Supervised vs. Unsupervised Learning for Intentional Process Model Discovery

Ghazaleh Khodabandelou, Charlotte Hug, Rebecca Deneckere, Camille Salinesi

Supervised vs. Unsupervised Learning for Intentional Process Model Discovery

Ghazaleh Khodabandelou, Charlotte Hug, Rebecca Deneckère, Camille Salinesi

Centre de Recherche en Informatique, Panthéon-Sorbonne University, France
{Ghazaleh.Khodabandelou}@malix.univ-paris1.fr
{Charlotte.Hug,Rebecca.Deneckere,Camille.Salinesi}@univ-paris1.fr

Abstract. Learning humans' behavior from activity logs requires choosing an adequate machine learning technique regarding the situation at hand. This choice impacts significantly results reliability. In this paper, Hidden Markov Models (HMMs) are used to build intentional process models (Maps) from activity logs. Since HMMs parameters require to be learned, the main contribution of this paper is to compare supervised and unsupervised learning approaches of HMMs. After a theoretical comparison of both approaches, they are applied on two controlled experiments to compare the Maps thereby obtained. The results demonstrate using supervised learning leads to a poor performance because it imposes binding conditions in terms of data labeling, introduces inherent humans' biases, provides unreliable results in the absence of ground truth, etc. Instead, unsupervised learning obtains efficient Maps with a higher performance and lower humans' effort.

Keywords: Supervised Learning, Unsupervised Learning, Intentional Process Modeling, Hidden Markov Models

1 Introduction

Fueled by the impressive growth of events logs in organizations, process mining field has emerged a few years ago as a key approach to design processes [1, 2]. Mining processes from logs can be useful for understanding how humans really work, analyzing how actual processes differ from the prescribed ones (conformance checking). This allows improving models, methods and products.

Whereas most process mining approaches specify behaviors in terms of sequences of tasks and branching [2], research on method engineering and guidance shows that an explicit use of intentions in process models structure could effectively mitigate the method engineering issues such as rigidity or lack of adaptation [3–7].

Intention-oriented process modeling emerged at the early 90s, as a driving paradigm. It allows supporting guidance [8], handling traceability matters [4], guiding requirements elicitation, surveying strategic alignment [9], defining actors and roles, specifying the outcome of business process models [10], describing intentional services [11], diagnosing use cases, analyzing users behavior, customizing methods or make them more flexible [3], etc. Defining strategies and

intentions in process model structure has convinced as a robust mean to identify and analyze the relationships between processes, to understand the deep nature of processes, and to visualize any process (simple or complex) under a reduced and human-understandable form [5]. While intention-oriented process modeling has a longer tradition, it has largely neglected event logs so far. Map Miner

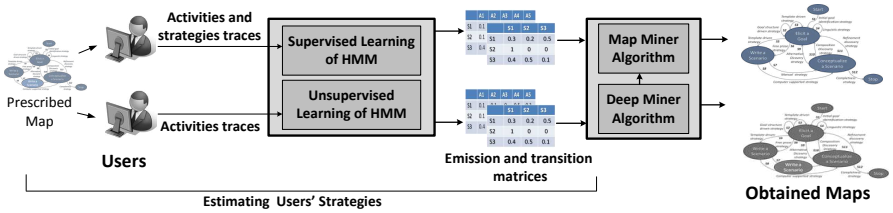

Fig. 1. The overview of Map Miner Method

Method (MMM) is a novel approach of process mining, which aims at constructing intentional process models from users' event logs. As a first step, the MMM framework uses Hidden Markov Models (HMMs) [12] to estimate users' strategies. Then, it generates intentional process models (Maps [5]) using Deep Miner and Map Miner algorithms [13]. This paper focuses on the first part of MMM: estimating users' strategies. These strategies can be estimated either with *supervised learning* or with *unsupervised learning*. While Supervised learning can be used when there is *a priori* knowledge about strategies in dataset, unsupervised learning can be used when there is no such knowledge available. Both learning approaches aim at characterizing the strategies that correspond the best to the users' activities in the event logs. These strategies will then be used to construct intentional process models. Thus, the choice of learning approach significantly impacts the discovered model accuracy. Hence, it is important to study limitations, advantages and conditions of use for these learning approaches.

The contribution of this paper is twofold: (i) first, in a theoretical context, it compares supervised and unsupervised learning of strategies in terms of convergence speed (complexity) and likelihood; and (ii) second, in an experimental context, it compares the intentional process models obtained with both learning approaches. The resulting Map process models provide a precious understanding of these approaches in terms of their performance as well as their conditions of use. Figure 1 depicts an overview of MMM framework, in which the focus of this paper is shown in the part of estimating users' strategies.

The remainder of this paper is organized as follows: in Section 2, we introduce the MMM and a brief definition of Map process models. In Section 3, supervised and unsupervised learning are described and then formally compared. In Section 4, both approaches are applied on two real datasets. Section 4.3 discusses the results of both approaches as well as the threats to validity. Related works are investigated in Section 5. Finally, Section 6 concludes this work and presents the perspectives.

