, when the certainty of its output is not high enough by computing an overall confidence value of the learning and classifying system. This confidence value is related to a normalized value of the classification margin for each data sample, that allows the multiclass classifier to decide the output label for the input data.

References

1. Tee, K.P., Yan, R., Chu, Y.: Gesture-based attention direction for a telepresence robot: design and experimental study. In: Proceedings 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE (2014)
2. Bhattacharya, S., Czejdo, B., Perez, N.: Gesture classification with machine learning using kinect sensor data. In: Third International Conference on Emerging Applications of Information Technology (EAIT) (2012)

3. Huang, J., Lee, C.W., Ma, J.: Gesture recognition and classification using the microsoft kinect. In: 5th International Conference on Automation, Robotics and Applications, ICARA 2011 (2009)
4. Zhang, H., Du, W.X., Li, H.: Kinect Gesture Recognition for Interactive System (2009)
5. Dhanalakshmi, P., Palanivel, S., Ramalingam, V.: Classification of audio signals using SVM and RBFNN. Expert Syst. Appl. **36**(3), 6069–6075 (2009)
6. Sarchaga, G., Sartori, V., Vignoli, L.: Identificacin automtica de resumen en canciones (2006)
7. Suresh, V., Mohan, C.K., Swamy, R.K., Yegnanarayana, B.: Content-based video classification using support vector machines. In: Pal, N.R., Kasabov, N., Mudi, R.K., Pal, S., Parui, S.K. (eds.) ICONIP 2004. LNCS, vol. 3316, pp. 726–731. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30499-9_111
8. Subashini, K., Palanivel, S., Ramaligam, V.: Audio-video based segmentation and classification using SVM (2012)
9. Subashini, K., Palanivel, S., Ramalingam, V.: Audio-video based classification using SVM and AANN. Int. J. Comput. Appl. **44**(6), 33–39 (2012)
10. Kittler, J., Hatef, M., Duin, R.P.W., Matas, J.: On combining classifiers. IEEE Trans. Pattern Anal. Mach. Intell. **20**(3), 226–239 (1998)
11. Babushkin, V., Oudah, M., Chenlinangjia, T., Alshaer, A., Crandall, J.: Online learning in repeated human-robot interactions. In: Artificial Intelligence for Human-robot Interaction: Papers from the AAAI Fall Symposium, pp. 42–44. AAAI Press (2014)
12. Molau, S., Pitz, M., Schluter, R., Ney, H.: Computing mel-frequency cepstral coefficients on the power spectrum. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 73–76. IEEE Press (2001)
13. Rodriguez, S., Pérez, K., Quintero, C., López, J., Rojas, E., Calderón, J.: Identification of multimodal human-robot interaction using combined kernels. In: Snášel, V., Abraham, A., Krömer, P., Pant, M., Muda, A.K. (eds.) Innovations in Bio-Inspired Computing and Applications. AISC, vol. 424, pp. 263–273. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-28031-8_23
14. Schölkopf, B., Williamson, R.C., Smola, A.J., Shawe-Taylor, J., Platt, J.C.: Support vector method for novelty detection. In: Solla, S.A., Leen, T.K., Müller, K. (eds.) Advances in Neural Information Processing Systems 12, pp. 582–588. MIT Press (2000)
15. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University (2003)