Predicting when to laugh with structured classification

Bilal Piot, Olivier Pietquin, Matthieu Geist

To cite this version:

Bilal Piot, Olivier Pietquin, Matthieu Geist. Predicting when to laugh with structured classification. InterSpeech 2014, Sep 2014, Singapore, Singapore. pp.1786-1790. hal-01104739
ABSTRACT

Today, Embodied Conversational Agents (ECAs) are emerging as natural media to interact with machines. Applications are numerous and ECAs can reduce the technological gap between people by providing user-friendly interfaces. Yet, ECAs are still unable to produce social signals appropriately during their interaction with humans, which tends to make the interaction less instinctive. Especially, very little attention has been paid to the use of laughter in human-avaatar interactions despite the crucial role played by laughter in human-human interaction. In this paper, a method for predicting the most appropriate moment for laughing for an ECA is proposed. Imitation learning via a structured classification algorithm is used in this purpose and is shown to produce a behavior similar to humans' on a practical application: the yes/no game.

Index Terms— Laughter; Imitation Learning; Structured Classification

1. INTRODUCTION

Building efficient and user-friendly human-machine interfaces is a key challenge for the future of computer science, enabling a large public to interact with complex systems and reducing the technological gap between people. In the last decade, Embodied Conversational Agents (ECAs) emerged as such interfaces. Yet, their behaviour is still perceived as quite unnatural to users. One of the reasons of this bad perception is the inability of ECAs to make a proper use of social signals, although there exists some research on this topic [1]. Among these signals, laughter is a prominent feature used by humans during interactions. Yet, very little attention has been paid to enable ECAs with laughter capabilities until recently [2].

Enabling ECAs with laughter capabilities is not only about being able to synthesize audio-visual laughter signals [3, 4]. It is also concerned by an appropriate management of laughter during the interaction. There is thus a need for a laughter-enabled interaction manager, able to decide when to laugh so that it is appropriate in the conversation. This being said, it remains uneasy to define what is an appropriate moment to laugh.

More formally, the task of the laughter-enabled Interaction Manager (IM) is to take decisions about whether to laugh or not. These decisions have to be taken according to the interaction context which can be inferred from laughter, speech and smile detection modules (detecting social signals emitted by the users) implemented in the ECA but also by the task context (for example, if the human is playing a game with the ECA, what is the status of the game). Formally, the IM is thus a module implementing a mapping between contexts (or states noted $s \in S$) and decisions (or actions noted $a \in A$). Let’s call this mapping a policy, noted $\pi(s) = a$. This mapping is quite difficult to learn from real data as the laughs are quite rare and very different from one user to another.

In this paper, we describe the research results for learning such a mapping from data, recorded during some human-human interactions, so as to implement, in the IM, a behavior similar to the one of a human. An imitation learning method is thus adopted. Especially, structured classification is investigated and proven to efficiently learn a behavior similar to human users where the similarity between human and algorithms is measured via a new criterion called Naturalness and defined in Sec. 5. In addition, we use a technique of boosting for the structured classification algorithm which makes it a non-parametric algorithm. This avoids the choice of meta-parameters. Finally, we test different imitation algorithms on data sets of real laughs in a natural interaction context which is the yes/no game described in Sec. 4.

2. IMITATION LEARNING

Describing the optimal behavior of the avatar is a very tricky task. It would require the perfect knowledge of rules prevailing to the generation of laughter by humans. Interpreting sources of laughter or predicting laughter from a cognitive or psychology perspective is non-trivial. Therefore, a data-driven method has been preferred here. Especially, learning by imitation seems the best suited framework to learn the IM policy. Indeed, humans are implementing such a policy and they can provide examples of natural behaviors.

Formally, in the learning by imitation framework, an artificial learning agent (here the IM) learns to behave optimally by observing an expert agent demonstrating the task. The expert is implementing a certain behavior which is to be learned and can provide examples of natural behaviors.

One way to address the problem of imitation learning is to reduce it to a Multi-Class Classification (MCC) problem [5, 6, 7, 8]. The goal of MCC is, given a training set $D = \{(s_i, a_i)\}_{1 \leq i \leq N}$ where $S$ is a compact set of inputs (generally a compact subset of $\mathbb{R}^n$) and $A$ a finite set of labels, to find a decision rule $\pi \in \mathcal{A}$ that generalizes the relation between inputs and labels. More formally, it consists in finding a decision rule $\pi$ such that $\pi(s) \approx \pi^*(s)$ from the set of demonstrations.