2 Map Miner Method

MMM automatically constructs intentional process model from users' event logs. MMM consists of three phases: (1) it estimates users' strategies from activity logs using HMMs, (2) it constructs fine-grained Map process models from estimated strategies using Deep Miner algorithm (1), and (3) it constructs coarse-grained Map process models from fine-grained ones using Map Miner algorithm (2) (figure 1). As mentioned earlier, this paper concentrates only on the first phase of MMM, *i.e.*, estimating users' strategies with the Map formalism. Among other intentional process models such as KAOS [14] or I* [15], we chose the Map formalism for several reasons: (a) it combines intentions and strategies at different abstraction levels, which allows handling large-scale and complex processes [5], (b) it supports process variability and flexibility by defining different strategies to fulfill a given intention, and (c) it has proven to be effective to specify business processes, user requirements, systems functionality, engineering methods, software engineering processes, etc [7]. Next parts explain briefly the Map metamodel and how HMMs can be adapted to it.

2.1 The Map Metamodel

The Map formalism [7] combines the concepts of *intention* and *strategy* with hierarchical abstraction levels and refinement links. In this framework, an intention is defined as a goal, an objective or a motivation to achieve with clear-cut criteria of satisfaction, which can be fulfilled by enacting a process [16]. The intentions are explicitly represented and form the high-level goals (*e.g.*, organizational goals). A Map process model (an instance of Map metamodel) specifies the multiple ways of working (*i.e.*, strategies) for fulfilling a set of intentions during the enactment of a process. For example, on figure 2, one way to fulfill the intention *Specify an entity* is to select the strategy S_4: *By generalization*. In the next section, we explain how strategies can be linked to the activities logs.

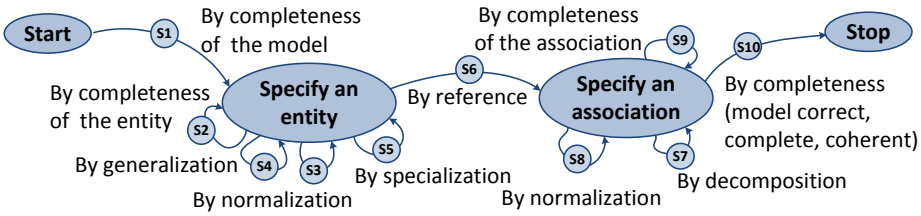

Fig. 2. Map process model for the construction of Entity/Relationship diagrams.

2.2 Estimating Strategies from Activity Logs

Among the techniques to model different aspects of humans' behavior [17], HMMs have been proven to be appropriate for modeling the real world process,

particularly unobservable cognitive states [18], such as underlying users' strategies. HMMs are stochastic Markov chains used for modeling a hidden sequence by a finite number of states. More precisely, HMMs consist of two complementary Markov processes: hidden and observed processes, such that the states of hidden process generate the symbol of observed process. It turns out that the topology of HMMs is particularly adapted to model the relation between strategies and activities in the Map formalism. To make it clear, let us consider an example for a Map process model enacted with 2 strategies and an HMM realized with 2 hidden states (see figure 3). As shown this figure, strategies are used to move from one intention to another and are made of one or several activities. For instance, the strategy 1 allows moving from intention a to intention b and it is made of activities a_1, a_3 and a_4. The same structure can be found in an HMM, where hidden states generate observations. In other words, hidden state 1 generates the observations a_1, a_3 and a_4, or hidden state 2 generates the observations a_4 and a_7. This similar topology motivates using HMMs to model activity logs and users' strategies.

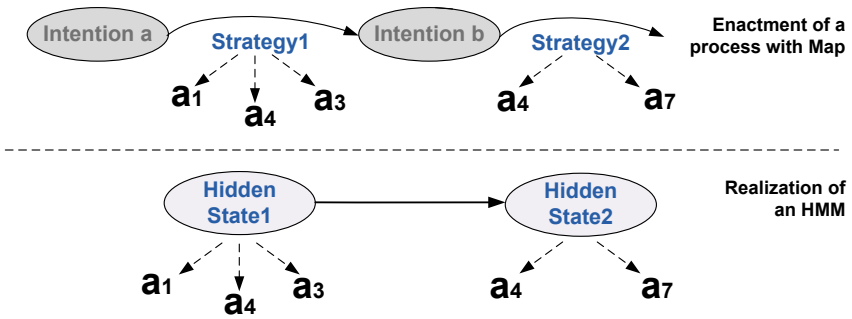

Fig. 3. An example for a Map process model enacted with 2 strategies (above) and an HMM realized with 2 hidden states (below)

Formally, the definition of an HMM is the tuple: $\mathcal{H} = \{\mathcal{S}, \mathcal{A}, \mathbf{E}, \mathbf{T}, \pi\}$, where $\mathcal{S}$ is the set of possible hidden states, $\mathcal{A}$ represents the set of possible observations, $\mathbf{T}$ is the states transition matrix, *i.e.*, the matrix which represents the probabilities of transition from one state to another, $\mathbf{E}$ is the observations emission matrix, *i.e.*, the matrix which represents the probabilities that a given observation appears in a given hidden state, and π is the vector made of the initial probabilities of hidden states. In the MMM framework, the users' strategies are modeled by the hidden process and the users' activities are modeled by the observed process.

