LECTURES 7 & 8
Intelligent collaboration

Andrea Zanchettin

Agenda

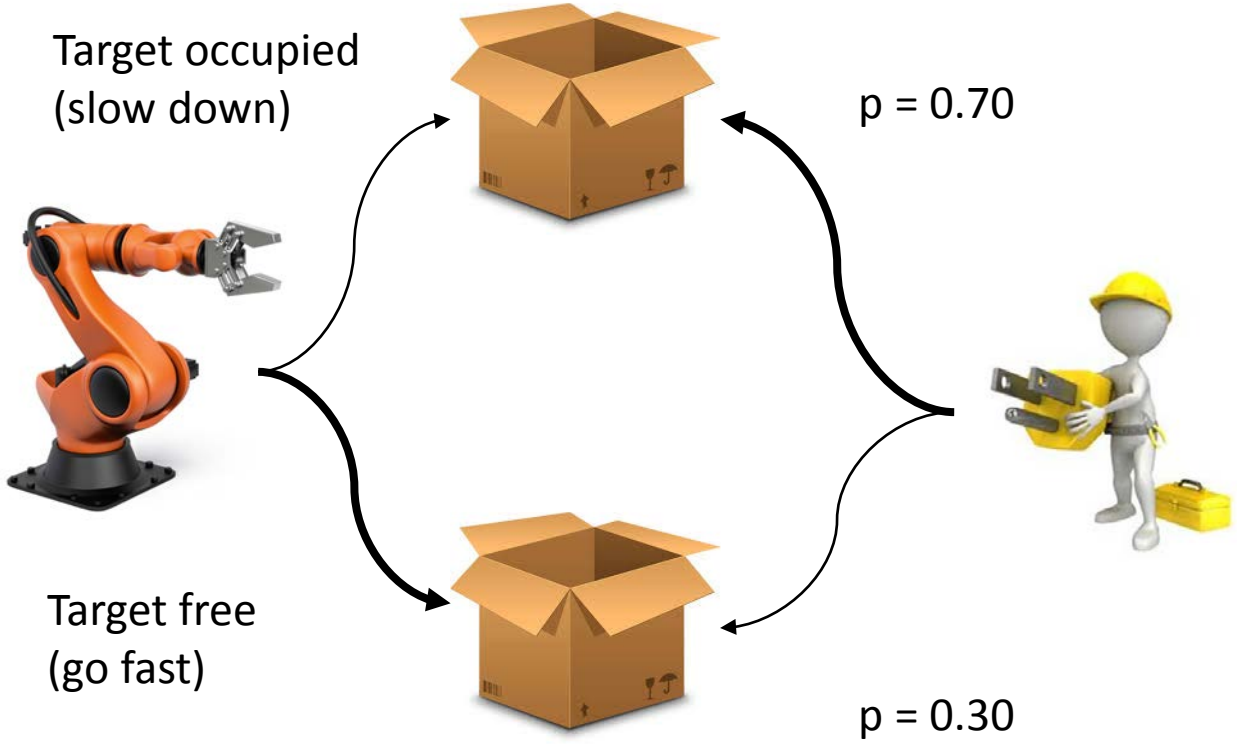
- Introduction and motivations

- Classification problems
  - Generalities
  - Examples: human reaching, human walking

- Prediction problems
  - Generalities
  - Continuous prediction problems
    - Examples: handling assistance, estimated time of arrival
  - Discrete prediction problems
    - Example: activity prediction

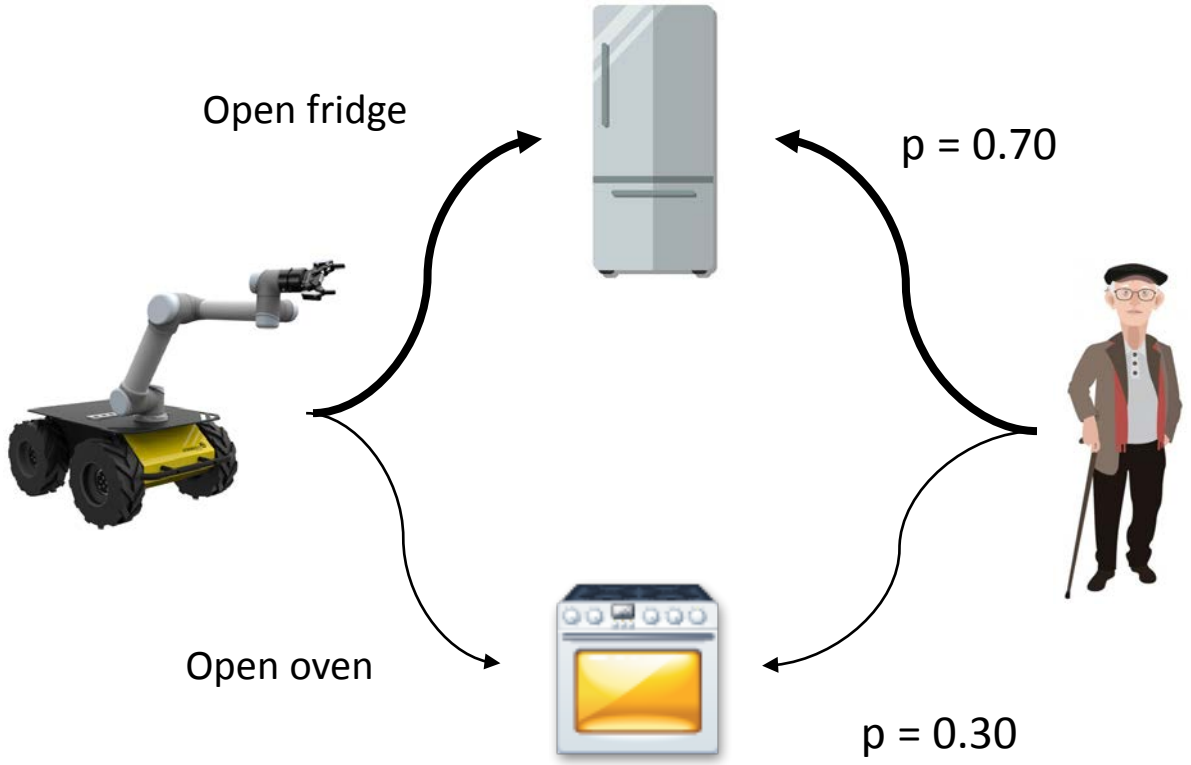
Introduction and motivations

Estimate the most likely target for **minimum interference** task scheduling.



Introduction and motivations

Estimate the most likely target for **prompt assistance**.



Agenda

- Introduction and motivations
- Classification problems
  - Generalities
  - Examples: human reaching, human walking
- Prediction problems
  - Generalities
  - Continuous prediction problems
    - Examples: handling assistance, estimated time of arrival
  - Discrete prediction problems
    - Example: activity prediction

Classification problems - generalities

Classification is adopted in many fields of science:

- object classification
  given a picture, is this a cat or a dog?
- fault detection and predictive maintenance
  given the data, is the machine faulty?
- quality control
  given a picture of a product, does it pass the quality requirements?
- medicine
  given the picture of a cell, is this a tumoral cell?
- computer science
  given the content of an email, is that spam/phishing?
- automotive
  given the data, did a crash happen?
  given the data, is the driver watching his/her phone?

Classification problems - generalities

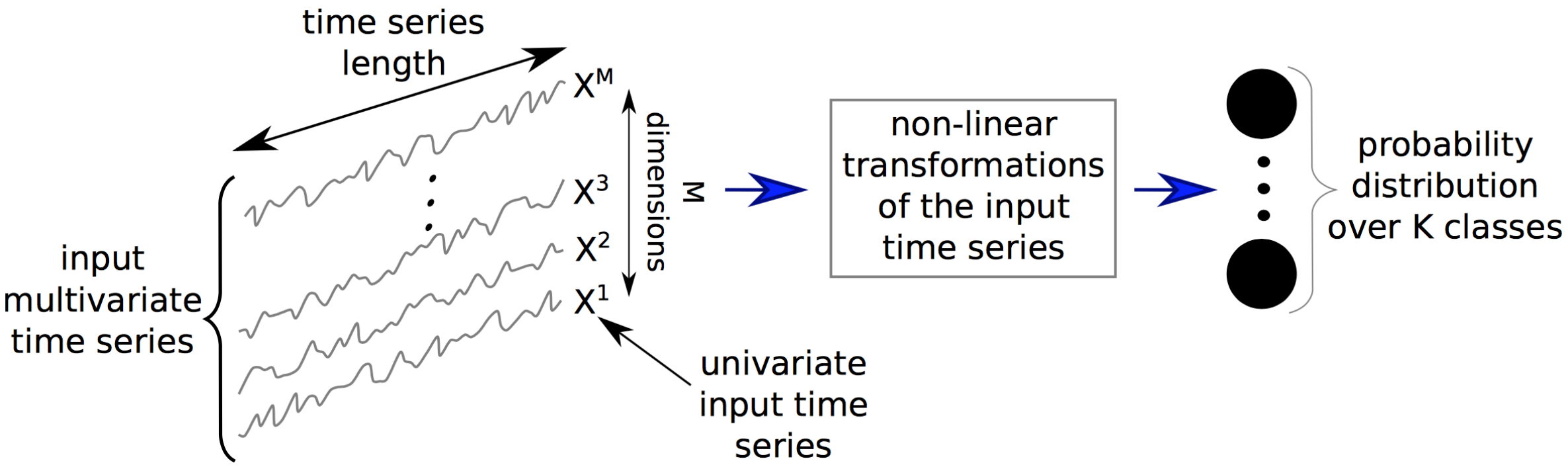
Classification is a process related to categorization, i.e. objects or other entities are assigned to a single class (within a finite set).

Given an observation (object, data, etc.), classification can be:

- deterministic: a deterministic classifier assigns the observation to the most likely class.
- probabilistic: a probabilistic classifier assigns a probability distribution over a set of classes

We will focus on the second kind (probabilistic classifiers).

Classification problems - generalities



Ismail Fawaz, H., Forestier, G., Weber, J. et al. Data Min Knowl Disc (2019)

Classification problems - generalities

Classification (as compared to clustering) is a supervised technique and is usually handled in two separate phases:

- if the models are not available, use labelled data (training examples) to learn them
- once the models are given (either because they have been trained, or because they were already available) use them to predict the class (or the distribution of the classes)

Classification problems - generalities

Few properties of probabilities:

- Chain rule: $P(A, B) = P(A|B)\,P(B)$
- Bayes rule: $P(A|B)\,P(B) = P(B|A)\,P(A)$
- Certainty: $\sum_{A \in \mathcal{A}} P(A) = 1$

Classification problems - generalities

In our setting, a probabilistic classification can be made based on

$$P\left(m_i \middle| y_0, \ldots, y_k\right)$$

i.e. the probability of the observed temporal series to belong to class m_i (to be consistent to the corresponding model).

