AoL: Action Learning: A methodology to capture expertise in adjustment tasks

Francisco J Ruiz Albert Samà Cristóbal Raya
BarcelonaTech
Vilanova i la Geltrú (Spain)

Núria Agell
ESADE-URL
Sant Cugat del Vallès (Spain)

Abstract

It is well known that some people can perform a task with greater precision and accuracy than others: they are experts. In the past, experts were interviewed to find out why they have this expertise, but this was not always completely effective because often experts "don't know what they know". In this paper we propose a model of the process of making decisions performed by experts in the final adjustment of products task. Based on this model, we also propose a system based on a machine learning module that facilitates the capture of these expert skills. We give an example to illustrate the process proposed.

1 Introduction

It is well known that some people can perform a task with greater precision and accuracy than others: they are *experts*. Industries in which product quality depends on specialized experts spent many resources managing and trying to replicate these skills. In particular, perfume, food, beverage, painting, and other creative industries continuously deal with problems in modeling processes based on the cognitive ability of these highly specialized individuals [1, 2]. In these tasks, the intervention of human experts, including colourists, perfumers, chefs, sommeliers, or brew masters, becomes necessary, preventing the complete process automation.

In the past, experts were interviewed to find out why they have this expertise, but this was not always completely effective because often experts "don't know what they know". In addition, experts are not always very enthusiastic with this collaboration.

Experts are involved in several tasks in the production process. Two of these tasks are the formulation task and the adjustment or tuning task. On the one hand, the formulation task concerns the process of finding an appropriate set of ingredients, their proportions and the process steps in order to get a target product. Once the formulation task is completed, the product is ready to be manufactured in the production phase.

On the other hand, the adjustment or fine-tuning task is performed during manufacturing. This task must be performed whenever the product is nearly finished and has to be corrected in order to achieve the target product with the desired precision and quality. In the adjustment process, the expert, based on his/her sensory experience and abilities, determines slight variations in the proportions of one or more ingredients. The intuition of the expert, usually, does not allow him/her to know the exact quantities to increase to achieve the goal or target. However, he/she is able of iteratively determining approximate quantities to add until the final target is met. Not only formulation but also adjustment requires a lot of highly qualified human and time resources.

In this paper we propose a process model of making decisions performed by experts in the adjustment task and, based on this model, we also introduce an innovative artificial cognitive system to support decision making in adjustment processes based on human sensory abilities. The proposed system, based on expert knowledge management, draws on a machine learning tool jointly with an actions' generator module. A specifically-adapted Support Vector Machine (SVM) [1][2] is previously trained with 'state-action' type patterns provided by experts. Then, it enables identifying and selecting the most adequate action among those provided by the generator module for a particular state. The coupled actions' generation-selection process is iterated until the final state satisfies certain conditions, i.e. until the target is achieved.

The remainder of this paper is organized as follows. In section 2, an expert decision making modelling in the adjustment task is presented. Section 3 is devoted to the architecture of the system based on a machine learning module that facilitates the capture of expert skills. In section 4, a specific machine learning system, Support Vector Machines is proposed to be part of the system. In section 5, the active learning paradigm applied in this case is explained. An artificial example is shown in order to illustrate the model and the system. Finally, section 6 includes the conclusion and future research issues.

2. Expert adjustment model

The process of adjustment is formulated as a Deterministic Markov Decision Process (DMDP) composed mainly by a set of states, a set of actions and an immediate reward function that quantifies the benefit of choosing one or another action. In a DMDP, a policy is a function that specifies the action chosen in each state. The core problem of DMDP is finding an optimum policy, which is the policy that maximizes some cumulative function of rewards. Reinforcement Learning is a common paradigm to deal with DMDP. It comes into play when examples of desired behavior are not available and it is based on the trade-off between *exploration*, or discover new actions, and *exploitation*, that is related to the preference of actions that have already been shown to be useful. In contrast to Reinforcement Learning paradigm, our proposal takes advantage of expert knowledge by reducing the cost of the exploration phase using a standard supervised machine learning to induce the reward function. Our proposal is similar to methods commonly used in Robotics and known as Learning by Demonstration [8]. However, robotics approaches are more aimed at solving simple tasks rather that capturing expert specialized skills

