Advances in Reliability, Safety and Security, Part 3 – Kołowrocki, Dąbrowska (eds.)

Advances in Reliability, Safety and Security

ESREL 2024
Monograph Book Series

Hidden Markov Model For Remaining Useful Lifetime: Towards A New Strategy For Training Models

Manuel Avila-Gomez[a], Nesrine Khodja[a], Stéphane Bégot[a], Florent Duculty[a], Pascal Vrignat[a], Frédéric Kratz[b]

[a]*Univ. Orléans, PRISME, EA 4229, F45072, Orléans, France*
[b]*INSA-Loire Valley, PRISME, EA 4229, F18020, Bourges, France*

Abstract

Prognostic and Health Management is one of the keys of competitiveness for companies. In this study, we present our approaches, based on Hidden Markov Model, to help maintenance manager to plan preventive actions. HMM can be used to estimate health state of a process. We propose our approach, based on a left-right topology that can assimilate failure signatures. Then according to the mean times before absorption, we propose a Remaining Useful Lifetime estimation. This estimation follows the health state estimation. These methods are illustrated on two different case studies. The last part of this paper deals with multiclass problems where different HMM are in competition, for example different failure modes. In this case, we propose a new strategy for training HMM. Classical approaches only look for features specific to a class (what we are) and this for each class independently. We propose a new Viterbi algorithm (NB Viterbi) that consider specific features of a class (what we are) and differences with other classes (what we are not). We show encouraging results that require further study.

Keywords: Hidden Markov Model, PHM, RUL, Viterbi, training

1. Introduction

Prognostic and Health Management is a topic subject of numerous studies as described in this state of the art (Vrignat et al., 2022). In case of data driven methods, Hidden Markov Models are often used to prevent failure (Robles et al., 2014; Vrignat et al., 2015; Alejandro, 2012), or to propose a prognostic of failure (Aggab, 2021). These estimations can be used to help maintenance manager to plan efficiently preventive actions (Figure 1). A good prognostic provide enough time to plan preventive maintenance action before the failure occurs.

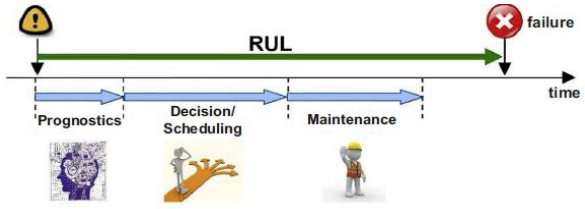

Fig. 1. Efficient planning of preventive actions.

We use Scopus, the scientific database reference, to search papers dealing with HMM and maintenance. We see in Figure 2, produced with VOSviewer, that there are a lot of papers dealing with Remaining Useful Lifetime (RUL) and Hidden Markov Model (HMM). This figure mentions many keywords present in this paper. The scientific community is very active in these fields.

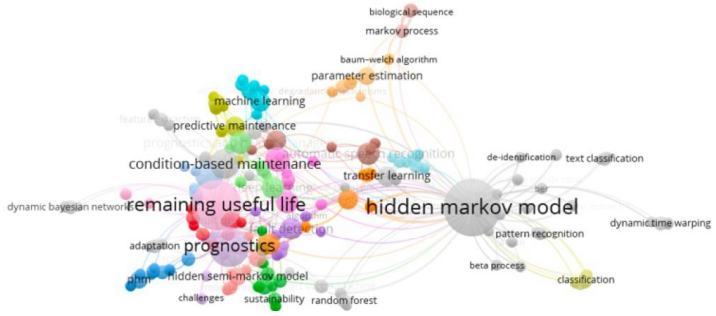

Fig. 2. VosViewere vue of keywords research (HMM & maintenance).

A lot of method, based on Markov Models, are proposed in literature. In this paper, we focus on discrete Hidden Markov Models. As in Figure 3, hidden states of the Markov Models can be considered as health level of a process or a system. The visible part of the model, observations, could be events collected on the process or value provided by sensors after, if necessary, some pre-processing task (Medjaher, 2012). The hidden states can be considered as traffic lights (from green to red) for an industrial process which inform of imminence of failure (when light pass to the orange).