Hidden Process: Users' Strategies. Let $\mathbf{s} = (s_1, \ldots, s_L) \in \mathcal{S}^L$ be a temporal sequence of users' strategies of length L. The hidden process of strategies is parametrized by the vector π, $\pi(u) = \Pr(s_1 = u) \ \forall u \in \mathcal{S}$ and the matrix $\mathbf{T}$ such

that:

$$\mathbf{T}(u,v) = \Pr\left(s_\ell = v | s_{\ell-1} = u\right) \ \forall u, v \in \mathcal{S},\ \ell \in [2, L], \tag{1}$$

Observed Process: Users' Activities. Let $\mathbf{a} = (a_1, \ldots, a_L) \in \mathcal{A}^L$ be a temporal sequence of users' activities of length L. The emission probability of an observation $a \in \mathcal{A}$ for a given strategy $u \in \mathcal{S}$ is given by:

$$\mathbf{E}(a,u) = \Pr\left(a|u\right). \tag{2}$$

Assuming that $\mathcal{S}$, $\mathcal{A}$ and π are known, the HMM model is fully described by $\{\mathbf{E}, \mathbf{T}\}$, which represent the core information about the HMM behavior. These two matrices provide all the necessary information to characterize the strategies of the Map. Indeed $\mathbf{E}$ gives the relation between each strategy and the activities in the event logs, and $\mathbf{T}$ gives the transition probabilities between strategies. These two matrices have to be learned from the logs and the choice of the learning approach is crucial to ensure that the strategies are correctly characterized.

3 Learning Approaches to Estimate Users' Strategies

As discussed in section 2, the characterization of the strategies of the Map completely relies on the model parameters of the HMM, *i.e.*, the emission matrix $\mathbf{E}$ and the transition matrix $\mathbf{T}$. The learning problem is to find $\mathbf{E}$ and $\mathbf{T}$ that maximize the probability of generating the observed sequences of activities. There are two learning approaches for estimating these matrices: *Supervised* or *Unsupervised* learning. this section discusses in a theoretical context the necessary conditions for using both approaches as well as their respective performances.

3.1 Supervised Learning

Supervised learning aims at learning $\mathbf{E}$ and $\mathbf{T}$. However, the conditions under which it can be used are very restrictive and the results might be biased.

Conditions of Use. The application of this method requires the knowledge of: (a) the sets $\mathcal{A}$ and $\mathcal{S}$, and (b) some sequences of activities $\mathbf{a}_1, \ldots, \mathbf{a}_N$ and their associated sequences of strategies $\mathbf{s}_1, \ldots, \mathbf{s}_N$.

While the knowledge of $\mathcal{A}$ and $\mathbf{a}_1, \ldots, \mathbf{a}_N$ is generally not an issue (the possible activities of a given process are usually known and are recorded in traces), the knowledge of $\mathcal{S}$ and $\mathbf{s}_1, \ldots, \mathbf{s}_N$ is more problematic. Indeed, since strategies are the cognitive operators, the usual way to obtain the set $\mathcal{S}$ is to refer to experts. Since humans' judgment is involved, the obtained set $\mathcal{S}$ can be biased. We argue that in a cognitive context such as a humans' strategy and intention, it is impossible to properly label the training data, because this information is not observable. Moreover, humans' bias [19] is unavoidably introduced into training data labeling, which significantly impacts the learning process and may produce incorrect or uninformative process models. Moreover, strategies are usually not recorded in traces [20]. Applying this learning method implies to conduct experiments specially designed to record traces of activities and traces of strategies. This condition highly restricts the range of use of this method in large-scale.

Performance. Given N sequences of activities $\mathbf{a}_1, \ldots, \mathbf{a}_N$ and their associated N sequences of strategies $\mathbf{s}_1, \ldots, \mathbf{s}_N$, the aim of supervised learning is to find the couple $(\mathbf{E}^*, \mathbf{T}^*)$ which maximizes the likelihood of generating $\mathbf{a}_1, \ldots, \mathbf{a}_N$ and $\mathbf{s}_1, \ldots, \mathbf{s}_N$:

$$(\mathbf{E}^*, \mathbf{T}^*) = \arg\max_{\mathbf{E},\mathbf{T}} \prod_{n=1}^{N} \Pr\left(\mathbf{a}_n | \mathbf{s}_n, \mathbf{E}, \mathbf{T}\right) \tag{3}$$

Obtaining the coefficient of $\mathbf{T}^*$ amounts in counting the number of transitions from one strategy to another and obtaining the coefficients of $\mathbf{E}^*$ amounts to count the number of occurrences of each activity during each strategy, as shown below:

$$\mathbf{T}^*(u, v) = \frac{\mathrm{Num}(u, v)}{\sum_{w \in \mathcal{S}} \mathrm{Num}(u, w)}, \ \forall (u, v) \in \mathcal{S}^2, \tag{4}$$

$$\mathbf{E}^*(u, a) = \frac{\mathrm{Num}(a|u)}{\mathrm{Num}(a)}, \ \forall u \in \mathcal{S}, \ \forall v \in \mathcal{A}, \tag{5}$$

where $\mathrm{Num}(u, v)$ denotes the number of transitions from strategy u to strategy v in the traces $\mathbf{s}_1, \ldots, \mathbf{s}_N$, $\mathrm{Num}(a)$ denotes the number of occurrences of activity a in $\mathbf{a}_1, \ldots, \mathbf{a}_N$ and $\mathrm{Num}(a|u)$ denotes the number of occurrences of activity a while the strategy is u, in $\mathbf{s}_1, \ldots, \mathbf{s}_N$ and $\mathbf{a}_1, \ldots, \mathbf{a}_N$. The computation complexity of this method is very low since all the coefficients of $\mathbf{E}^*$ and $\mathbf{T}^*$ can be directly computed from the traces used for learning with (4) and (5).