Formally, we consider a system that is determined at a given moment by a state $s_i \in \mathbf{S}$. For each state s_i, a set of actions A_i (either finite or not) is associated. When an action $a \in A_i$ is carried out, a transition takes place, so that the system moves from the state s_i to the state s_{i+1}. In the deterministic cases, it is verified that the final state is a function of the current state s_i and the performed action a, so $s_{i+1}=F(s_i,a)$. Function F is named *effect function*. Effect function is normally not known by the expert. The truly effect of an action is only known when the action is actually performed. The expertise of the expert comes from the partial knowledge of other structures: the *fitness quasi-order relation* and the *immediate reward*.

The fitness quasi-order relation allows ordering the states. We use the term quasi-order to emphasize that it is not strictly an order, it is a reflexive and transitive binary relation but not necessarily anti-symmetric. The fitness quasi-order relation can be derived from a fitness function, i.e. a function from $\mathbf{S}$ to R that quantifies the suitability of each state. However, most of the cases, only the quasi-order relation is known by the expert, that is, given two states, the expert can order them but he/she is unable to assign an absolute value to each state. Depending on the mathematical structure of $\mathbf{S}$, it is possible to assign an objective fitness function. For instance, if there exists a distance d defined on $\mathbf{S}$, and the target state s_i is known, the function $d(.,s_i)$: $\mathbf{S} \rightarrow R$ is a fitness function. Of course, it is possible to consider more than one fitness function in the same problem, and also more than one fitness quasi-order relation.

The immediate reward is a function from $\mathbf{S}\text{x}A \rightarrow R$ that quantifies the effect of an action given a state. It might appear that immediate reward can be directly deduced from the effect function and the fitness function f, that is: $R_w(s,a)=\pm(f(F(s,a))-f(s))$. However, the immediate reward may depend on other factors such as the cost of performing the action, time constraints, etc.

Our objective is to extract the knowledge that allows the expert to choose the correct action given a state, that is, a policy. The main drawback in this task is that the expert sensory knowledge is not structured. It consists in subjective interpretation of the current state and partial knowledge of the effect function, fitness function, fitness quasi-order relation and immediate reward. This knowledge is based on the expert intuition, that is, he/she knows which action is more suitable in each case but normally the expert cannot explain why.

3. Architecture System

In order to replicate expert abilities it is necessary to observe how experts work. First, when experts adjust a product, they rarely obtain the target state using just one action; instead they usually perform a sequence of actions so that each action try to correct the state obtained from the previous action. Second, when the experts decide to perform an action, they are quite sure that this action will improve the current state, but normally they are not so sure how much the state will be improved. If a set of actions is involved in the decision, experts, at most, are able to roughly rank the actions and select the most value action.

In regard to the first observation, we propose a closed-loop architecture in which once an action is selected, it is performed and the state obtained is feed again to the system for deciding another action. This process is continuing until the target state is obtained with the desire level of precision. In regard to the second observation, we propose a supervised learning algorithm that tries to capture a simplified version of the immediate reward. The most simplified version is one that associate value 1 (good) or value -1 (bad) to each pair state-action.

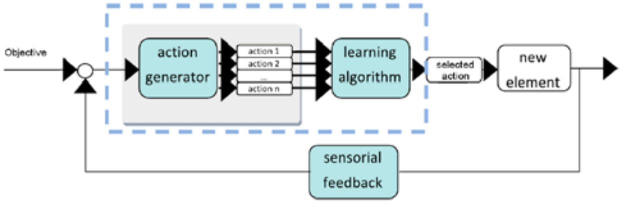

Fig 1 Architecture of the system,

Our proposal basically consists of a closed loop architecture formed by an action generator used jointly with a supervised learning algorithm, concretely a Support Vector Machine (SVM), although other algorithms may also be considered. The suitability of SVM in this approach is based on the work of Platt [1], which proposes to map the SVM outputs to posterior probabilities and hence outputs of an SVM for classification algorithm are not only used to assign a label to a test pattern but also to map it to a graduate scale of belonging to one class or another. In this way, the actions generated by the action generator can be used as inputs for the trained SVM and the outputs allow us to order these actions for selecting the best one.