We can apply the Bayes rule:

$$P\left(m_i \middle| y_0, \ldots, y_k\right) = \frac{P\left(y_0, \ldots, y_k \middle| m_i\right) P\left(m_i\right)}{P\left(y_0, \ldots, y_k\right)}$$

and apply some algebraic manipulations:

$$P\left(m_i \middle| y_0, \dots y_k\right) = \frac{P\left(y_0, \dots, y_k \middle| m_i\right) P\left(m_i\right)}{P\left(y_0, \dots, y_k\right)}$$

Chain rule

$$= \frac{P\left(y_0, \dots, y_{k-1} \middle| y_k, m_i\right) P\left(y_k \middle| m_i\right) P\left(m_i\right)}{P\left(y_0, \dots, y_k\right)}$$

Bayes rule

$$= \frac{P\left(y_k, m_i \middle| y_0, \dots, y_{k-1}\right) P\left(y_0, \dots, y_{k-1}\right) P\left(y_k, m_i\right)}{P\left(y_k, m_i\right) P\left(y_0, \dots, y_k\right)} =$$

Simplification and chain rule

$$= \frac{P\left(y_k \middle| m_i, y_0, \dots, y_{k-1}\right) P\left(m_i \middle| y_0, \dots, y_{k-1}\right) P\left(y_0, \dots, y_{k-1}\right)}{P\left(y_0, \dots, y_k\right)}$$

Classification problems - generalities

$$P\left(m_i \middle| y_0, \ldots, y_k\right) = \frac{P\left(y_k \middle| m_i, y_0, \ldots, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right) P\left(y_0, \ldots, y_{k-1}\right)}{P\left(y_0, \ldots, y_k\right)}$$

Chain rule

$$= \frac{P\left(y_k \middle| m_i, y_0, \ldots, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right) P\left(y_0, \ldots, y_{k-1}\right)}{P\left(y_k \middle| y_0, \ldots, y_{k-1}\right) P\left(y_0, \ldots, y_{k-1}\right)}$$

Simplification

$$= \frac{P\left(y_k \middle| m_i, y_0, \ldots, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}{P\left(y_k \middle| y_0, \ldots, y_{k-1}\right)}$$

We will apply the Markov's hypothesis $P\left(y_k \middle| y_0, \ldots, y_{k-1}\right) = P\left(y_k \middle| y_{k-1}\right)$ that simply states that the observation should be **informative enough** (e.g. a state vector).

We finally have

Classification after k samples

Classification after k-1 samples

$$\boxed{P\left(m_i \middle| y_0, \ldots, y_k\right)} = \frac{P\left(y_k \middle| m_i, y_{k-1}\right) \boxed{P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}}{P\left(y_k \middle| y_{k-1}\right)}$$

Prediction model

Does not depend on classification hypothesis

Classification problems - generalities

Also notice that since $\sum_{m_i \in \mathcal{M}} P(m_i | y_0, \ldots, y_k) = 1$, substituting we have

$$\sum_{m_i \in \mathcal{M}} P(y_k | m_i, y_{k-1}) P(m_i | y_0, \ldots, y_{k-1}) = P(y_k | y_{k-1})$$

So finally:

Classification after k-1 samples

$$\boxed{P(m_i | y_0, \ldots, y_k)} = \frac{P(y_k | m_i, y_{k-1}) \boxed{P(m_i | y_0, \ldots, y_{k-1})}}{\sum_{m_i \in \mathcal{M}} P(y_k | m_i, y_{k-1}) P(m_i | y_0, \ldots, y_{k-1})}$$

Classification after k samples

Prediction model

Normalization

Classification problems - generalities

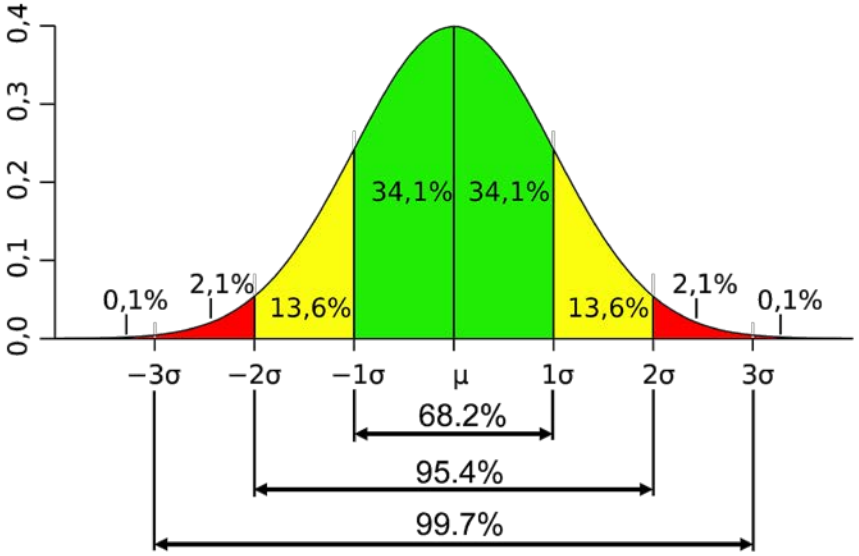
$$P\left(m_i \middle| y_0, \ldots, y_k\right) = \frac{P\left(y_k \middle| m_i, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}{\sum_{m_i \in \mathcal{M}} P\left(y_k \middle| m_i, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}$$

Some remark:

- as the quantity $P\left(y_k \middle| m_i, y_{k-1}\right)$ is both at the numerator and at the denominator, it does not have to be necessarily a probability.
- practitioners usually adopts functions whose values are just indicative of how the measured data matches the model

For example:

$$P\left(m_i \middle| y_0, \ldots, y_k\right) = \frac{f\left(y_k - \hat{y}_{k|k-1,m_i}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}{\sum_{m_i \in \mathcal{M}} f\left(y_k - \hat{y}_{k|k-1,m_i}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}$$



$$f\left(y_k - \hat{y}_{k|k-1,m_i}\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(y_k - \hat{y}_{k|k-1,m_i}\right)^2}{2\sigma^2}\right)$$

Predicted value

Classification problems - generalities

Moreover, one typically defines *M* models but accounts for *M+1* classes.

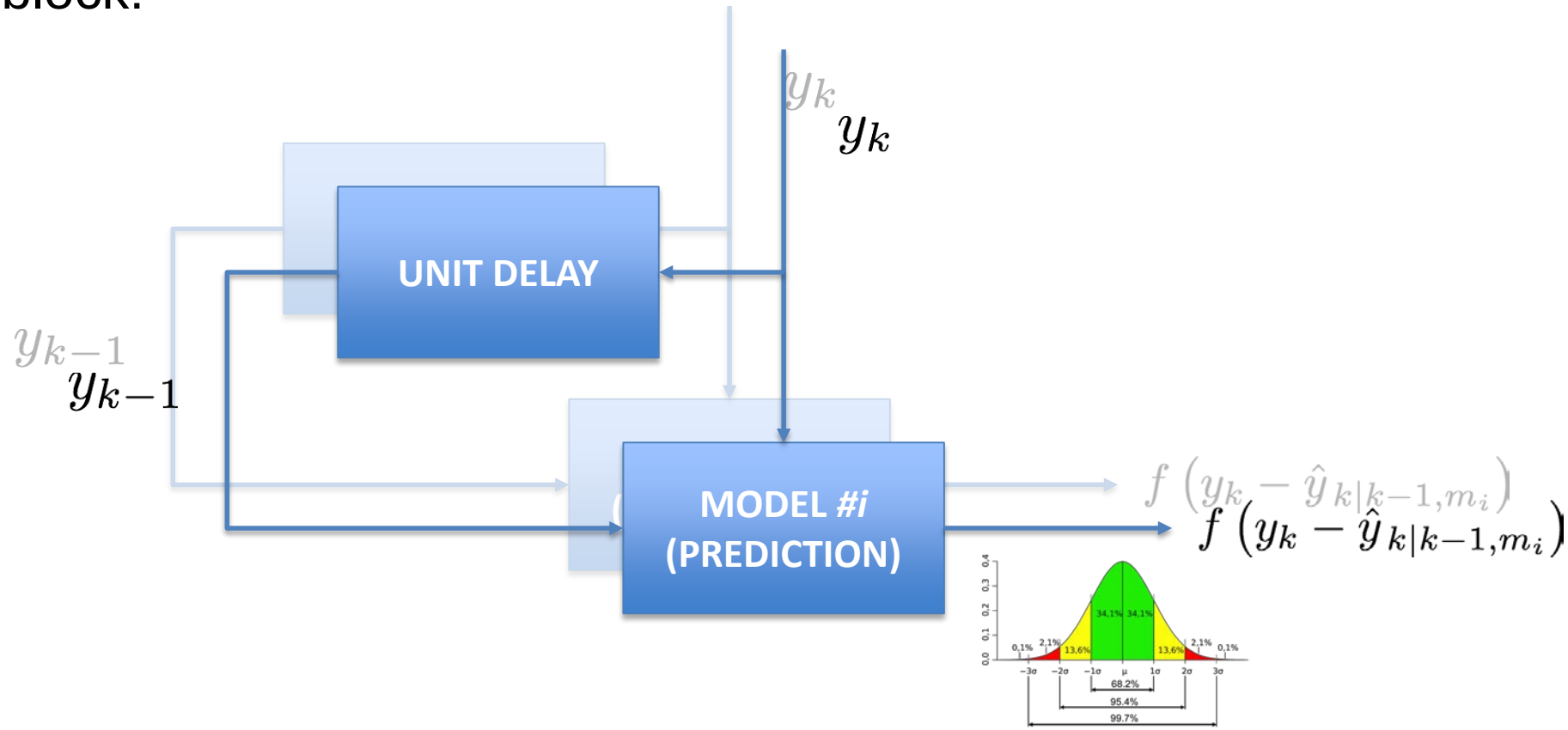
The last class simply indicates that none of the models are suited to describe the data.