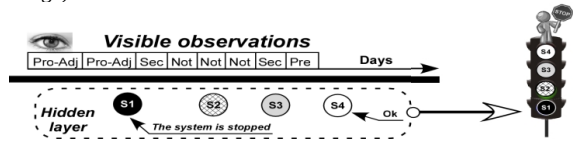

Fig. 3. Visible and hidden layers (system states estimation).

2. Hidden Markov Models for prognostic health maintenance

2.1. Elements of a HMM

A HMM could be described with a stochastic automaton where a set of states is related to observations which can be either discrete or continuous. We focus only on HMM for discrete states and observations. N is the number of states in the model. States are labelled from $S = \{S_1, S_2, \ldots S_N\}$, and the state at time t is q_t. A state sequence is $Q = \{q_1, q_2, \ldots q_T\}$ with T the length of the sequence. M is the number of different observations. Observation symbols are labelled from $V = \{V_1, V_2, \ldots V_M\}$. and the observation at time t is o_t. An observation sequence is

$O = \{o_1, o_2, \ldots o_T\}$.

A is the transition matrix as $A = \{a_{ij}\}$, where a_{ij} is the state-transition probability distribution.

$$a_{ij} = P\left(q_t = S_j \middle| q_{t-1} = S_i\right) \quad (1)$$

B is the observation matrix as $B = \{b_{ij}\}$, where b_{ij} is the observation symbol probability distribution.

$$b_{ij} = P\left(o_t = V_j \middle| q_t = S_i\right) \quad (2)$$

The initial state distribution is $\pi = \{\pi_1 \ldots \pi_N\}$, where:

$$\pi_i = P\left(q_1 = S_i\right) \quad (3)$$

A HMM could be defined by $\lambda = \{A, B, \pi\}$ where A, B and π have to meet the conditions:

$$\sum_{j=1}^{N} a_{ij} = 1, a_{ij} \geq 0, \sum_{j=1}^{M} b_{ij} = 1, b_{ij} \geq 0, \sum_{i=1}^{N} \pi_i = 1, \pi_i \geq 0.$$

The structure of the automaton or topology (Figure 4) is often the result of an empirical choice. But this choice could be made to represent a physical property of the system. For example, a left-right topology could be used to model a written language (for languages that are written from left to right). This topology could be defined to correspond to a level of availability of a system with 4 states (Vrignat et al., 2010) (Figure 5).

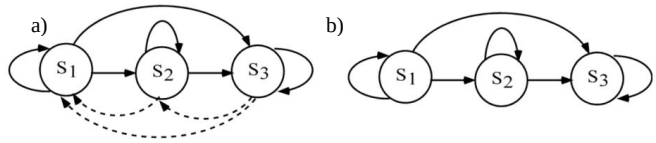

Fig. 4. Example of topology: (a) ergodic; (b) left-right.

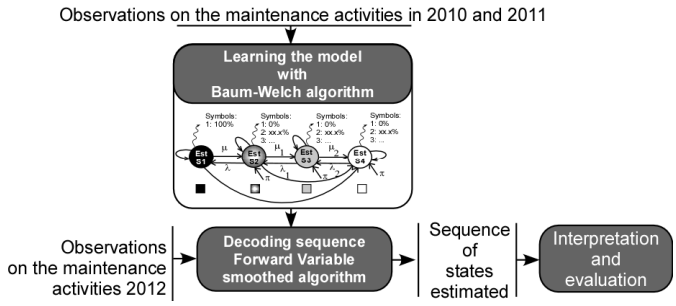

Fig. 5. Strategy used for 4 states degradation level estimation.