The action generator is the module that contains the restriction and previous knowledge of the problem to be solved. It proposes performing only those actions that are available and compatible with the current state and with the product to be manufactured. The efficiency of the method depends heavily on the design of this module.

4. Learning module: Support Vector Machines

SVM are a class of learning algorithms that combine a strong theoretical motivation from the Statistical Learning Theory, optimization techniques, and the kernel mapping idea [2]. The original input vector from the input space X is mapped by means a proper kernel function to a higher-dimensional feature space F where the pattern discrimination is simpler, i.e. where a linear separating hyperplane exists.

Each hyperplane in feature space is determined by the expression: $\mathbf{w}\cdot\mathbf{x}+\mathrm{b}=0$ where $\mathbf{x}\in F$. Do not matter if F is an infinite dimensional space since, in practice, the discriminant function is written in terms of the original patterns from input space X, the kernel function and the Lagrange multipliers obtained by solving the dual optimization problem: $\sum \alpha_i \cdot k(x_i, x) + b = 0$, where x_i, $x\in X$.

The SVM algorithm search for the separating hyperplane maximizing the margin, that is, maximizing the minimum distance between the hyperplane and the training patterns. Training patterns closest to the hyperplane are known as support vector and it depends only of them.

For a vector $\mathbf{x}\in F$ it is satisfied that: $\mathbf{w}\cdot\mathbf{x}=\pm d\cdot\|\mathbf{w}\|$, where d is the distance of $\mathbf{x}$ to the hyperplane and the sign determines the vector'**x** side of the hyperplane, i.e. the class of the vector **x**. Therefore, it is clear from the last expression that the output of the SVM algorithm allows to know how far **x** is from the hyperplane in F or from the non-linear discriminate surface in X. Therefore, each hyperplane in F induces an order relation into the input space X that permits to select from among several inputs which is in the farthest from the hyperplane in the feature space. If the two classes are relating with a positive and a negative characteristic, the distance from the hyperplane determines a graduation of this characteristic.

5. Active learning on action learning

The learning phase of action learning is not very different to other classification problems. However, in this case, the patterns are pairs state-action and normally they are hard to label. The active learning process proposed starts by selecting a few patterns at random and from these to find a classifier by means a SVM algorithm. The trained model allows to obtain with little cost the decision value for any unlabeled pattern state-action. The next step is to select a new pattern to be labeled from the SVM decision.

The process steps are:

- step 1: Seed the search with representative patterns of patterns selected randomly.
- step 2: Train an SVM on all labeled examples.
- step 3: Use the trained SVM to obtain the decision values of a large amount of unlabeled instances (pairs state-action) and selecting one (or more) to query with the lower decision values (those closest to the separating hyperplane in feature space).
- step 4: With the help of an automatic o human ”‘oracle”’, label the instances obtained from step 3.
- step 5: Repeat step 2 to 4 until a desired number of training patterns.

6. An artificial example: The learning phase

In this section, we propose an artificial example in order to understand the concepts defined above. Let us suppose that the space of states **S** is the real line and the space of action A is also the real line, so given a state s and an action a, the effect of performing the action *a*, given *s* is *F(s,a)=s+a*. Function *F* is the function that we called *effect function*.

We consider a fitness function *f* which assigns each state $s \in \mathbf{S} = R$ a value related to the suitability of *s* in a specific task. The function used in this example is:

$$f(x)=x^2+x+20\cdot\sin(x)$$

that is represented in figure 2. This function has a global minimum at x≈-1.47 but has other local minima.

In this section, we compare the active action learning methodology with the passive one on the learning phase via the artificial experiment.