In this case, we have:

$$P\left(m_i \middle| y_0, \ldots, y_k\right) = \frac{P\left(y_k \middle| m_i, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)}{1 - \max\left(1, \sum_{m_i \in \mathcal{M}} P\left(y_k \middle| m_i, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)\right)}$$

$$P\left(m_\phi \middle| y_0, \ldots, y_k\right) = 1 - \sum_{m_i \in \mathcal{M}} P\left(y_k \middle| m_i, y_{k-1}\right) P\left(m_i \middle| y_0, \ldots, y_{k-1}\right)$$

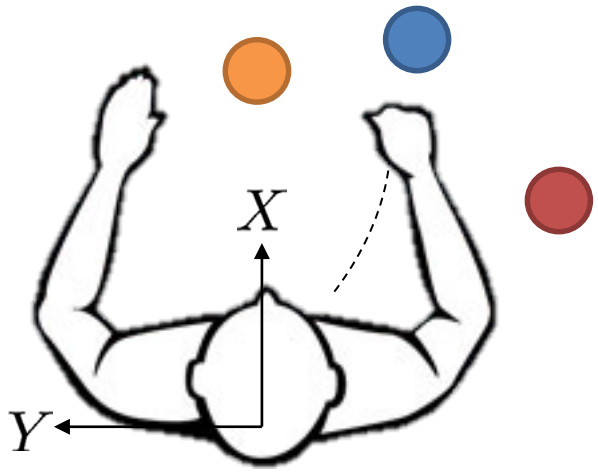
Building block:



Classification problems - reaching path

Problem formulation:

given an observation of the human hand, we want to infer which area he/she is going to reach.



What is an observation?

an observation is the set $y = \begin{bmatrix} x_p & y_p & x'_p & y'_p \end{bmatrix}^T$

The **prediction model** requires more details. First, define $\boldsymbol{p} = \begin{bmatrix} x_p & y_p \end{bmatrix}^T$.

Classification problems - reaching path

The model of human arm reaching path is modelled in an implicit way by defining:

a **dynamical system** representing the human reaching motion

$$\boldsymbol{p}'' = \boldsymbol{u}$$

a **cost functional** to be minimized

$$\min_{\boldsymbol{u}(s)} \int_0^L \|\boldsymbol{u}\|^2 \, ds$$

The solution of the constrained optimal control problem can be written in closed-form.

Zanchettin, A.M., Rocco, P. IROS (2017)

Classification problems - reaching path

The prediction model is as follows:

$$\hat{\boldsymbol{p}}_{k|k-1,m_i} = \sum_{j=0}^{3} \boldsymbol{a}_{j,m_i} \left(s_k - s_{k-1}\right)^j$$

where

$$\begin{aligned}
\boldsymbol{a}_{0,m_i} &= \boldsymbol{p}_{k-1} \\
\boldsymbol{a}_{1,m_i} &= \boldsymbol{p}'_{k-1} \\
\boldsymbol{a}_{2,m_i} &= 1.5\left(\boldsymbol{p}_{m_i} - \boldsymbol{p}_{k-1} - \boldsymbol{p}'_{k-1}\right) \\
\boldsymbol{a}_{3,m_i} &= 0.5\left(\boldsymbol{p}_{k-1} + \boldsymbol{p}'_{k-1} - \boldsymbol{p}_{m_i}\right)
\end{aligned}$$

Goal position

Classification problems - reaching path

The prediction model is then used as follows:

$$f\left(y_k - \hat{y}_{k|k-1,m_i}\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\angle\left(\boldsymbol{p'}_k, \hat{\boldsymbol{p}}'_{k|k-1,m_i}\right)^2}{\sigma^2}\right)$$

where

$$\hat{\boldsymbol{p}}'_{k|k-1,m_i} = \frac{d\hat{\boldsymbol{p}}_{k|k-1,m_i}}{ds} = \sum_{j=0}^{3} j\boldsymbol{a}_{j,m_i}\left(s_k - s_{k-1}\right)^{j-1}$$

Classification problems - reaching path

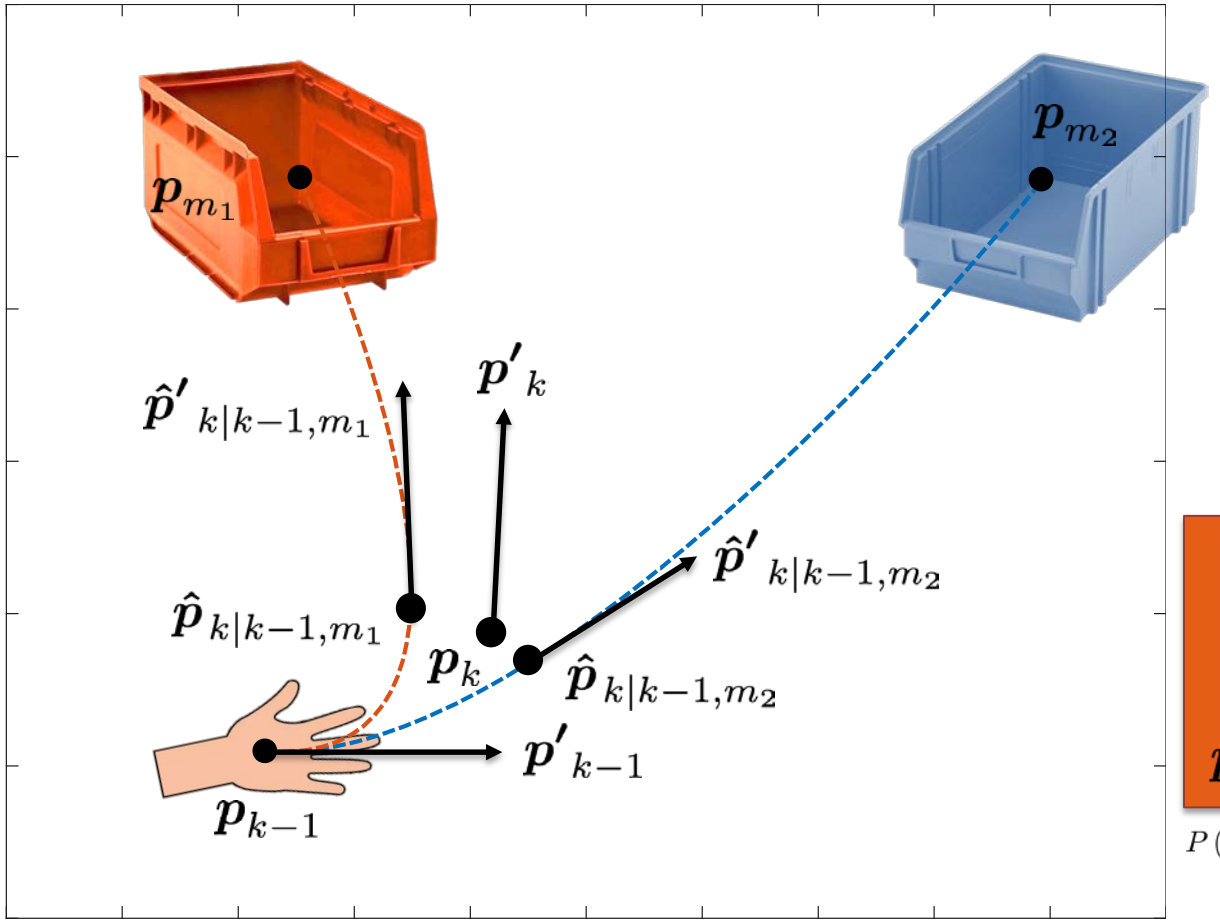
The procedure to update the classification is as follows.

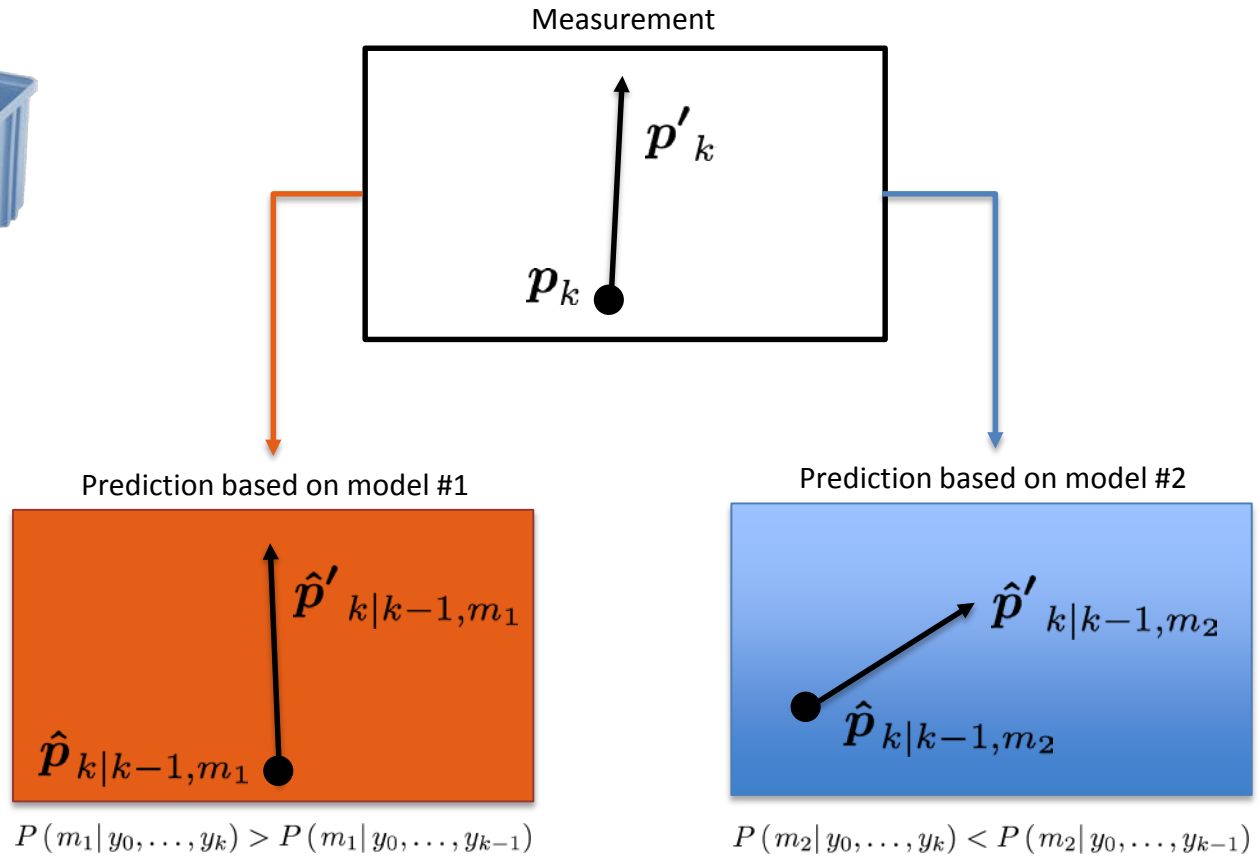
Whenever a new observation is available, for each possible goal (model), one has to:

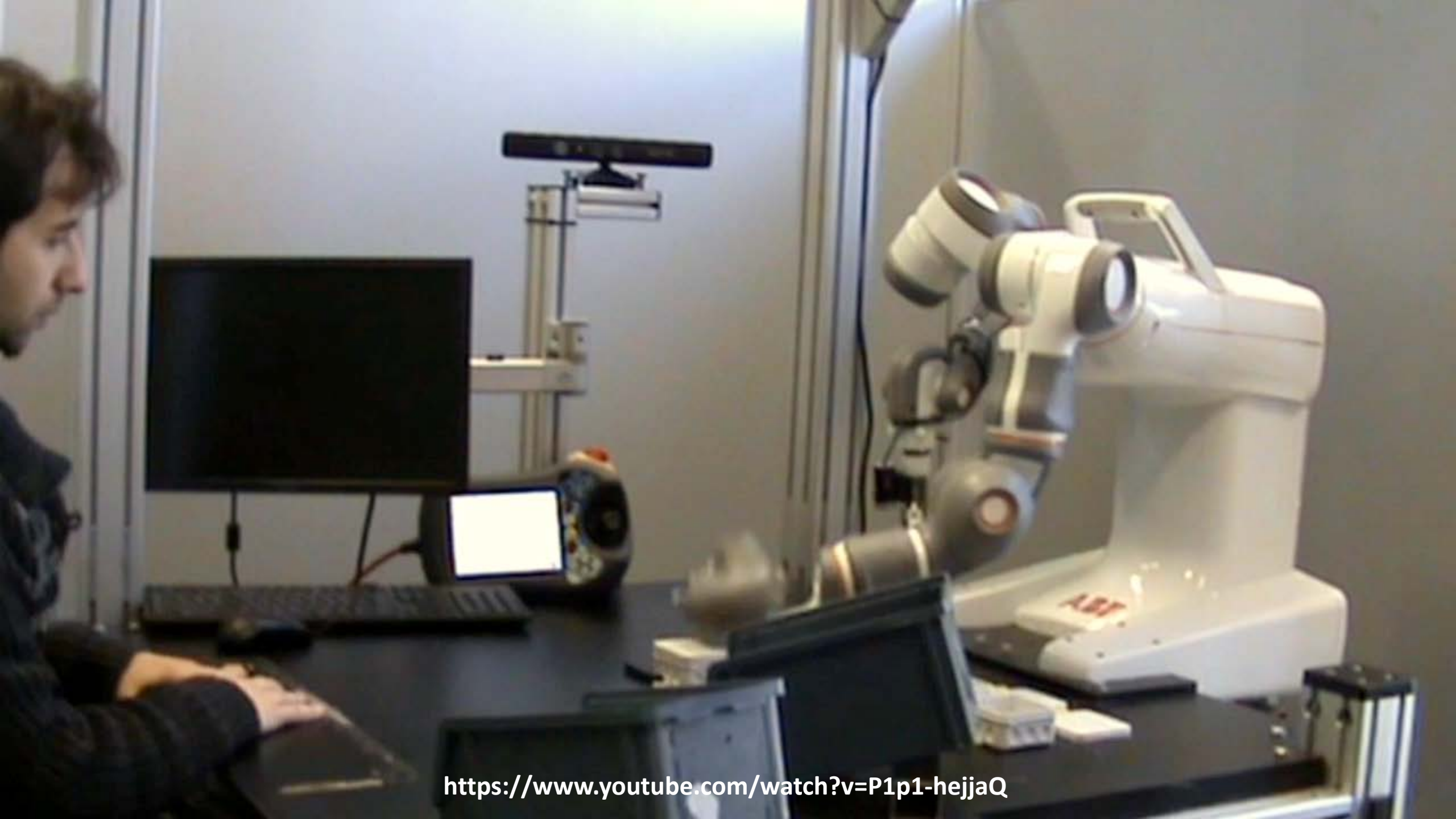
- solve the (constrained) optimal control problem;
- compute a model-based prediction of where the human should be next;
- compute how likely is the measured data according to the model.

Finally update the probability distribution for each classifier.

Classification problems - reaching path






https://www.youtube.com/watch?v=P1p1-hejjaQ

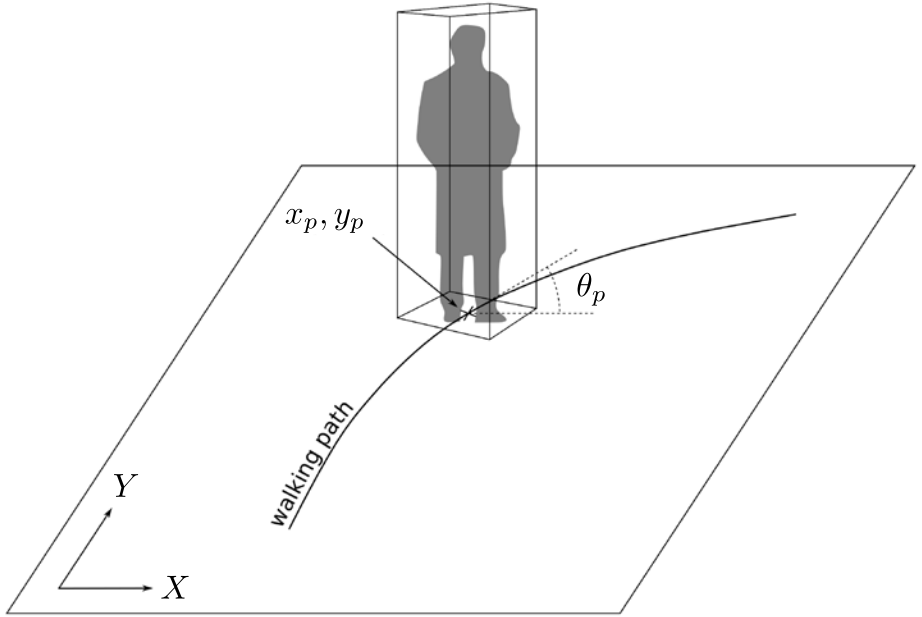
Classification problems - walking path

Problem formulation:

given an observation of the human, we want to infer where his/her walk is heading to.

What is an observation?

an observation is the set $y = \begin{bmatrix} x_p & y_p & \theta_p \end{bmatrix}^T$



Also in this case, the **prediction model** requires more details.

Classification problems - walking path

The model of goal-oriented human walking path is modelled in an implicit way by defining:

a **dynamical system** representing the human walking kinematics (unicycle model)

$$x_p' = \cos\theta_p$$
$$y_p' = \sin\theta_p$$
$$\theta' = u$$

Arechavaleta, G. , Laumond, J.P., Hicheur, H. & Berthoz, A. IEEE Tran on Rob (2008), 24:1.

a **cost functional** to be minimized (parameterized through a diagonal matrix *C*)

$$\min_{u(s)} \int_0^L \left[u(s)^2 + \left(1 + \delta(s)^T C \delta(s)\right)\right] ds$$

$$\delta = \begin{bmatrix} x_p - x_G & y_p - y_G & \theta_p - \theta_G \end{bmatrix}^T$$

Papadopoulos, A.V., Bascetta, L. & Ferretti, G. Auton Robot (2016) 40:59.

Classification problems - walking path

As in the case of reaching motion, the procedure to update the classification is as follows.

Whenever a new observation is available, for each possible goal (model), one has to:

- solve the (constrained) optimal control problem;
- compute a model-based prediction of where the human should be next;
- compute how likely is the measured data according to the model.

Finally update the probability distribution for each classifier.

Agenda

- Introduction and motivations
- Classification problems
  - Generalities
  - Examples: human reaching, human walking
- Prediction problems
  - Generalities
  - Continuous prediction problems
    - Examples: handling assistance, estimated time of arrival
  - Discrete prediction problems
    - Example: activity prediction

Prediction problems - generalities

We have understood that in order to classify a certain behavior or time series, a prediction model is necessary.

In general, we will look for:

- prediction models for time series of **continuous variables**
  - examples: value of a stock, position of an object, etc.
- prediction models for time series of **discrete and finite variables**
  - examples: weather (sunny, cloudy, …), automatic completion for text messages, recommender systems, etc.

Continuous prediction problems - generalities

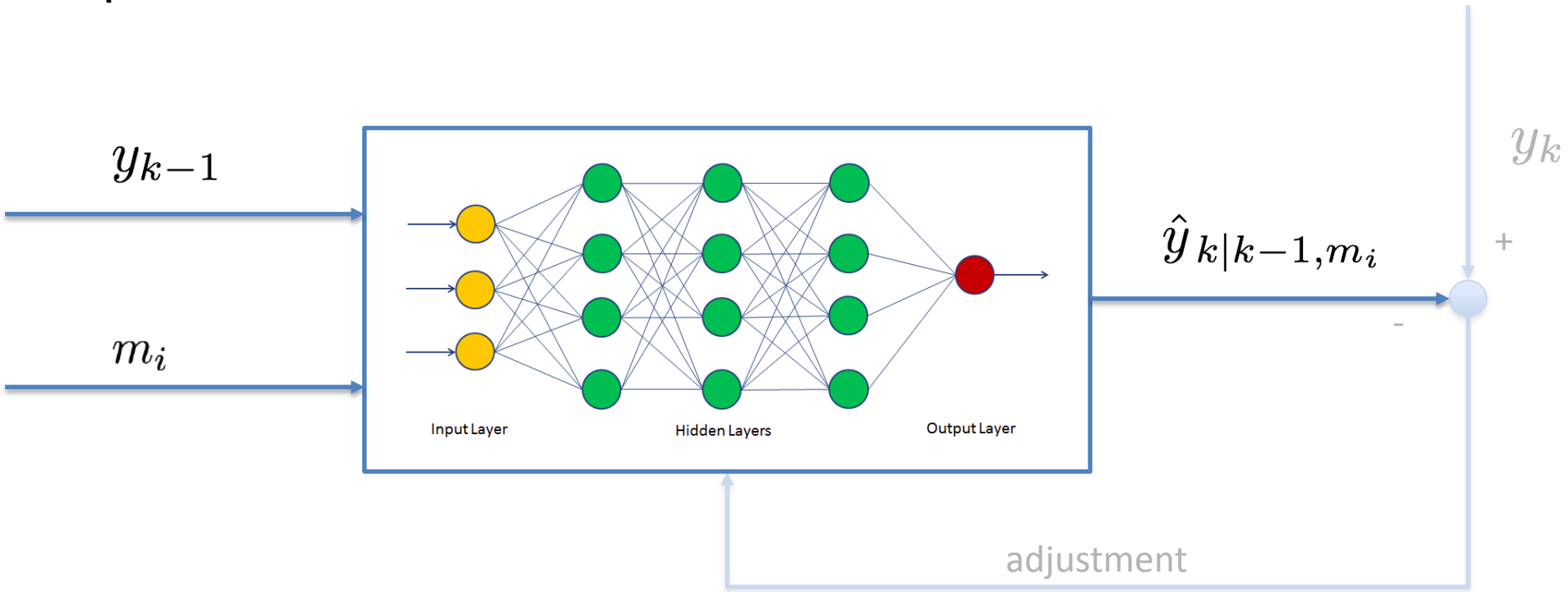
The problem of predicting a **continuous variable** is typically a regression one: given a model, one would like to estimate its parameters to best fit the data.