One way to address the problem of imitation learning is to reduce it to a Multi-Class Classification (MCC) problem [5, 6, 7, 8]. The goal of MCC is, given a training set $D = \{(s_i, a_i)\}_{1 \leq i \leq N}$ where $S$ is a compact set of inputs (generally a compact subset of $\mathbb{R}^n$) and $A$ a finite set of labels, to find a decision rule $\pi \in \mathcal{A}$ that generalizes the relation between inputs and labels. More formally, it consists in finding a decision rule $\pi \in \mathcal{H}$, where $H \subset \mathbb{R}^S \times A$ is called the hypothesis space, that tries to minimize the following empirical risk:

$$T(\pi) = \frac{1}{N} \sum_{i=1}^{N} 1(\pi(s_i) \neq a_i),$$

where $1(x) = 1$ if $a \neq b$ and 0 otherwise.

A large literature already exists about the MCC problem. Well known methods such as Classification Trees [9], K-Nearest Neighbors (KNN) [10] and Support Vector Machines (SVM) [11, 12] are widely used and statistically studied. In [5], the authors use an arti-
cial neural network to learn a driving policy for a robotic arm.
Neural nets are also used in [7] to learn to play video games (al-though the method is more generic and could use other MCC meth-
ods). KNN’s where used in [2] in a similar application as the one
described in this paper. In [4], structured classification [13] is used
to learn a grasping control policy for a robotic arm.

3. STRUCTURED CLASSIFICATION FOR IMITATION
LEARNING
In [4], the authors use a large margin approach which allows adding
some prior (or structure) via a margin function in the classification
method. That is why it is considered as a structured classification
method. The large margin approach is a score-based MCC where
the decision rule $\pi \in A^S$ is obtained via a score function $q \in \mathbb{R}^{S \times A}$
such that $\forall s \in S, \pi(s) = \text{argmax}_{a \in A} q(s, a)$. The large margin
approach consists, given the training set $D$, in solving the following
optimization problem:

$$q^* = \text{argmin}_{q \in \mathbb{R}^{S \times A}} J(q),$$
$$J(q) = \frac{1}{N} \sum_{i=1}^{N} \max_{a \in A} \{g(s_i, a) + l(s_i, a, a)\} - q(s_i, a_i),$$

where $l \in \mathbb{R}^{S \times A}$ is called the margin function. If it is zero, mini-
mizing $J(q)$ attempts to find a score function $q^*$ for which the exam-
ple labels are scored higher than all other labels. Choosing a nonzero
margin function improves generalization [6]. Instead of requiring
only that the demonstrated label is scored higher than all other labels,
it requires it to be better than each label $a$ by an amount given by the
margin function. Thus, the margin function allows deciding which
samples are required to be well classified by putting an important
margin on this particular example compared to the others. The policy
outputted by this algorithm would be $\pi(s) = \text{argmax}_{a \in A} q(s, a)$
where $q$ is the output of the minimization of $J(q)$. The advantages
of this method is its simplicity and the possibility to change the
margin that allows us to adapt to specific characteristics of the prob-
lem. In addition, in [4], the authors use a boosting technique to
solve the optimization problem given by Eq. (1) which is advanta-
geous. A boosting method is an interesting optimization technique: it
minimizes directly the criterion in Eq. (1) without the step of choos-
ing features. As presented in [15], a boosting algorithm is a pro-
jected sub-gradient descent [16] of a convex functional (here $J$ is a
convex relatively to the variable $q$) in a specific functions space (here
$\mathbb{R}^{S \times A}$) which has to be a Hilbert space. Boosting algorithms use a
projection step on a restriction set of functions when optimizing over
functions space, because the functions representing the gradient are
often computationally difficult to manipulate and do not generalize
well to new inputs [15]. In boosting literature, the restriction set
corresponds directly to the set of hypotheses generated by a weak
learner. In our experiments, we choose as restriction set the set of
classification trees with two classes.

4. EXPERIMENTAL SETUP
The yes/no game is one of the possible scenarios of an interac-
tion between humans and avatars where laughter is involved. In this game,
players must respond to questions without saying “yes” or “no”. The
experiment scenario we present in this article is illustrated in Fig. 2.
Two users are sitting on one side of a table while a virtual agent pro-
tected on a large screen is placed on the opposite side of the table.
The users start to play the yes/no game, one asking questions (e.g.,
“what’s your nationality?”, “are you sure?”), this user is named $U_1$,
and the other one answering trying to avoid to say “yes” or “no”
(e.g., “I’m not sure” or “definitely”), this user is named $U_2$. The
avatar, named $A$, participates to the interaction by laughing and ask-

Fig. 1. Experimental setup.
5. RESULTS

In this section, we present the results obtained by applying different imitation learning algorithms to the expert data set. We use 4 different algorithms, 3 classical classification algorithms, which are KNN, Classification Tree and SVM, and the large-margin algorithm presented in Section 3. The KNN algorithm was previously used in [2] where we chose those different criteria in order to see the quality of each algorithm on the laugh samples and the no laugh samples because those two classes are not well balanced (basically there is 5 times more no laugh samples than laugh samples). In Table 1 we have the results of the different algorithms in terms of classification rates with $P = 5$.