As it is described by Rabiner (Rabiner, 1989) in case of speech recognition, usuals algorithms like Baum-Welch or Viterbi can be used to implement Hidden Markov Model. With enough data collected on the system (speech recording, industrial sensors, …), these observations can be used to learn models, i.e. to find the model parameters $\lambda = \{A, B, \pi\}$ which maximize the likelihood of observation sequences according to the model $\wedge$ for the training data set. Then, this model can be used to identify unknown sequences by calculating the likelihood of this sequences with the forward algorithm or by identifying states of the sequence with the Viterbi algorithm. In the example described on the Figure 5, maintenance activities collected or recorded for two years (2010 and 2011) were used to train the model. Activities collected for the year 2012 were used to test capacities of the model for failure prediction.

2.2. Example of use for the diagnostic of a process

In this example, the studied system, is part of a complete line of production of bread (Figure 6). A storing zone, located at the beginning of the line, provide different ingredients that are weighted according to the cooking recipe. Then, these products are mixed. The pancake mixture can be weighted with accuracy to form bread or others bakery products. This is the part studied in this example.

After this stage, dough balls are placed in molds that are sent in an oven for cooking. Then bread is unmolded and cooled on a conveyor belt tour. Finally, the bakery product is prepared to be send with controls and conditioning, with an adequate packaging.

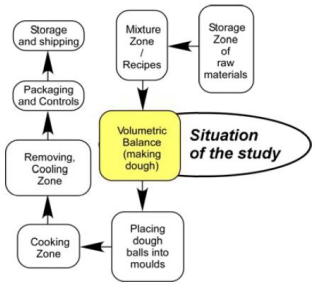

Fig. 6. Volumetric balance in the center of line of production of bread.

Events corresponding to maintenance activities were recorded on the volumetric balance (Table 1). In this example, maintenance agents consign their actions and observations in a centralized database. Symbols "TEP, AU, DEP..." characterize maintenance activities carried out on this process. "DEP" symbol corresponds to a failure: the balance system is stopped. It is a critical condition that we try to minimize.

The list of these actions is given in Table 2 and coded using a set of 10 symbols (1 to 10). Symbol 1 (red) means that the system is stopped.

Table 1. Example of recorded events

Name	Date	Ope.	Cd	IT	N°	Symbol
Dupond	11/01/2007	Lubrication	TEP	20	1	9
Dupond	11/01/2007	Lubrication	TEP	20	2	9
Dupond	12/01/2007	Lubrication	SEC	30	3	5
Dupond	12/01/2007	Lubrication	TEP	30	4	5
Dupond	13/01/2007	Padlock	TEP	10	5	6
Dupond	13/01/2007	Padlock	RAS	30	6	5
Dupond	13/01/2007	Padlock	RAS	30	7	5
Dupond	16/01/2007	Lubrication	DEP	90	8	1
Dupond	19/01/2007	Padlock	AU	10	9	3

Table 2. Symbolic coding system of maintenance interventions.

Process state	
RUN	
STOP	

	Interventions type
1	DEP (Troubleshooting / Stop Production)
2	RM (Setting Machine)
3	AU (Other)
4	OBS (Observation)
5	TEP (Preventive Maintenance, Production not stopped)
6	SEC (Security)
7	RAN (Planified Upgrading)
8	NET (Cleaning Machine)
9	VEP (Preventive Maintenance Visit)
10	RAS (Nothing to report)

About 2 years of recorded history made with events collected on the balance are used in the training phase. The Baum-Welch algorithm is used to enhance probability to generate observation sequences of the training dataset considering the model $P(O|\lambda)$. In this training phase, we consider that the system starts with a high health level (Blue state). We consider that end of sequences are failure situations (stop production: DEP) and that the system is in stop state (lowest health level: Red state). In this training phase, we align start and end of sequences with respectively states S_4 and S_1, the other events are "assimilated" by the topology of the HMM (Figure 7a).

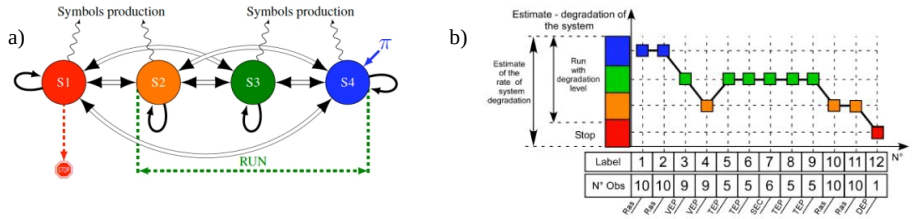

Fig. 7. Degradation of a process.