There is a plethora of well-known models from identification theory, .e.g. MA, AR, ARMA, etc. that can be easily regressed (identified) via linear regression methods.

Recently, nonlinear models, like (recurrent) neural networks, have started being popular, thanks to deep learning algorithms.

Continuous prediction problems - generalities

For example:



Continuous prediction problems - handling assistance

Problem formulation:

given the state of motion of the robot manually guided by the operator, we want to give assistance to reduce his/her fatigue in guiding the robot towards the inferred goal.

PbD

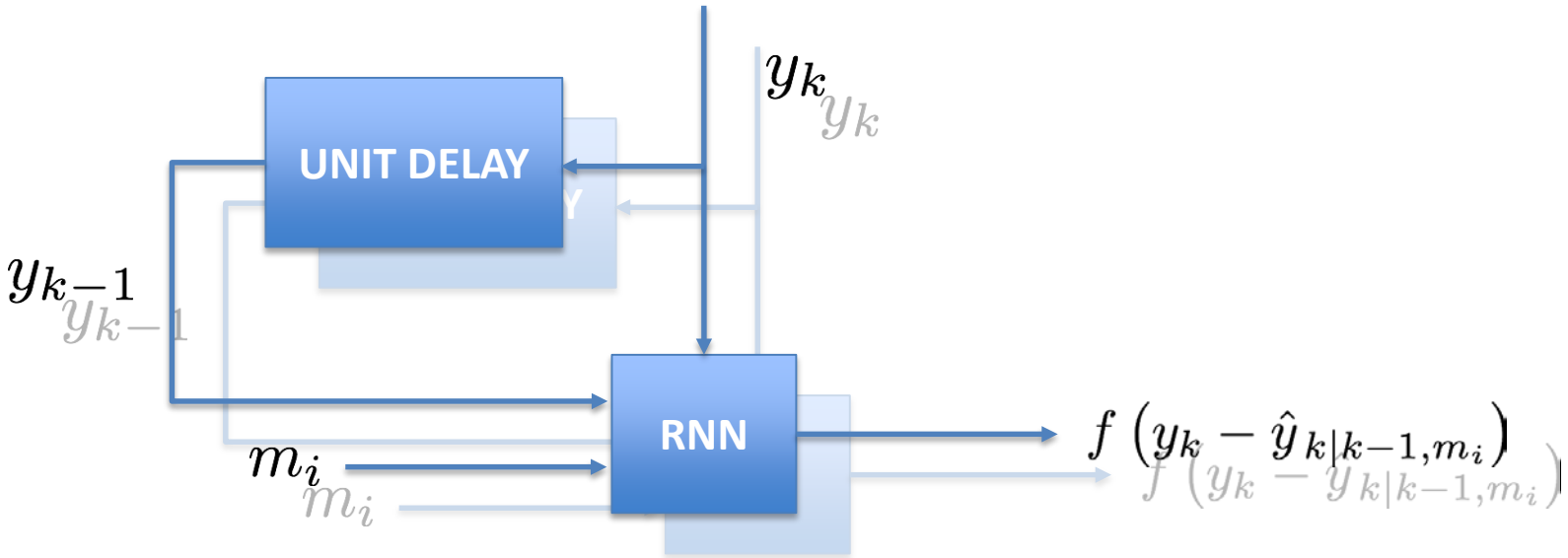
What is an observation?

an observation is the set $y = \begin{bmatrix} x_R & y_R & z_R & \dot{x}_R & \dot{y}_R & \dot{z}_R \end{bmatrix}^T$ in short

$$y = \begin{bmatrix} \boldsymbol{p}_R^T & \dot{\boldsymbol{p}}_R^T \end{bmatrix}^T$$

Continuous prediction problems - handling assistance

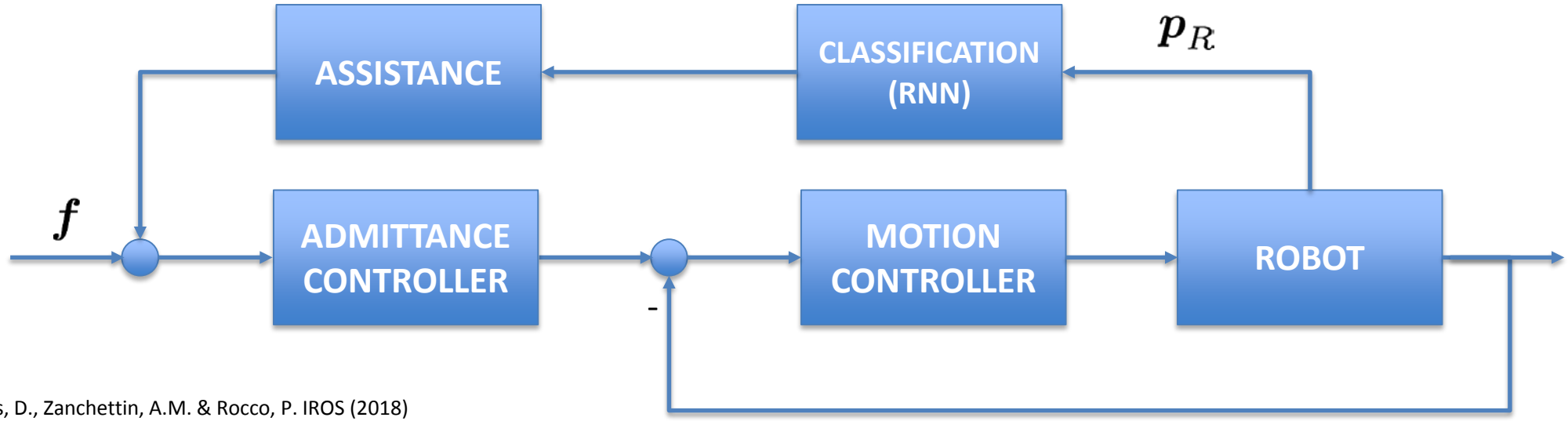
A recursive NN (RNN) is used as a **prediction model**. One RNN per each possible target is used, their outputs are then used for classification.



Continuous prediction problems - handling assistance

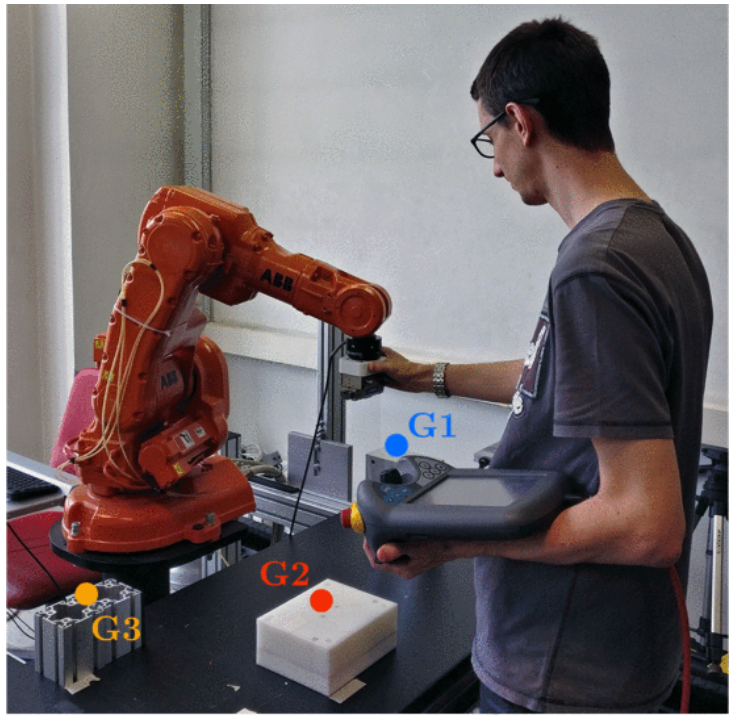
Using the classification, one can feed the admittance controller with the measured force together as well as with some additional assistance:

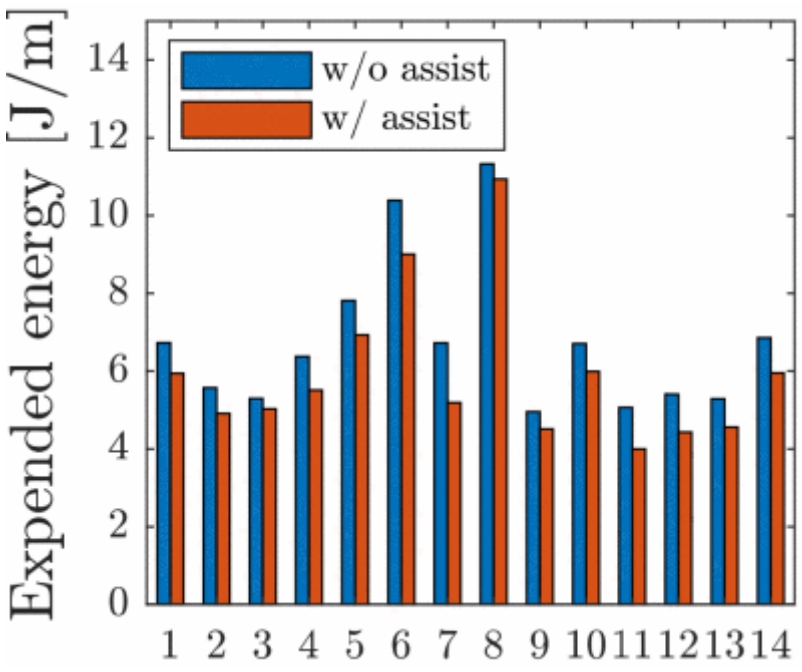
$$\boldsymbol{M}\ddot{\boldsymbol{p}}_R + \boldsymbol{D}\dot{\boldsymbol{p}}_R + \boldsymbol{K} \sum_{m_i \in \mathcal{M}} P\left(m_i \,|\, \boldsymbol{p}_0, \ldots, \boldsymbol{p}_k\right)\left(\boldsymbol{p}_R - \boldsymbol{p}_{m_i}\right) = \boldsymbol{f}$$