The set of training sequences $\mathbf{a}_1, \ldots, \mathbf{a}_N$ and $\mathbf{s}_1, \ldots, \mathbf{s}_N$, is extremely important for the accuracy of the estimation of $\mathbf{E}^*$ and $\mathbf{T}^*$. If the set contains few traces, or they are not fully representative of all the traces that can be produced by the process, the HMM model learned out of it might suffer underfitting issues. From a practical point of view, this issue is common since the conditions to get usable training traces are complex (resulting in few usable traces).

3.2 Unsupervised Learning

Unsupervised learning estimates the matrices $\mathbf{E}$ and $\mathbf{T}$ based only on traces of activities. Since there is almost no prior knowledge on the strategies set $\mathcal{S}$, this method is significantly less biased than supervised learning but the associated computational complexity is high.

Conditions of Use. For unsupervised learning, the required knowledge includes the set of activities $\mathcal{A}$, some traces of activities $\mathbf{a}_1, \ldots, \mathbf{a}_N$ and the cardinality of the set $|\mathcal{S}|$, *i.e.* the number of possible strategies. Regarding strategies, neither the set $\mathcal{S}$ nor some traces of strategies $\mathbf{s}_1, \ldots, \mathbf{s}_N$ should be known, only the number of possible strategies is required. This parameter can be chosen by experts (*e.g.* as a way to set the level of complexity of the model) or can be set with techniques such as BIC [21], which makes a trade-off between the likelihood of the model and its complexity. Similarly to supervised learning, this choice introduces a bias, but given that only the number of strategies is set and not the strategies themselves, this bias is less important. The advantage of unsupervised learning is being applicable on datasets comprising only activities traces.

Performance. The Baum-Welch algorithm (BWA) [22] is the most commonly used in HMMs framework to estimate the model parameters $\mathbf{E}$ and $\mathbf{T}$. It uses the Expectation Maximization (EM) algorithm [23]. The aim of the EM algorithm is estimating the *maximum a posteriori* of the statistical model (with latent variable) parameters in an iterative way. Given a dataset made of N observed sequences of activities $\mathbf{a}_1, \ldots, \mathbf{a}_N$, the BWA finds the HMM matrices $\tilde{\mathbf{E}}$ and $\tilde{\mathbf{T}}$ that locally maximize the probability of having these sequences generated by the HMM. More precisely, the BWA maximizes the likelihood of $\mathbf{E}$ and $\mathbf{T}$:

$$\left(\tilde{\mathbf{E}}, \tilde{\mathbf{T}}\right) = \arg\max_{\mathbf{E},\mathbf{T}} \prod_{n=1}^{N} \Pr\left(\mathbf{a}_n | \mathbf{E}, \mathbf{T}\right) \tag{6}$$

As we mentioned earlier, the number of strategies is required to know the dimensions of matrices $\tilde{\mathbf{E}}$ and $\tilde{\mathbf{T}}$ since the BWA could not run without $\tilde{\mathbf{E}}$ and $\tilde{\mathbf{T}}$ being initialized.

What is interesting to note here is the fact the likelihood is not maximized depending on some traces of strategies $\mathbf{s}_1, \ldots, \mathbf{s}_N$, as it was the case for supervised learning. It means that the space in which the likelihood is maximized is larger than the space for supervised learning. As a consequence,

$$\max_{\mathbf{E},\mathbf{T}} \prod_{n=1}^{N} \Pr\left(\mathbf{a}_n | \mathbf{E}, \mathbf{T}\right) \geqslant \max_{\mathbf{E},\mathbf{T}} \prod_{n=1}^{N} \Pr\left(\mathbf{a}_n | \mathbf{s}_n, \mathbf{E}, \mathbf{T}\right). \tag{7}$$

In other words, the maximum likelihood obtained by unsupervised learning is always higher than the maximum likelihood obtained by supervised learning since the latter comes from a constrained space. Unfortunately, the BWA cannot be guaranteed to converge to the global maximum likelihood since it is only proved to converge to a local optimum [12]. The limit of convergence depends on the initialization of the matrices $\mathbf{T}$ and $\mathbf{E}$ and it is verified by our experimental results (see section 4), that a simple initialization of $\mathbf{T}$ and $\mathbf{E}$ leads to a maximum likelihood of unsupervised learning higher than supervised learning.

Another difference with supervised learning is the computational complexity. While the complexity of supervised learning is very low, the BWA requires several iterations to converge to a local optimum. These iterations make unsupervised learning a more expensive method than supervised learning. The precise computation of both methods, applied on tow experiments, are given in section 4.3.