About 1 year has been used to evaluate capacity of the model to provide failure prediction. All observation sequences were computed with the estimated model. For each sequence of observation, a state sequence was provided by the Viterbi algorithm (Figure 7b). Then, an analysis of the S_2 state was made to evaluate the capacity of this state to announce the failure (Table 3). The S_2 state need to be observe before the failure, but not to early (many days before failure) and not to late (to enable planification of maintenance).

Table 3. Failure prediction on test database.

Number of days before failure in S_2 state (Orange Light)			
1 day	2 days	3 days	7 days
28,6%	71,4%	85,7%	100%

2.3. Estimation of the RUL with HMM

The RUL estimation is made by using the Viterbi decoding algorithm and the mean times before absorption. These mean times estimation and the calculation of the RUL are described below. Mean times before absorption could be calculated analytically from the transition matrix A. We sort the states to place first the non-absorbing

states ($q = N - 1$), then the absorbing states, considering that in this degradation model, the absorbing state is assigned to highest degradation (failure of the system). Thus, the transition matrix A takes the following canonical form:

$$A = \begin{pmatrix} Q & R \\ 0 & 1 \end{pmatrix} \tag{4}$$

where Q is a matrix of size $q \times q$, R is a matrix of size $q \times 1$ and 0 denotes the zero matrix of size $1 \times q$. By recurrence, we write:

$$A^n = \begin{pmatrix} Q^n & [I + Q + \cdots + Q^{n-1}]R \\ 0 & 1 \end{pmatrix} \tag{5}$$

and $lim_{n\to\infty} Q^n = 0$, so

$$lim_{n\to\infty} A^n = \begin{pmatrix} 0 & FR \\ 0 & 1 \end{pmatrix}. \tag{6}$$

The matrix $F = [I - Q]^{-1}$ is called the fundamental matrix of the model. f_{ij} value of the matrix F is the mean number of courses from states i to j. For the variable τ, giving time until absorption from the state i, we have:

$$\mathbb{E}_i(\tau) = c * \sum_{j=1}^{q} f_{ij} \tag{7}$$

with c the sampling refresh time.

Once the mean times to absorption have been calculated, we assess the level of degradation reached by the system using the Viterbi algorithm.

Considering history collected on a process, the HMM parameters can be trained to form the degradation model $\lambda = \{A, B, \pi\}$ of the process (like in part 2.1). In the monitoring phase, the sequence of hidden states $Q^* = q_1{}^*, q_2{}^*, \ldots . q_T{}^*$ which probably engendered the sequence of observations O is estimated with the Viterbi algorithm. This calculation consists in solving:

$$q_T{}^* = \arg\ max_{i=1:N}\ P\,(O_{1:T}, q_{T=}\, Q_i/\lambda) \tag{8}$$

The level of degradation is the final or most persistent level from past states. Knowing the level of degradation and the mean time to absorption for this state, the RUL is directly estimated by:

$$RUL(t) = \mathbb{E}_l(\tau) - t_l \tag{9}$$

where $l \in \{Q_i \ldots Q_N\}$ and t_l are respectively the current state (level) and the time spent in this state, which is calculated from the state sequence Q^*.

For the illustration of the RUL, we implement the method on a system of level control on a double tank. All details on this study is in (Aggab et al., 2021). The figure 8 shows the double tank level control system. Water is injected into the tank 1 with a cross-section S_1 by a pump motor. Then, the tank 2 is supplied by the valve 1 with a cross-sectional area S_2. The level of water in the tank 2 is monitored by the system (controlled measurement). This level is provided by a sensor and controlled by adjusting the pump motor power which is calculated by a PID controller.