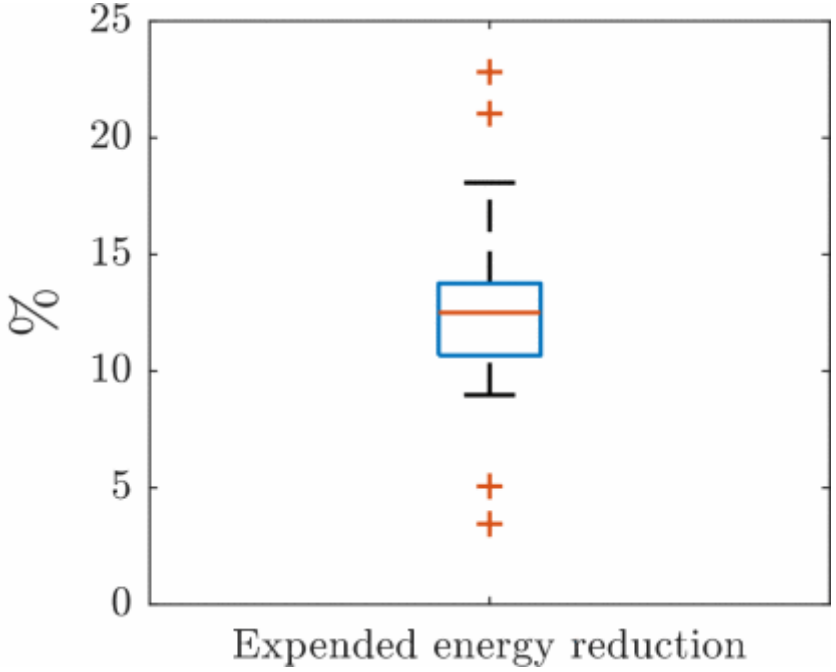


Nicolis, D., Zanchettin, A.M. & Rocco, P. IROS (2018)

Continuous prediction problems - handling assistance

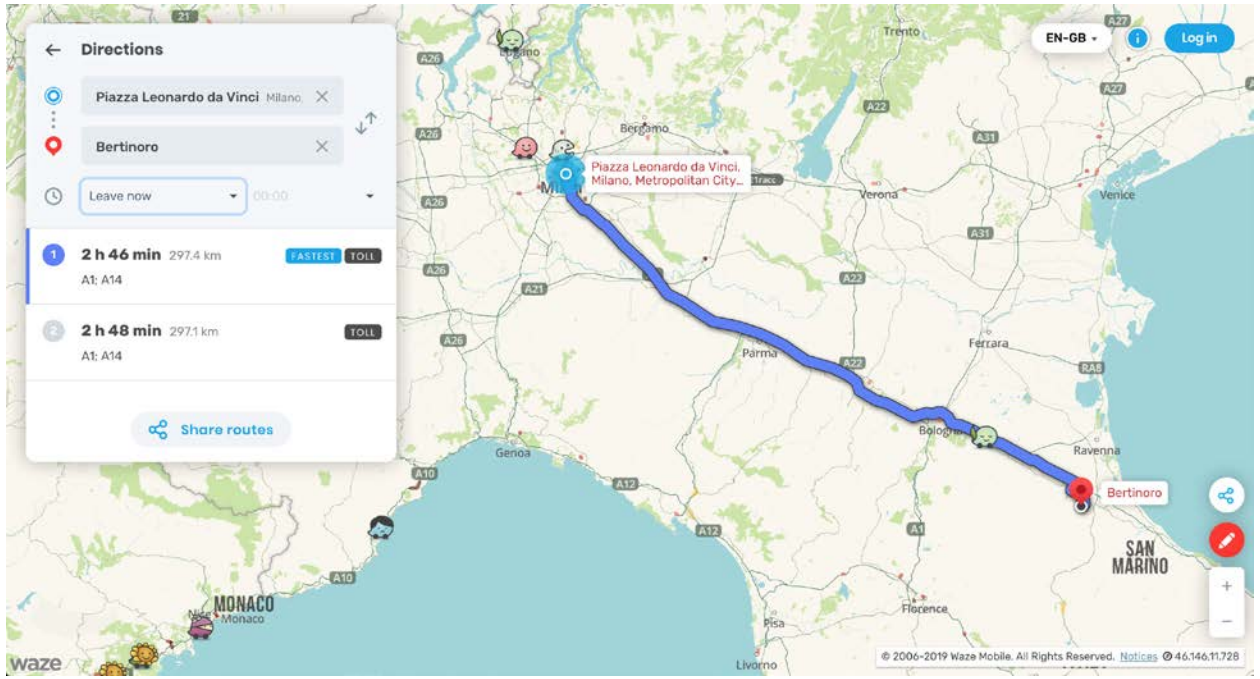






Continuous prediction problems - estimated time of arrival

Those who travel are familiar with the concept of Estimated Time of Arrival (ETA). These systems are usually online updated based on data (traffic).



Continuous prediction problems - estimated time of arrival

This problem is also typical in collaborative robotics. An intelligent collaborative robot, in fact, can ask itself:

- when the operator will delivery the part that I need to process?
- when will I be requested to assist the operator?
- when will the operator need the part/tool that I am using?
- when will the operator leave the collaborative workspace so I can step in?
- etc.

These questions can be answered by predicting when the operator will complete the current activity.

Continuous prediction problems - estimated time of arrival

In principle, this problem can be easily addressed through conditional probabilities.

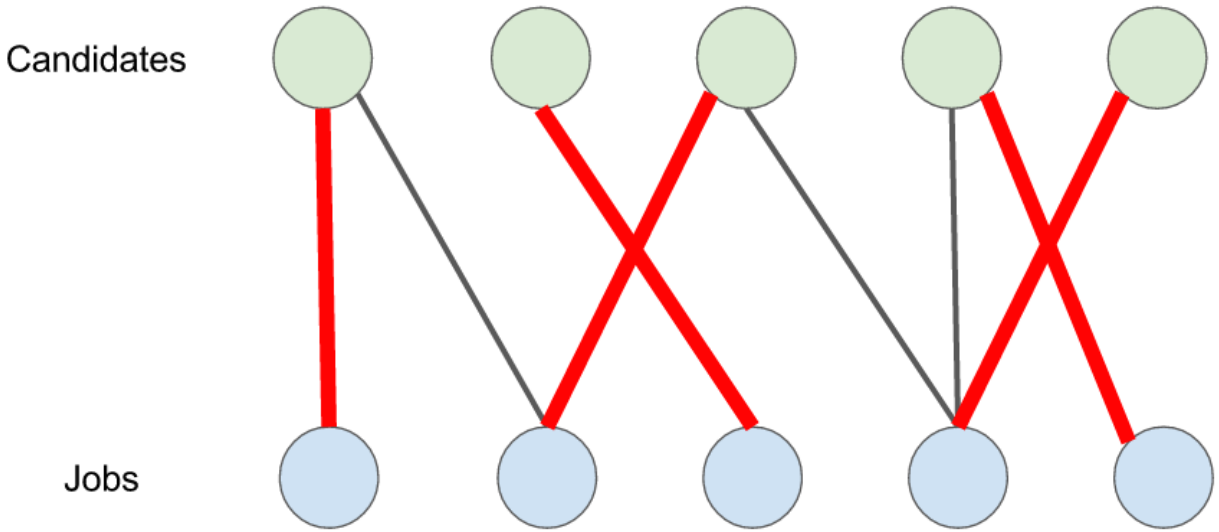
For example, I can collect all the travel durations from Milan to Bertinoro and estimate the time of arrival after having travelled for 2 hours by

$$P\left(t \middle| t \geq 2\right)$$

Clearly, the information about **where I am** after having travelled for 2 hours can give me some additional insights, i.e. being either in Parma or Bologna after 2 hours can dramatically change my estimated time of arrival.

Continuous prediction problems - estimated time of arrival

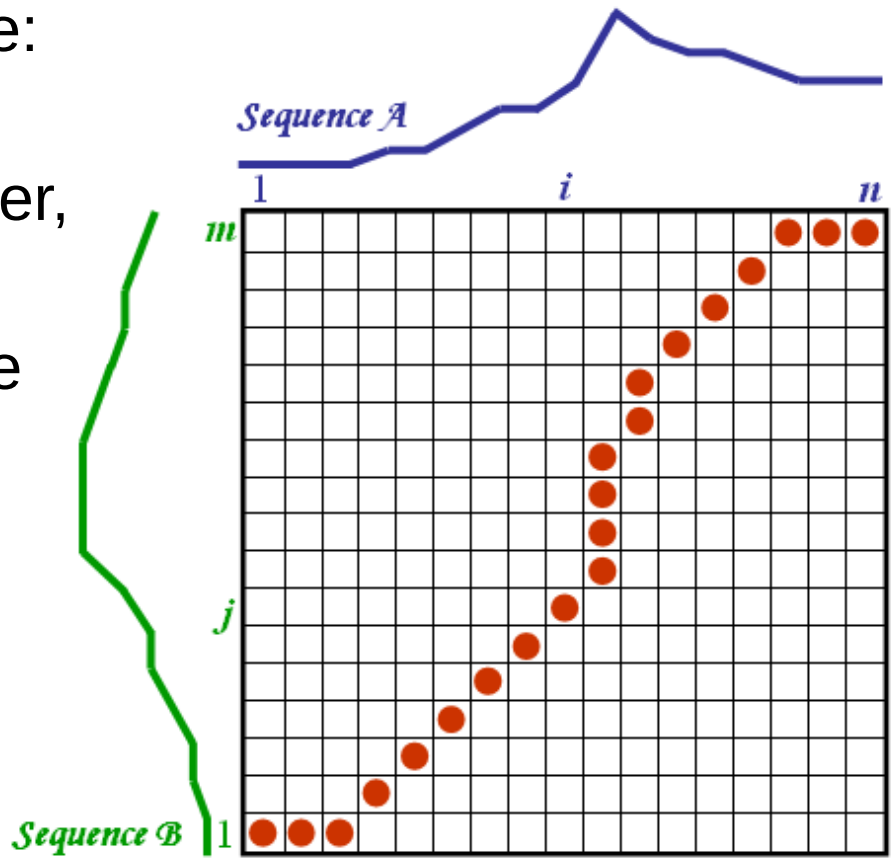
Dynamic Time Warping (DTW) is an algorithm to measure the “distance” between two time series. It is essentially a particular case of the Minimum Weight B-Matching problem (MWM), i.e. pairing two sets of vertices in a bipartite graph.