3.3 Summary of the Two Learning Approaches

In table 1, we present a theoretical comparison of the two learning approaches, based on the properties defined in sections 3.1 and 3.2. Regarding the conditions of use, unsupervised learning can be applied on any dataset comprising traces of activities, contrary to supervised learning which can be applied under more restrictive conditions. This makes unsupervised learning the most convenient method for a practical use. However, since unsupervised learning is only proved

Table 1. Theoretical comparison of supervised and unsupervised learning

	Traces for learning	**A-priori knowledge**	**Convergence speed (complexity)**	**Likelihood of the estimated parameters**
Supervised learning	Activities, Strategies	Set of activities, Set of strategies	Fast (one iteration)	Maximum over a restrained set
Unsupervised learning	Activities	Set of activities, Number of strategies	Slow (several iterations)	Local maximum

to converge to a local maximum, it is not guaranteed to provide an estimated model with a better likelihood than supervised learning. In order to investigate about this point, we compare both approaches on the same datasets in the following section.

4 Comparison of the Approaches in Experiments

To compare the supervised and unsupervised learning in a experimental context, we conducted two tailored experiments with the Master students of computer science of Sorbonne University: *Entity/relationship (E/R) diagrams* and *E-shopping*. Due to lack of space, we only show and analyze in details the strategies obtained for E/R diagrams in the current section. In section 4.3, the results of the two learning approaches for the two experiments are compared in terms of several comparison criteria, such as performance (indicated by the likelihood to generate the activities in the event logs), humans efforts, convergence speed, and computation complexity. Note that the traces used for this comparison have to be compatible with both learning approaches. Indeed, they have to comprise traces of activities and corresponding traces of strategies. Although strategies are generally not accompanied recorded logs, we intentionally asked the students participating in the experiments to label their traces of activities to indicate which strategies they followed. The traces of activities are recorded by a web-based tool.

• **E-shopping experiment:** we guided students through a prescribed intentional process model to buy a present. We recorded 90 traces of activity produced by 90 students for which we know the sequence of selected strategies. The size of each trace varies between 6 and 40 activities. Note that with both learning approaches, all the traces are used for training.

• **The Entity/Relationship diagrams experiment:** according the intentional process model given by figure 2, 66 students created E/R diagrams. Here again, all the traces are used for training for both learning approaches. Regarding to this model, students can select ten strategies to fulfill three intentions, *i.e.*, *Specify an entity*, *Specify an association* and *Stop*. From *Start*, it is possible to progress in the process by selecting strategies leading to the intentions but once the intention of *Stop* is achieved, the enactment of the process finishes. To fulfill an intention following a strategy, students have to carry out activities. There are 12 unique activities related to the process. Table 3 gives the name of

the strategies, related activities and the corresponding labels. The size of each trace varies between 3 and 68 performed activities. Table 2 illustrates a fragment of some event logs of the E/R diagrams experiment. Each row represents

UserID	**TraceID**	**Timestamps**	**StrategyLabel**	Activities	...
45	7	31/10/12 14:54:00	1	Create entity	...
22	1	31/10/12 15:14:00	4	Create generalization link	...
12	8	31/10/12 14:54:00	7	Create association	...
45	7	23/10/12 09:41:00	2	Link attribute to entity	...
...	...	...	...	...	...

Table 2. A fragment of some event Log of E/R diagrams experiment

an activity, with its corresponding labeled strategy, its timestamps, the trace ID, and the ID of the user who performed the activity. A trace of activities is the ordered (by timestamps) sequence of activities that a user performed. In sections 4.1 and 4.2, the MMM is applied on this dataset with both supervised and unsupervised learning and strategies and associated Maps are discussed.

Table 3. Strategies and related activities

Labels	**Name of strategies**	**Related activities (activities labels)**
S_1	By completeness (model)	Create entity (a_1)
S_2	By completeness (entity)	Link attribute to entity (a_2)
S_3	By normalization	Delete Link attribute to entity (a_6), Delete entity (a_5), Define primary key (a_7)
S_4	By generalization	Create entity (a_1), Create generalization link (a_3)
S_5	By specialization	Create entity (a_1), Create specialization link (a_4)
S_6	By reference	{Delete link attribute to entity, Create entity, Create association, Link association to entity} (a_8), {Create association, Link association to entity} (a_9)
S_7	By decomposition	{Create association, Link association to entity} (a_9)
S_8	By normalization	{Delete association, Delete Link attribute to association}(a_{10})
S_9	By completeness (assoc.)	Link attribute to association (a_{11})
S_{10}	By completeness (final)	Check the model (a_{12})

4.1 Supervised Learning for MMM

First, we apply supervised learning on activities traces, to estimate the strategies, as explained in section 2. Since, we had the advantage of setting up the experiments, we were able to record students' strategies sequences in addition to their activity sequences (labeling the activities). This allows estimating the strategies by supervised learning. The inputs of the algorithm are the activities traces and

their related strategies. The matrices **E** and **T** can then be computed, and provide the relation between strategies and activities and the transition probabilities between strategies, respectively. Since the learning was supervised, the relations between the strategies and the activities are in line with table 3. However, the transitions between strategies indicate that these latter have not been followed exactly as they were prescribed. By successively applying the Deep Miner algorithm and the Map Miner algorithm to the transition matrix **T**, we extract a Map which emphasizes this phenomenon. This Map is displayed in figure 4.