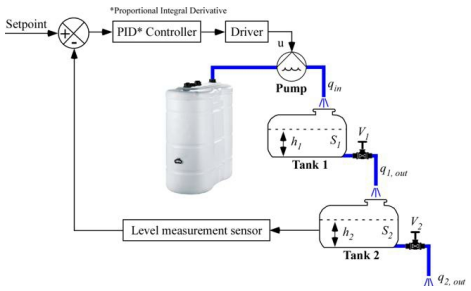

Fig. 8. A double-tank level control system.

We have simulated the system considering that the pump could degrade during time. Shocks occur according to a homogeneous Poisson process and the pump capacity decreases. By its structure (loop system), compensation can be done by PID controller until a limit when pump capacity is under the needs. To provide events to the HMM, a generation of residue is used (Figure 9). Several sequences are generated by simulation

with randomly curves of pump degradation to form the training dataset. KNN clustering is used with Residues to provide observations sequences.

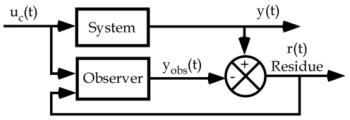

Fig. 9. Generation of residues by the observer.

A four states HMM is trained with this dataset. New sequences are generated to test the approach. The figure 10 shows a sequence of observations and the health state estimation. In the figure, we see the system failure (when pump capacity is under the needs) and the pump failure (capacity = 0). The health state estimation provides the same results as in the example (2.1).

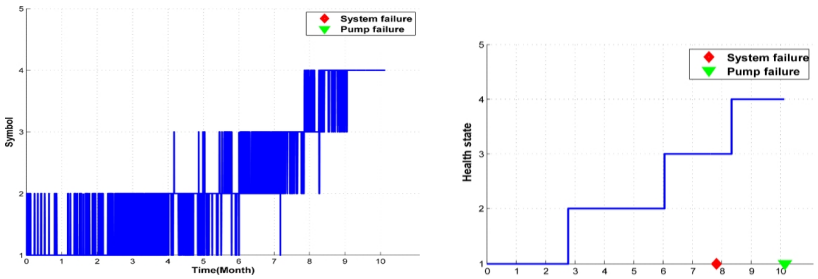

Fig. 10. Sequence of symbols and health state estimation.

Implementing the RUL as described in this part, we obtain results shown in Figure 11. We see that at each state shift, the RUL is re-estimated. The figure shows the pertinence of this prognostic.

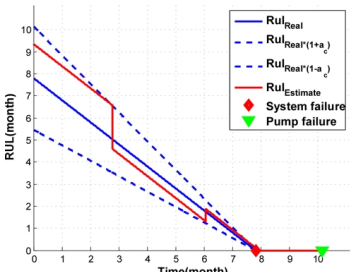

Fig. 11. RUL estimation for the example.

3. Focus on multi-classes problem

In this part, we consider that the system could be subject of various mode of failure. We suppose that we dispose of return of experience for these failures as recorded events for each kind of failure or for absence of failure.

These modes could be considered as a multi-class problem with samples recorded for each class (Figure 12). In usual HMM classification, like examples of part 2, only one model is trained for failure diagnostic. In this part, we consider different classes, one for the non-failure mode and the others for different failure modes. Then, one model is trained for each class.

In this study, we will consider 4 classes for illustration of the method. We consider that observation sequences are denoted $O^r = \{O_1^r, O_2^r \ldots\ldots, O_L^r,\}$ with r ∈[1, 4]. These classes are illustrated in Figure 12. In this part, after explaining our synthetic data production step, we show how HMMs are used to classify failure mode. These results will be used as reference for the contribution proposed in part 4 for a new approach for HMM training.

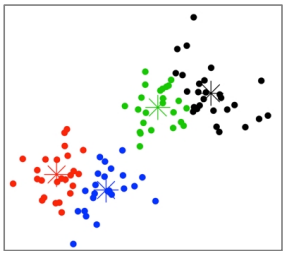

Fig. 12. Multi class problem.