Continuous prediction problems - estimated time of arrival

The main differences between DTW and MWM are:

- every element from the first sequence must be matched with one or more element from the other, and vice versa;
- the first element from the first sequence must be matched with the first element from the other;
- the same applies to the last elements;
- the mapping of the element from the first sequence to elements from the other must be **monotonically increasing**, and vice versa.



Pseudo-code of the DTW algorithm:

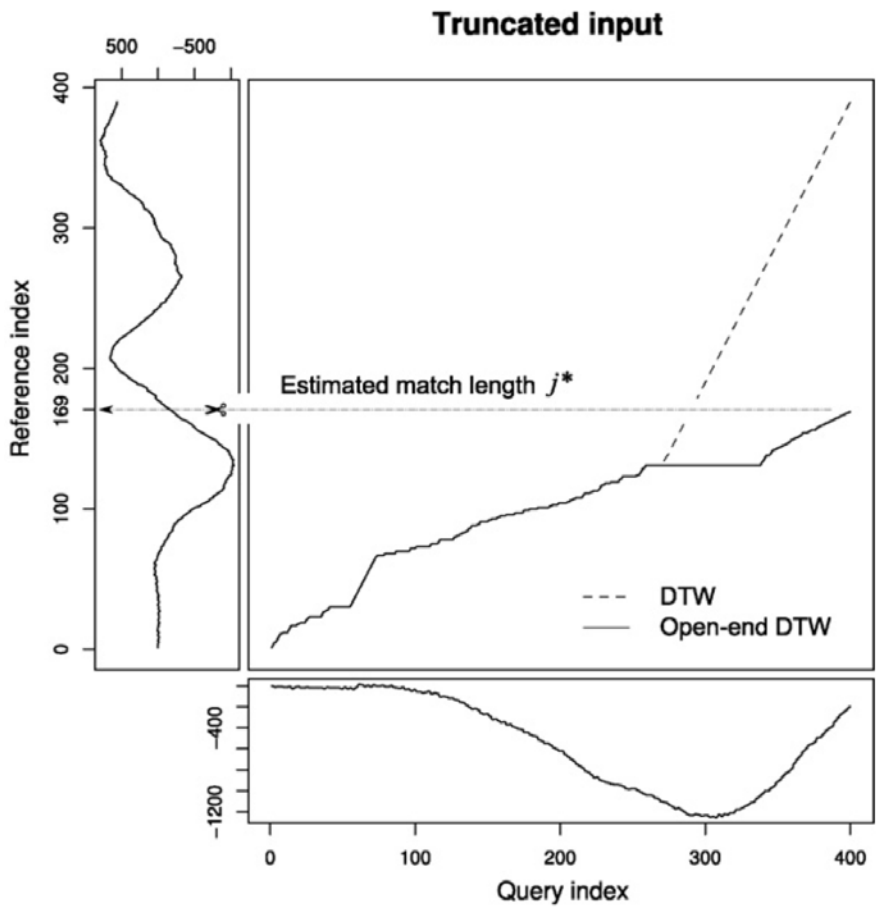
```
int DTWDistance(s: array [1..n], t: array [1..m]) {
    DTW := array [0..n, 0..m]
    for i := 1 to n for j := 1 to m DTW[i, j] := infinity
    DTW[0, 0] := 0
    for i := 1 to n
        for j := 1 to m
            DTW[i, j] := dist(s[i], t[j]) + minimum(DTW[i-1, j ],  // insertion
                                                    DTW[i , j-1],  // deletion
                                                    DTW[i-1, j-1]) // match
    return DTW[n, m]
}
```

Continuous prediction problems - estimated time of arrival

There exists a so-called “open-ended” version of the DTW algorithm (OE-DTW).

The OE-DTW allows the comparison of incomplete input time series with complete references.

OE-DTW outputs the **fraction of reference time series that is matched by the given one**.



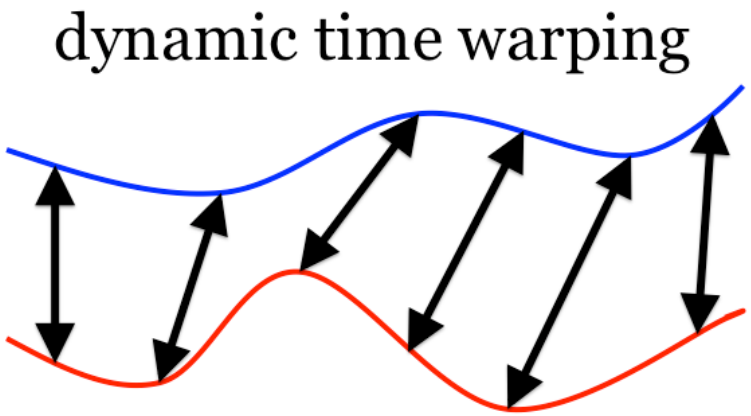
Tormene, P., Giorgino, T., Quaglini, S. & Stefanelli, M.. Artificial intelligence in medicine (2009) 45:1.

Pseudo-code of the open-ended DTW:

```
int DTWDistance(s: array [1..n], t: array [1..m]) {
    DTW := array [0..n, 0..m]
    for i := 1 to n for j := 1 to m DTW[i, j] := infinity
    DTW[0, 0] := 0
    for i := 1 to n
        for j := 1 to m
            DTW[i, j] := dist(s[i], t[j]) + minimum(DTW[i-1, j ],  // insertion
                                                    DTW[i , j-1],  // deletion
                                                    DTW[i-1, j-1]) // match
    return argmin(DTW[1..n, m])
}
```

Continuous prediction problems - estimated time of arrival

Problem formulation:
given an observation of an ongoing activity, we want to predict when the activity is going to finish.



What is an observation?

an observation is the sequence of hand positions $\boldsymbol{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m\}$

A **model** to compare the observed time series with is needed.

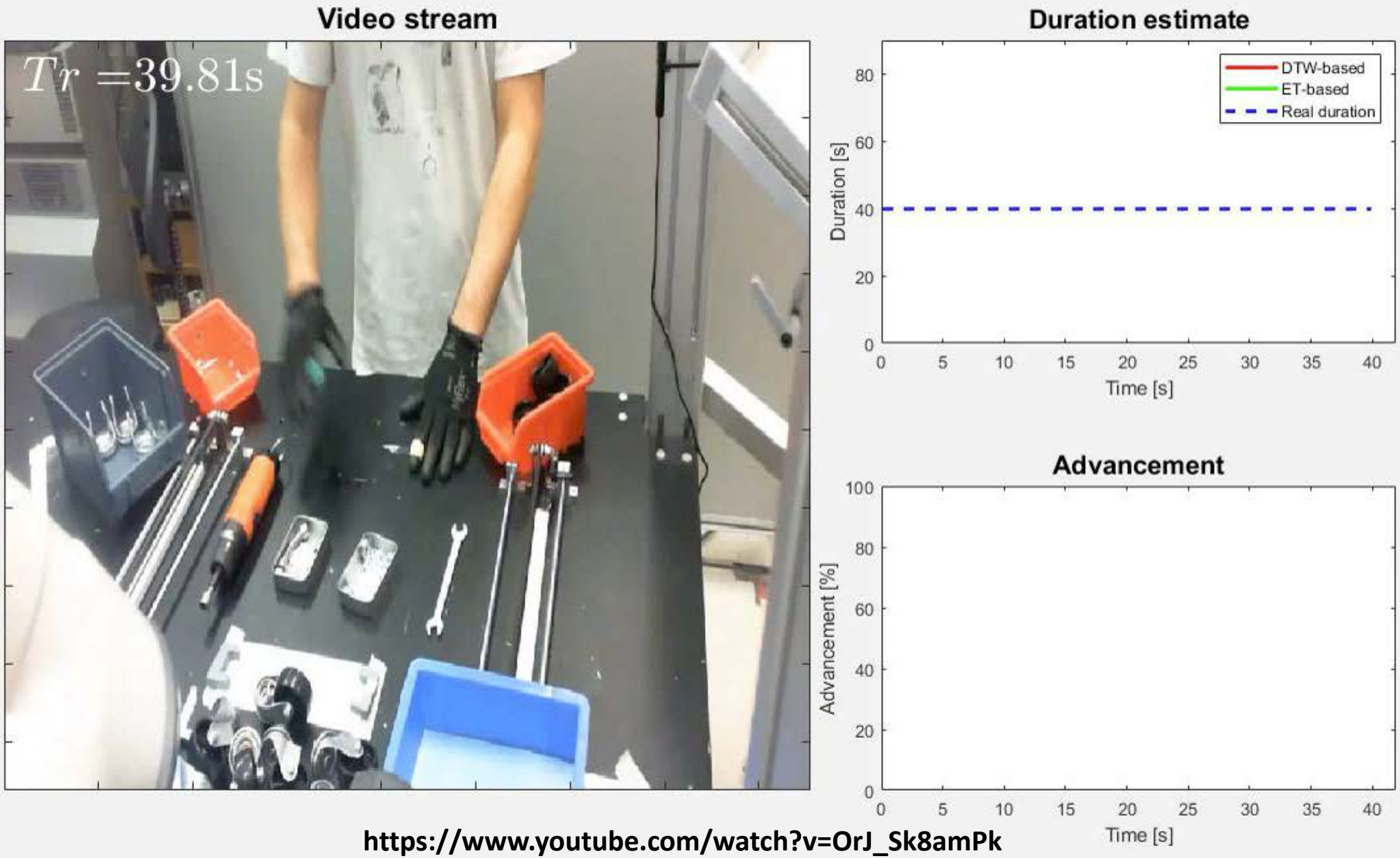
Continuous prediction problems - estimated time of arrival

We assume a template action is already available (time series to compare with). Then we can run the OE-DTW algorithm and update the ETA as follows:

$$\mathrm{ETA} = \mathrm{ET}\frac{n}{\mathrm{DTW}} - \mathrm{ET} = \mathrm{ET}\left(\frac{n}{\mathrm{DTW}} - 1\right)$$

Elapsed Time

where DTW is the number of matching samples between the two time series, ET is the elapsed time.