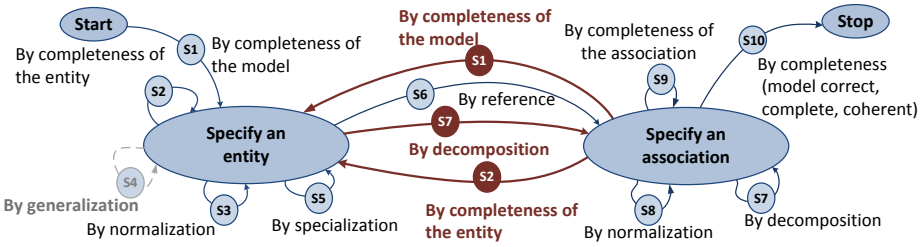

Fig. 4. Map process model obtained by supervised MMM

Therefore, it is possible to detect students' deviations from the prescribed Map. In particular, S_4 is never taken by students and S_7 is chosen in the wrong section. Regarding the strategies S_1 and S_2, they have selected by students from intention *Specify an association* to intention *Specify an entity*. This is not shown in the prescribed Map. However, these transitions are not surprising since they are allowed by the intrinsic semantic of the Map process model. Indeed, a given user can return to an intention already fulfilled to start another section.

4.2 Unsupervised Learning for MMM

By applying unsupervised learning on the activities traces, we discover a different set of strategies and a different Map process model. We recall that in this case, no strategies sequences are necessary as inputs to run the learning algorithm. Consequently, only the traces of activities are necessary used. the discovered strategies are detailed in table 4. In particular, from the emission matrix **E**, we obtain the relation between strategies and activities. Contrary to supervised learning, since no prior information about strategies is available, the names of strategies and intentions are not known. However, based on the names of activities, it is possible to discover the main topics of the strategies. Through a semantic analysis the strategies name can be inferred. As for supervised learning, a Map can be extracted from the estimated transition matrix **T**, it is displayed in figure 5. Except *Start* and *Stop*, two intentions denoted by I'_2 and I'_3 are inferred. The comparison of this Map with the one obtained from supervised learning is made in the next section.

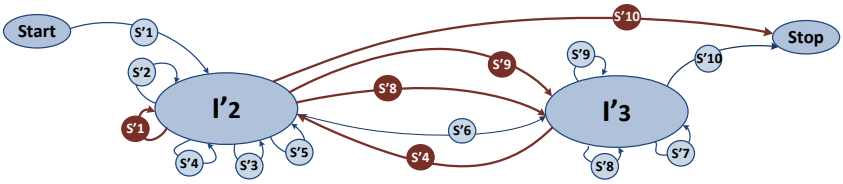

Fig. 5. Map process model obtained by unsupervised MMM

Table 4. Intentions, Strategies and Activities obtained by unsupervised learning for ER diagrams.

Intentions	Code	Activities	Strategies Topics obtained by MMM
Start $\rightarrow I'_2$	S'_1	a_1 (0.94)	entity, creation, specify
I'_2	S'_1	a_1 (0.94)	entity, creation, specify
	S'_2	a_2 (0.88), a_1 (0.09)	attribute, entity, creation
	S'_3	a_2 (0.09), a_3 (0.13), a_4 (0.39), a_5 (0.40)	entity, delete, creation, specialize
	S'_4	a_1 (0.11), a_2 (0.54), a_8 (0.25)	delete, creation, attribute, entity, association
	S'_5	a_5 (0.1), a_6 (0.63), a_7 (0.28)	primary key, creation, entity
$I'_2 \rightarrow I'_3$	S'_6	a_1 (0.15), a_2 (0.79)	creation, entity, attribute, link
	S'_8	a_1 (0.09), a_2 (0.81), a_9 (0.08)	association, entity, link, attribute, creation
	S'_9	a_1 (0.37), a_9 (0.19), a_{11} (0.34)	creation, association, entity, attributes
$I'_3 \rightarrow I'_2$	S'_4	a_1 (0.11), a_2 (0.54), a_8 (0.25)	delete, creation, attribute, entity, association
I'_3	S'_7	a_9 (0.83), a_{10} (0.05), a_{11}(0.05)	link, creation, delete, entity, association
	S'_8	a_1 (0.09), a_2 (0.81), a_9 (0.08)	association, entity, link, attribute, creation
	S_9	a_1 (0.37), a_9 (0.19), a_{11}(0.34)	creation, association, entity, attributes
$I'_3 \rightarrow$ **Stop**	S'_{10}	a_9 (0.08), a_{12} (0.87)	check, model, coherent
$I'_2 \rightarrow$ **Stop**	S'_{10}	a_9 (0.08), a_{12} (0.87)	check, model, coherent

4.3 Discussion and Threats to Validity

It is interesting to discuss the results obtained in previous sections from both quantitative (models likelihood, algorithm convergence, complexity) and qualitative (models interpretation) points of view.

• Adopting a qualitative point of view, for the E/R diagrams experiment, although some strategies from the prescribed Map and the Map obtained by unsupervised learning are similar (S_1 and S'_1, S_2 and S'_2, S_{10} and S'_{10}, S_7 and S'_7), most strategies from unsupervised learning cannot be exactly identified to prescribed strategies. It is not due to a poor compliance of the Map obtained by unsupervised learning but due to the supervised learning assumption, *i.e.* the prescribed Map is actually followed by students. This assumption is not true. Indeed, during the enactment of the process, students may deliberately or accidentally not follow the prescribed Map. Consequently, assuming that the prescribed model is followed by students creates a bias in the definition of strategies and intentions. In addition, there is no ground truth for labeling the activities sequences. Con-

sequently, the labeling could be flawed as it is a subjective process. Moreover, assigning the labels to the strategies and intentions constrains the discovered Map to a limited space which leads to poor performance of supervised learning. This phenomenon can be verified with the deviations of students detected by obtained Maps. Whereas the Map obtained by supervised learning detected only two deviations (S_4 and S_7), the Map obtained by unsupervised learning detected five deviations which are not the same as the supervised learning ones (S'_1, S'_4, S'_8, S'_9, S'_{10}).