3.1. Synthetic data production for tests

In order to provide datasets to validate our approach, we use HMM Matlab tools to produce data. We chose a left-right topology with 4 states (N=4) and 16 symbols (M=16). Four set of model parameters λr were computed randomly. For each model, observation sequences with 20 symbols (T=20) were generated. 1000 sequences were generated for each model and then split to form the train dataset and the test dataset (70% for training and 30% for testing). For example for the model 1: $O^1_{train} = \{O^1_{train\ 1}, O^1_{train\ 2} \ldots\ldots, O^1_{train\ L1}\}$, for training and $O^1_{test} = \{O^1_{test\ 1}, O^1_{test\ 2} \ldots\ldots, O^1_{test\ L2}\}$ for testing (L1=700, L2=300). With this strategy, several tests can be performed to verify pertinence of the approach. With synthetic data production, we have also access to state sequence that produces observation sequences.

The different steps of the data production can be summarized as follows:

- *Step 1*: Random selection of models
  The first step consists in generating 4 λr models with Matlab. These models represent 4 different classes.
- *Step 2*: Data generation
  Synthetic observations are generated from these 4 models, by defining a number of observations sequences *O*, of length *T*.
- *Step 3*: Training and test corpus
  These observations are divided into two corpora: training and test data. We have opted for 2/3 of the data used for model training and 1/3 for test data.

3.2. Finding Failure Mode.

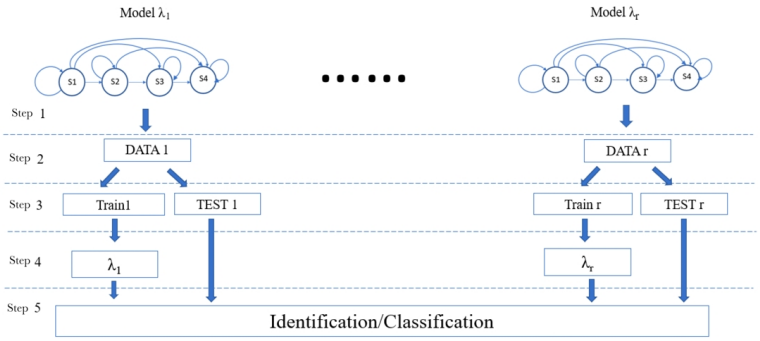

Fig. 13. Usual classification by HMM.

Figure 13 describe the methodology used to find a failure mode. The 3first steps are described in part 3.1. The step 4 consist in training models for each class with the corresponding dataset $O^r_{train} = \{O^r_{train\ 1}, O^r_{train\ 2} \ldots\ldots, O^r_{train\ L1}\}$ with the Baum-Welch algorithm or the Viterbi training algorithm.

Classification is made in step 5, on the test datasets (each O^r_{test}) calculating likelihood of each "unknown" sequences with the 4 models we have trained. The higher likelihood identifies the class.

3.3. Finding non-failure mode

In this part, we consider that a model is associated to a "non-model" (Figure 14). The non-model is a class formed with samples of the other classes. The, using the same strategy as in part 3.2, but using non-class dataset for training, we trained 4 "non-models" (Figure 15). Then, for the same test database of part 3.2, we try to identify "non-classes" of "unknown" sequences. But, in this case, we look for the lowest likelihood to identify the class. We learn "what we are not" (non-model), so we look for "what we are less similar".

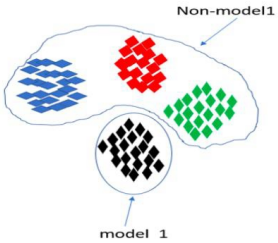

Fig. 14. Model and non-model principle.

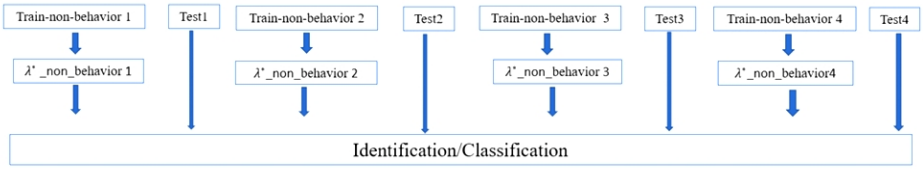

Fig. 15. Classification method by non-model.