Obviously, we have an infinite pool of unlabeled pairs state-action (R^2) and we need to select a subset to assign a class for training the SVM. In this artificial experiment we compare the performance of the training phase in two cases: selecting the training pattern at random (passive learning) or selecting a few pattern at random (we have selected 10 patterns) and use them to iterative selecting a new pattern and increasing the training set.

The above procedure was repeated thirty times and the results were averaged. Figure 3 and 4 show an example of 200 patterns selected using active and passive methodology.

It can be seen that active learning only select a few patterns far the discriminate surface.

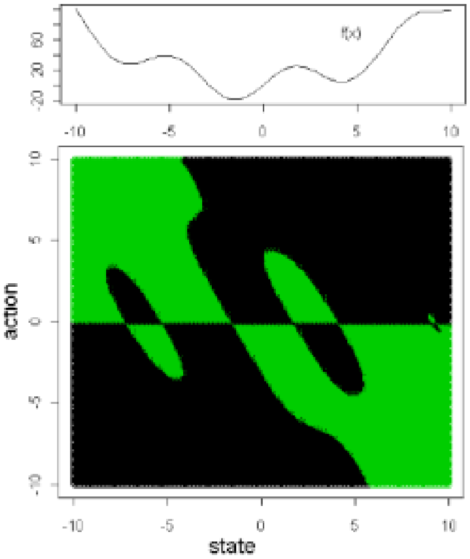

Fig 2 Function f used in the example and 2D state-action space associated to *f*,

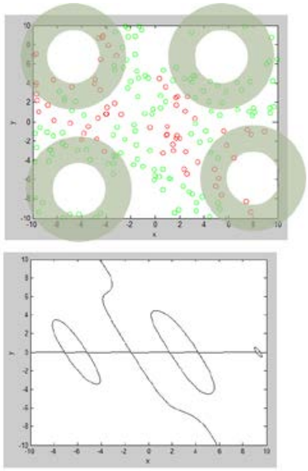

Fig 3 Action states space and selected patterns using an active learning paradigm

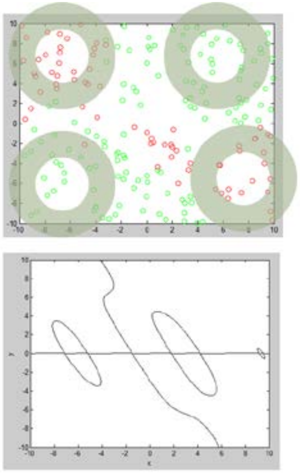

Fig 3 Action states space and selected patterns using a pasive learning paradigm

Conclusion and Future Works

In some industrial processes, the final adjustment task has always been considered as inevitably manual. This task is performed by experts whose intuitive knowledge is used in order to improve the final product. For addressing this challenge, previous works proposed a methodology named action learning that uses a classification learner for classifying action given a state. This paper tries to improve the most difficult phase of the action learning process, the learning process. It has been proved that taken an active learning methodology this learning phase is significantly improved.

References

[1] J. Platt. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods.Advances in Large

Margin Classifiers (A. Smola, P. Bartlett, B. Scholkopf, D. Schuurmans, (Ed), MIT Press, 1999.

[2] N. Cristianini, J. Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, 2000.

[3] F. Ruiz, C. Angulo, N. Agell. Support Vector Machines for color adjustment in automotive basecoat. In Proceedings of Catalan Congress of Artificial Intelligence. CCIA 2006.

[4] D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In Proceedings of the 11th International Conference on Machine Learning. Morgan Kaufmann. 1994.

[5] T. Mitchell. Generalization as search. Artificial Intelligence, 28:203-226. 1982.

[6] S. Tong and D. Koller. Support Vector Machine Active Learning with Applications to Text Classification. Journal of Machine Learning Resarch, 45-66. 2001

[7] B Settles. Active Learning Literature Survey. Computer Sciences Technical Report 1648. University of Wisconsin-Madison. 2010

[8] Brenna D. Argall, Sonia Chernova, Manuela Veloso and Brett Browning. A survey of robot learning from demonstration. Robotics and Autonomous Systems 2008