• The log-likelihood of the strategies estimated by unsupervised learning in both experiments is higher than supervised learning (see table 5). In other words, the strategies estimated by unsupervised learning have a higher chance to generate the activities observed in the event logs. It makes unsupervised learning more trustworthy than supervised learning. Note that this result is in line with the theoretical study performed in section 3.

Table 5. Practical comparison of supervised and unsupervised learning.

Experiments	**Learning**	**Traces for learning**	**A-priori knowledge**	**Convergence speed (complexity)**	**The estimated parameters Log-likelihood**
E/R diagrams	**Supervised learning**	66 traces of activities, 66 traces of strategies	set of activities, set of strategies	1 iteration	$-2.54e^3$
	Unsupervised learning	66 traces of activities	set of activities, number of strategies	9, 986 iterations	$-2.36e^3$
E-Shopping	**Supervised learning**	90 traces of activities, 90 traces of strategies	set of activities, set of strategies	1 iteration	$-2.81e^3$
	Unsupervised learning	90 traces of activities	set of activities, number of strategies	4, 325 iterations	$-1.69e^3$

• From a cost-benefit and human-centric point of view, cognitive tasks are time-consuming and labor intensive. Since the methodology of labeling activities cannot be generalized to common event logs, this is one serious drawback for supervised learning. Thus, the cost of labeling the data for supervised learning approach is quite high as it involves the students' commitment to label and comment their activities at each step of the process. In comparison, the only humans' effort for unsupervised learning is to choose the number of strategies for the intentional process model. Nevertheless, the unsupervised learning requires a minimal humans' intervention and it allows obtaining intentional process models that match the actual enacted process. The drawbacks of unsupervised learning are a higher computation complexity and the need to automate the naming of

obtained strategies and intentions.

• The BWA cannot be guaranteed to converge to the global maximum likelihood (see section 3.2). The convergence depends on the initialization of the matrices **T** and **E** and it converges at 9,986 learning iterations for E/R diagrams and at 4,325 learning iterations for E-shopping. The supervised algorithm converges in the first iteration.

• While the complexity of the BWA is high due to its requirement to several iterations until the convergence to a local optimum, complexity of supervised learning is very low. Table 5 presents a summary of both learning approaches for tow experiments.

5 Related Work

To the best of our knowledge, there is no work comparing supervised and unsupervised learning for process model discovery from event logs. For this reason we position our work with respect to process mining techniques. Process mining approaches propose to model users' behaviors in terms of activities [2]. These techniques aim at recovering the original workflow from event logs, which containing the traces of processes enactments. Process mining approaches use different algorithms and techniques, such as classification and learning techniques to extract the information from event logs [24–27]. Some of these techniques are investigated hereafter.

Inference methods infer process models with a tradeoff between accuracy and noise robustness. Cook compares in [28] three inference algorithms of RNet [29], Ktail [30] and Markov models [31] for process discovery. The latter two are considered as the most promising approaches. Genetic algorithm [27] provides process models (Petri nets) built on causal matrix, *i.e.*, input and output dependencies for each activity. This technique tackles problems such as noise, incomplete data, non-free-choice constructs, hidden activities, concurrency, and duplicate activities. Nevertheless, it requires the configuration of many parameters to deal with noise and irrelevant data, which is a complex task. Directed acyclic graphs [32] proposes to transform the events into dependency graphs or workflow graphs using directed acyclic graph, representing events and their causal relations without loop. However, using this kind of graphs to model the processes is delicate as loops exist in process models. To tackle this challenge, this approach tries to count the tasks frequencies and then fold the graph. Nevertheless, the results are partially satisfying and the model does not completely fit the actual process. However, all these approaches neglect the underlying humans' cognitive operators such as users' strategies and intentions. In process mining field, HMMs are used in context of inductive workflow acquisition [26] to model a workflow or in [25] as a conformance checking technique. Each state of HMMs corresponds to a task node. The event logs can be observed and generated into workflow nets by inductive learning. This approach supports appearing the same tasks at several times in the model (duplicate tasks). It is similar to

the approach of directed acyclic graphs [32] due to the presence of the splits and joins in the transformation step. However, the works using HMMs do not consider the hidden states as the cognitive states such as intentions/strategies.

6 Conclusions

In this paper, we have compared the supervised and unsupervised learning approaches, in both theoretical and practical contexts, to understand which approach allows discovering underlying users' strategies optimally. Applied on a real dataset to obtain Map process models, the results demonstrate that several issues hinder the application of supervised learning in modeling humans' cognitive process, such as considerable humans' involvement in terms of data labeling, introducing inherent humans' biases and lack of accurate ground truth. Therefore, we deduce from our study that unsupervised learning offers better results than supervised learning to discover intentional process models (Maps).