Table 4 gives recognition rates for models and non-models using Baum-Welch or Viterbi algorithm for training. We see that the non-models have correct capacities to identify "unknown" sequences (near 70% for Baum-Welch or Viterbi).

In the next part, we propose a new strategy to combine these twice kind of information to train the models.

Table 4. Results of Models and Non-Models classification.

Algorithms	Models Results	Non-Models Results
Baum-Welch	81,47%	70,75%
Viterbi	71,52%	68%

4. A new approach for Multi-Classes training

We propose, in this paper, a new version of the Viterbi training algorithm. Its kind of implementation made possible to propose this new strategy. We first recall the usual implementation.

4.1. Viterbi training algorithm

Algorithm 1. Viterbi Training

Training DATA

$O_{train} = \{O_{train\ 1}, O_{train\ 2}, \ldots O_{train\ L}\}$

Choose an initial HMM setting $\lambda = (A\ B, \pi)$

Repeat

Initialize all variables: n_i, $n_{i\to j}$ *and* $n_{i\omega}$ *to 0*

For *each* $O_{train\ l} \in O_{train}$ *do*

Calculate the Viterbi path for $O_{train\ l}$

Increment the variables n_i, $n_{i\to j}$ *and* $n_{i\omega}$ *according to* $O_{train\ l}$

End for

Estimate new parameters $\lambda = (A\ B, \pi)$

Until parameters stability.

The Viterbi training algorithm (Algorithm 1) uses best path (computed with the Viterbi algorithm) of each sequence of the training dataset to count each sequence of 2 states (ie S_i to S_j) as illustrated in the Figure 16 and then in the Table 5.

The Figure 16 shows best path for 2 sequences and in the Table 5 we show the sums or contribution of each sequence of 2 states.

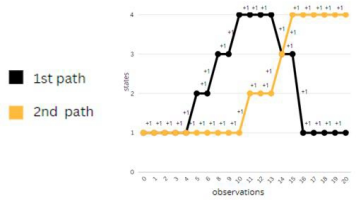

Fig. 16. Viterbi paths for 2 sequences of training database

Table 5. Evolution of $n_{i\to j}$ for the 2 paths.

$n_{i\to j}$	1	2	3	4
1	+4 +10	+1 +1	0	0
2	0	+1+2	+1 +1	0
3	1	0	+2	+1
4	0	0	+1	+3 + 5

The principle of the Viterbi training algorithm is to increment counters for each sequence of 2 states that have been identified to enhance the corresponding probability in the model. We propose to do the same with the non-model database, but, in this case, we want to reduce the corresponding probability. To do this, we use sequences of 2 states to decrement counters.

The size of the non-model database being higher than the model one, we weight the value of decrement.

The Figure 17 shows best path for 2 sequences of non-model database and in the Table 6 we show the weighted decrementation contribution of each sequence of 2 states.

Algorithm 2. Non-Behavior Viterbi Training

Training DATA

$O_{train} = \{O_{train\ 1}, O_{train\ 2}, \ldots O_{train\ L}\}$

$O_{train\_nb} = \{O_{train\_nb\ 1}, O_{train\_nb\ 2}, \ldots O_{train\_nb\ L}\}$

Choose an initial HMM setting $\lambda = (A\ B, \pi)$

Repeat

Initialize all variables: n_i, $n_{i\to j}$ *and* $n_{i\varnothing}$ *to 0*

For *each* $O_{train\ l} \in O_{train}$ *do*

Calculate the Viterbi path for $O_{train\ l}$

Increment the variables n_i, $n_{i\to j}$ *and* $n_{i\varnothing}$ *according to* $O_{train\ l}$

End for

For *each* $O_{train\_nb\ l} \in O_{train\_nb}$ *do*

Calculate the Viterbi path for $O_{train\_nb\ l}$

Decrement the variables n_i, $n_{i\to j}$ *and* $n_{i\varnothing}$ *according to* $O_{train\_nb\ l}$

End for

Add offsets to avoid negative elements in n_i, $n_{i\to j}$ *or* $n_{i\varnothing}$

Estimate new parameters $\lambda = (A\ B, \pi)$

Until parameters stability.