The Large Margin has the best results for the global classification rate and the no laugh rate. The structure of the margin favors the performance on the no laugh samples and it is reflected in the results. KNN works well on the laugh samples which is also the case of the Classification Tree but has a really poor global performance. It seems that the avatar is too reactive (laughs too often) which can be problematic if the laughs happen on inappropriate moments: this behavior appears unnatural. In order to check if the good performance on laughs of KNN is due to the fact that it is too reactive, we computed the number of laughs produced for each fold and took the mean. Results are provided in Table 1.

The Classification Tree and the KNN avatar are too reactive which can explain their good performance on laughs but their behavior is not natural compared to the expert. The most natural behavior is the one produced by the Large Margin algorithm which laughs in the same proportion than the expert. So the classification rates are not appropriate measures to assess the algorithms according to this application.

For this reason, we came up with a measure for naturalness which indicates if the policy produced by the algorithm corresponds to that of the expert. We define the policy $\pi^{\text{algo}}_p \in A^P$ learned on the remaining $P - 1$ sub-samples. In addition, we define, for each sub-sample $D_p$, the number of laugh samples $N^{\text{laugh}}_p = \sum_{j=1}^{N_p} \mathbf{1}\{a_{j,p} = \text{laugh}\}$ and the number of no laugh samples $N^{\text{no laugh}}_p = N_p - N^{\text{laugh}}_p$. Several quality evaluation criteria were used for each algorithm. The first criterion is the mean over the $P$ folds of the global classification rate:

$$\frac{1}{P} \sum_{p=1}^{P} \frac{1}{N_p} \sum_{j=1}^{N_p} \mathbf{1}\{\pi^{\text{algo}}_p(s_{j,p}) = a_{j,p}\}.$$ 

The second criterion is the mean over the $P$ folds of the classification rate on laugh samples:

$$\frac{1}{P} \sum_{p=1}^{P} \frac{1}{N^{\text{laugh}}_p} \sum_{j=1}^{N^{\text{laugh}}_p} \mathbf{1}\{\pi^{\text{algo}}_p(s_{j,p}) = a_{j,p}\} \mathbf{1}\{a_{j,p} = \text{laugh}\}.$$ 

The third criterion is the mean over the $P$ folds of the classification rate on no laugh samples:

$$\frac{1}{P} \sum_{p=1}^{P} \frac{1}{N^{\text{no laugh}}_p} \sum_{j=1}^{N^{\text{no laugh}}_p} \mathbf{1}\{\pi^{\text{algo}}_p(s_{j,p}) = a_{j,p}\} \mathbf{1}\{a_{j,p} = \text{no laugh}\}.$$ 

We choose those different criteria in order to see the quality of each algorithm on the laugh samples and the no laugh samples because those two classes are not well balanced (basically there is 5 times more no laugh samples than laugh samples). In Table 1 we have the results of the different algorithms in terms of classification rates with $P = 5$.

![Real Demo.](image)

Table 1. Classification rates.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Global</th>
<th>laugh</th>
<th>no_laugh</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large Margin</td>
<td>0.6871</td>
<td>0.3256</td>
<td>0.8081</td>
</tr>
<tr>
<td>SVM</td>
<td>0.6723</td>
<td>0.3106</td>
<td>0.7893</td>
</tr>
<tr>
<td>KNN</td>
<td>0.5440</td>
<td>0.6347</td>
<td>0.5173</td>
</tr>
<tr>
<td>Tree</td>
<td>0.5570</td>
<td>0.5106</td>
<td>0.5732</td>
</tr>
</tbody>
</table>
to the behavior of the expert. The idea is to compare if relatively to the two other users the human playing the avatar and the algorithm have the same behavior.