Although the MMM automatically discovers the topology of the intentional model, the names of strategies and intentions are still inferred semi-automatically. However, the logical relations between activities, strategies and intentions established in the intentional model could be exploited to build an ontology to fully automate the process of inferring the names of strategies and intentions. In addition, we are developing an ProM [33] plug-in which will allow modeling processes in intentional manner. This provides our community to have a vision of processes from an intentional angle.

References

1. Cook, J.E., Wolf, A.L.: Discovering models of software processes from event-based data. ACM TOSEM **7**(3) (1998) 215–249
2. Van der Aalst, W.M., van der Aalst, W.: Process mining: discovery, conformance and enhancement of business processes. Springer (2011)
3. Mirbel, I., Ralyté, J.: Situational method engineering: combining assembly-based and roadmap-driven approaches. Requirements Engineering **11**(1) (2006) 58–78
4. Jarke, M., Pohl, K.: Establishing visions in context: towards a model of requirements processes. In: ICIS. (1993) 23–34
5. Rolland, C., Salinesi, C.: Modeling goals and reasoning with them. In: Engineering and Managing Software Requirements. Springer (2005) 189–217
6. Plihon, V., Rolland, C.: Modelling ways-of-working. In: Advanced Information Systems Engineering, Springer (1995) 126–139
7. Rolland, C., Prakash, N., Benjamen, A.: A multi-model view of process modelling. Requirements Engineering **4**(4) (1999) 169–187
8. Rolland, C.: Modeling the requirements engineering process. In: Information Modelling and Knowledge Bases, Budapest, Hungary, May. (1993) 85–96
9. Thevenet, L.H., Salinesi, C.: Aligning is to organization's strategy: the instal method. In: Advanced Information Systems Engineering, Springer (2007) 203–217
10. Salinesi, C., Rolland, C.: Fitting business models to system functionality exploring the fitness relationship. In: Advanced IS Engineering, Springer (2003) 647–664
11. Rolland, C., Kirsch-Pinheiro, M., Souveyet, C.: An intentional approach to service engineering. Services Computing, IEEE Transactions on **3**(4) (2010) 292–305

12. Rabiner, L.R.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE **77**(2) (1989) 257–286
13. Khodabandelou, G., Hug, C., Deneckere, R., Salinesi, C.: Unsupervised discovery of intentional process models from event logs. Submitted at: The 11th Working Conference on Mining Software Repositories (MSR) (2014)
14. Dardenne, A., Van Lamsweerde, A., Fickas, S.: Goal-directed requirements acquisition. Science of computer programming **20**(1) (1993) 3–50
15. Yu, E.: Modelling strategic relationships for process reengineering. Social Modeling for Requirements Engineering **11** (2011) 2011
16. Soffer, P., Rolland, C.: Combining intention-oriented and state-based process modeling. In: Conceptual Modeling–ER 2005. Springer (2005) 47–62
17. Gray, W.D., John, B.E., Atwood, M.E.: The precis of project ernestine or an overview of a validation of goms. In: Human factors in CS, ACM (1992) 307–312
18. Hayashi, M.: Hidden markov models to identify pilot instrument scanning and attention patterns. In: Systems, Man and Cybernetics, IEEE (2003) 2889–2896
19. Tversky, A., Kahneman, D.: Judgment under uncertainty: Heuristics and biases. science **185**(4157) (1974) 1124–1131
20. Khodabandelou, G., Hug, C., Deneckere, R., Salinesi, C.: Supervised intentional process models discovery using hidden markov models. In: Proceed. of 7th Int. Conf. on Research Challenges in Information Science. (2013)
21. Burnham, K.P., Anderson, D.R.: Model selection and multi-model inference: a practical information-theoretic approach. Springer (2002)
22. Baum, L.E., Petrie, T., Soules, G., Weiss, N.: A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. The annals of mathematical statistics **41**(1) (1970) 164–171
23. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. The Royal Statistical Society (1977) 1–38
24. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2005)
25. Rozinat, A., Veloso, M., van der Aalst, W.M.: Evaluating the quality of discovered process models. In: 2nd WS on the Induction of Process Models, Citeseer (2008)
26. Herbst, J., Karagiannis, D.: Integrating machine learning and workflow management to support acquisition and adaptation of workflow models. In: Proceed. 9th Intl. Workshop on Database and Expert Systems Applications, IEEE (1998)
27. De Medeiros, A.A., Weijters, A.: Genetic process mining. In: Applications and Theory of Petri Nets, Lecture Notes in Computer Science, Citeseer (2005)
28. Cook, J.E., Wolf, A.L.: Automating process discovery through event-data analysis. In: Software Engineering, 1995. ICSE 1995, IEEE (1995) 73–73
29. Das, S., Mozer, M.C.: A unified gradient-descent/clustering architecture for finite state machine induction. In: NIPS, Morgan Kaufmann (1994) 19–26
30. Biermann, A.W., Feldman, J.A.: On the synthesis of finite-state machines from samples of their behavior. Computers, IEEE Transactions on (1972) 592–597
31. Baum, L.E., Petrie, T.: Statistical inference for probabilistic functions of finite state markov chains. The annals of mathematical statistics **37**(6) (1966) 1554–1563
32. Agrawal, R., Gunopulos, D., Leymann, F.: Mining process models from workflow logs. Springer (1998)
33. EUT: Prom. `http://www.processmining.org/prom/start` (November 2013)