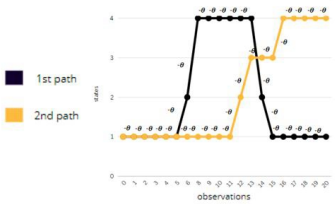

Fig. 17. Viterbi paths for 2 sequences of NB training database.

Table 6. Evolution of $n_{i\to j}$ for the 2 NB paths.

$n_{i\to j}$	1	2	3	4
1	-5θ -10θ	-θ -θ	0	0
2	-θ	0	-θ	-θ
3	0	0	-θ	-θ
4	0	-θ	0	-5θ -4θ

The new Viterbi training algorithm, Non-Behavior Viterbi (NB Viterbi), is detailed in Algorithm 2. To prevent negative values due to decrementation, offsets could be added in each variables n_i, $n_{i\to j}$ *and* .

In Table 7, we show results of this new algorithm. We pass from 71,52% for the Viterbi training algorithm to 79,55% for the NB Viterbi algorithm, 8% better. These results are under whose provided by the Baum-Welch algorithm, but we show that considering "non-classes" information could really improve training phase.

Table 7. Results of classification by NB Viterbi training algorithm.

Algorithms	Models Results
Baum-Welch	81,47%
Viterbi	71,52%
NB Viterbi	**79,55%**

5. Conclusion

In this paper, we present our approaches for RUL estimation based on HMM. In the first part, we show how to use a left right topology to "capture" failure signature. The method is based on training a set of recorded sequences to assimilate variation of events observed in case of failure. In a second part, we use the mean times spent in states or before absorption to estimate the Remaining Useful Lifetime of a system. Then we pass from a diagnostic of health state to a prognostic of RUL.

In the last part, we propose a new strategy for training HMM. This method is adapted when we need to identify a specific mode according to many possible: a multiclass problem. A new training Viterbi algorithm is proposed to consider all disponible data. Classical HMM approaches only use their own data for training their model. We proposed an algorithm that can enhance features of the class by considering difference with the other classes. The Viterbi training has been chosen, in this first step, because of its "simple" structure that can be adapted for training non-classes. We can then validate the interest of using information contained in "non-classes".

In future work, we would try to transpose this principle with the Baum-Welch algorithm.

References

Alejandro, D., Mejia, T., Medjaher, K., Zerhouni, N., Tripot, G. 2012. A Data-Driven Failure Prognostics Method Based on Mixture of Gaussians Hidden Markov Models, IEEE Transactions on Reliability 61(2), 491-503.

Aggab, T., Vrignat, P., Avila, M., Kratz, F. 2021. Remaining useful life estimation based on the joint use of an observer and a hidden Markov model. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, SAGE Publications, doi: 10.1177/1748006X211044343

Medjaher, K., Tobon-Mejia, D.A., Zerhouni, N. 2012. Remaining Useful Life Estimation of Critical Components With Application to Bearings, IEEE Transactions on Reliability 61(2), 292-302.

Rabiner, L.R. A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE *77*, 1989, 257–286.

Roblès, B., Avila, M., Duculty, F., Vrignat, P., Bégot, S., Kratz, F. 2014. Hidden Markov model framework for industrial maintenance activities, Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability 228(3), 230-242.

Vrignat, P., Kratz, F., Avila, M., 2022. Sustainable manufacturing, maintenance policies, prognostics and health management: A literature review, Reliability Engineering and System Safety 218,108140, 10.1016/j.ress.2021.108140

Vrignat, P., Avila, M., Duculty, F., Kratz, F. 2015. Failure Event Prediction Using Hidden Markov Model Approaches, IEEE Transactions on Reliability 99, 1-11.

Vrignat, P., Avila, M., Duculty, F., Kratz, F. 2010. Towards a Maintenance and Servicing Indicator, Advances in Production Management Systems. New Challenges, New Approaches 338, 113-120, Burbidge Award for the Best Presentation.