In order to see if there is a similarity between the behavior of the user playing the avatar $A_{\text{expert}}$ and the one learnt by the algorithms $A_{\text{algo}}$, we check if the behavior of the expert $A_{\text{expert}}$ compares to the users ($\langle U_1, U_2 \rangle$) similarly to the way the avatar's behaviour $A_{\text{algo}}$ compares to the users ($\langle U_1, U_2 \rangle$). The idea is to show that the avatar doesn't differ more from $U_1$ and $U_2$ than the expert does. To do so, for each user $U_q$ and each sub-sample $D_p$, we define the number of laugh samples $N_{\text{laugh}}^{p,U_q} = \sum_{j=1}^{N_p} 1_{\{a_{j,p} = \text{laugh}\}}$ and the number of no laugh samples $N_{\text{no laugh}}^{p,U_q} = N_p - N_{\text{laugh}}^{p,U_q}$. Three criterions were used: the global rate, the laugh rate and no laugh rate. The global criterion $\text{rate}_{\text{avatar}}$ is the rate of agreement in terms of actions between one of the user and an avatar sample by sample:

$$\frac{1}{2} \sum_{q=1}^{2} \sum_{p=1}^{P} \frac{1}{N_p} \sum_{j=1}^{N_p} 1_{\{p_{\text{ avatar}}^{\text{expert}}(s_j,p) = a_{j,p}^{\text{U_q}}\}},$$

where $p_{\text{ avatar}}^{\text{expert}}(s_j,p) = p_{\text{ expert}}^{\text{U_q}} = a_{j,p}$. The laugh criterion $\text{rate}_{\text{avatar}}^{\text{laugh}}$ gives the rate of agreed laughs between the avatar and one of the users:

$$\frac{1}{2} \sum_{q=1}^{2} \sum_{p=1}^{P} \frac{1}{N_p} \sum_{j=1}^{N_p} 1_{\{p_{\text{ avatar}}^{\text{expert}}(s_j,p) = \text{laugh}^{\text{U_q}} = a_{j,p}^{\text{U_q}}\}}.$$

The no laugh criterion $\text{rate}_{\text{avatar}}^{\text{no laugh}}$ gives the rate of agreed no laughs between the avatar and one of the users:

$$\frac{1}{2} \sum_{q=1}^{2} \sum_{p=1}^{P} \frac{1}{N_p} \sum_{j=1}^{N_p} 1_{\{p_{\text{ avatar}}^{\text{expert}}(s_j,p) = \text{no laugh}^{\text{U_q}} = a_{j,p}^{\text{U_q}}\}}.$$

In order to have a single number representing the similarity between the expert avatar $A$ and the avatars outputted by the algorithms, a new criterion, called $\text{Naturalness} N_{\text{algo}}$, is defined as follows:

$$N_{\text{algo}} = \prod_{i=1}^{3} \frac{\min(\text{rate}_{\text{algo}}^{\text{laugh}}, \text{rate}_{\text{expert}}^{\text{laugh}})}{\max(\text{rate}_{\text{algo}}^{\text{no laugh}}, \text{rate}_{\text{expert}}^{\text{no laugh}})}.$$

This criterion is thus a measure of the deviation between the behavior of the expert avatar and the behavior learnt by a given algorithm. If the $\text{Naturalness}$ is equal to 1, it means the avatar has the same behavior as the expert relatively to the other users and if it is equal to zero, it means that the avatar has a completely different behavior than the expert.

Table 2 gives the results. The Large Margin method clearly outperforms the other ones, which means that its behavior relatively to the other users corresponds closely to the one of the expert. We see that the KNN and the Tree have poor $\text{Naturalness}$ as they laugh too much relatively to the other users which is not what the expert does.

### 6. CONCLUSION AND PERSPECTIVES

In this paper, a method for learning when an avatar should laugh during an interaction with humans was presented. It is based on a data-driven imitation learning algorithm and especially on structured classification method. The structured margin implied in this method is used to weight the importance of laughter compared to silence so as to generate a more natural behaviour and deal with the unbalanced nature of data. It is shown, in a yes/no game setting, that the method outperforms other classification methods in terms of overall similarity with a human. Compared to previous experiments [2], this method objectively provides better results in terms of a newly introduced criterion.

Here, imitation learning is reduced to a multiclass classification problem. Yet, imitation learning can also be solved by other methods such as inverse reinforcement learning [18][19]. Actually, this method has been shown to work better for some types of problems [20] and has already been used to imitate human users in the case of spoken dialogue systems [21]. Therefore, we plan to extend this work to inverse reinforcement learning in the near future. Also, this method could be used to generate new simulation techniques for optimizing human machine interaction managers in other applications such as spoken dialogue systems [22][23].

### 7. ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement number 270780.

### 8. REFERENCES