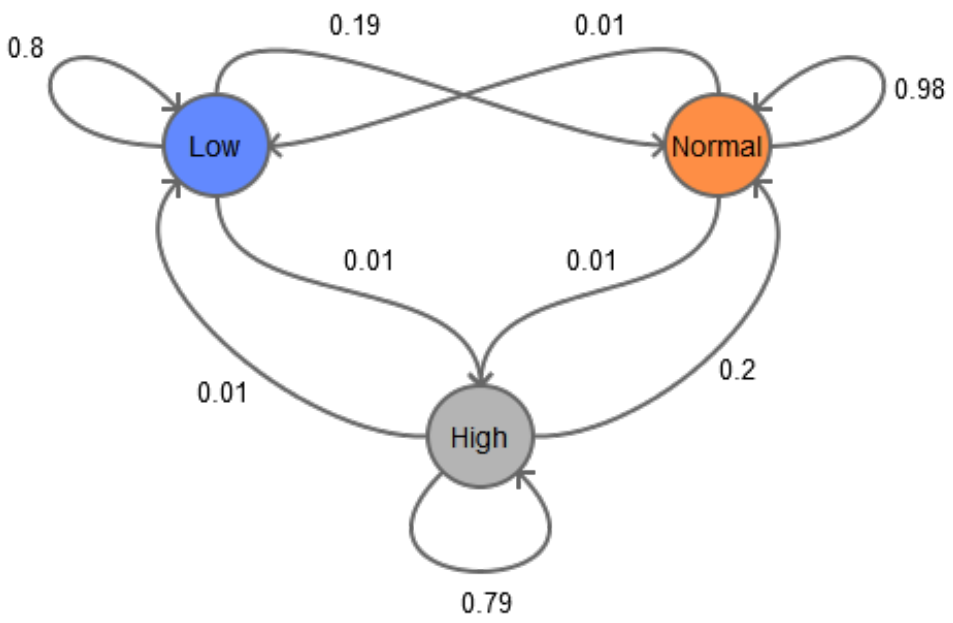


https://www.youtube.com/watch?v=OrJ_Sk8amPk

Discrete prediction problems - generalities

Another interesting problem is to predict discrete values, i.e. symbols. The tool typically adopted for this kind of problems are Markov chains.

A **Markov chain** is a stochastic model describing a sequence of possible symbols in which the probability of each symbol depends only on the previous one.



Roughly speaking, Markov chains are stochastic automata (state machines).

Discrete prediction problems - generalities

The Markov chain is well-represented by transition probabilities (weight on arcs of the graph):

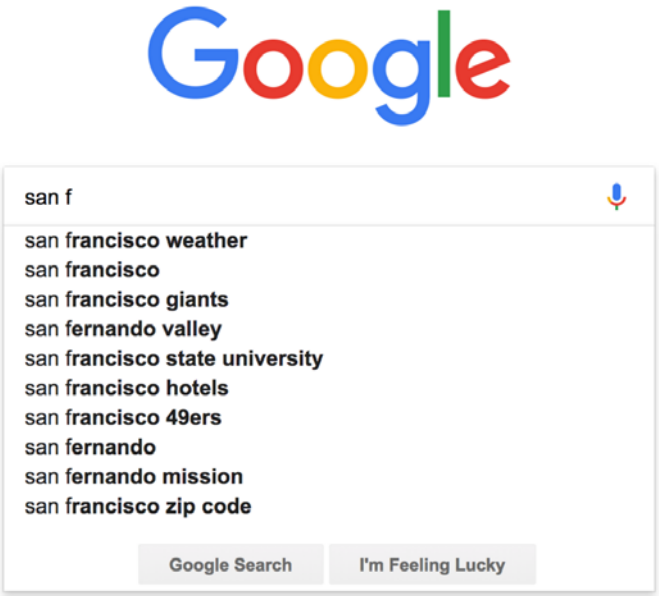
$$Q_1 = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1m} \\ \vdots & \vdots & \vdots & \vdots \\ q_{m1} & q_{m2} & \cdots & q_{mm} \end{bmatrix}^T_{m \times m}$$

where q_{ij} represents the probability of symbol *j* following symbol *i*.

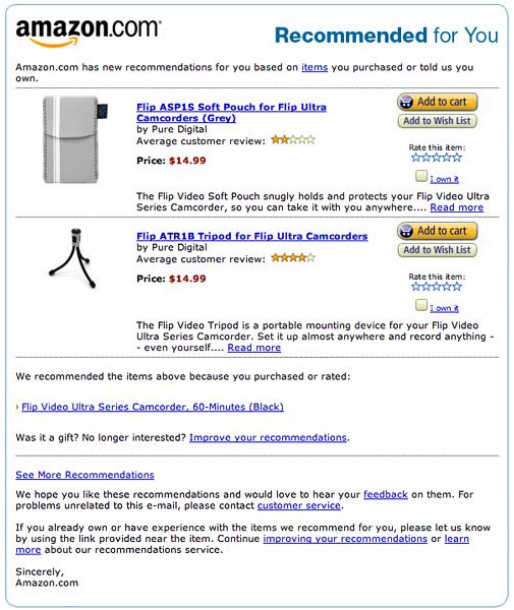
One step-ahead prediction is straightforward: $P\left(y_k = j \middle| y_{k-1} = i\right) = q_{ij}$.

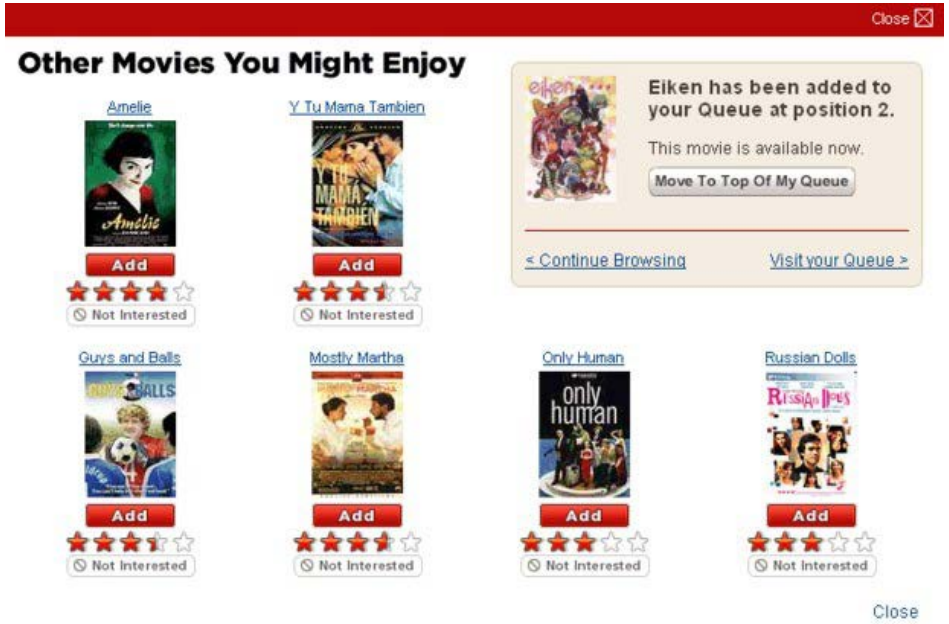
Discrete prediction problems - generalities

Many discrete processes, however, do not satisfy Markov's hypothesis, i.e. the next symbol (whatever this means) does not depend *just* on the previous one.



Text autocomplete





Recommender systems

Discrete prediction problems - generalities

In manufacturing (especially in assembly) the next activity does not depend solely on the current one (Markov's assumption), i.e. the process has memory (the assembly sequence).

$$P\left(y_k \middle| y_{k-1}, y_{k-2}, \ldots, y_{k-n}\right) \neq P\left(y_k \middle| y_{k-1}\right)$$

Therefore, we want to estimate the next activity based on the last *n* observations. A possible way is to (linearly) mix multiple-steps transition probabilities (i.e. using Mixture Transition Distribution, MTD, models):

$$P\left(y_k \middle| y_{k-1}, y_{k-2}, \ldots, y_{k-n}\right) = \sum_{i=1}^{n} \lambda_i P\left(y_k \middle| y_{k-i}\right)$$

Discrete prediction problems - generalities

Define $\boldsymbol{X}_k$ the probability distribution of all symbols (activities) at discrete time *k*, then a prediction of such a distribution can be obtained as follows

$$\hat{\boldsymbol{X}}_k = \sum_{i=1}^{n} \lambda_i \boldsymbol{Q}_i \boldsymbol{X}_{k-i} \qquad \boldsymbol{X}_k = \begin{bmatrix} 0 & 0 & \dots & \underset{\text{Observation}}{1} & 0 & \dots \end{bmatrix}^T$$

where matrices $\boldsymbol{Q}_i$ are the transition probabilities (as in standard Markov chains), while λ_i are non-negative weights to be tuned.

A simple (data-driven) procedure to tune these weights is given in the following.

Discrete prediction problems - generalities

We can use all the previous observations, and tune the weights in order to minimize the prediction error, i.e.

$$\min_{\lambda_i \geq 0} \sum_{j=0}^{N} \underbrace{\left\| \overbrace{\sum_{i=1}^{n} \lambda_i \boldsymbol{Q}_i \boldsymbol{X}_{k-i-j}}^{\text{Prediction at discrete time } k-j \text{ using previous observations}} - \boldsymbol{X}_{k-j} \right\|^2}_{\text{Prediction error at discrete time } k-j}, \sum_{i=1}^{n} \lambda_i = 1$$

Zanchettin, A.M., Casalino, A., Piroddi, L. & Rocco, P. IEEE Transactions on Industrial Informatics, 2018.

Discrete prediction problems - generalities

For those of you who are familiar with identification techniques, the method is very similar to the least-square estimation of the parameters of an auto-regressive model of order *n*, *AR(n)*.

$$\hat{\boldsymbol{X}}_{k-j} = \sum_{i=1}^{n} \lambda_i \boldsymbol{Q}_i \boldsymbol{X}_{k-i-j}$$

Prediction model

$$\hat{y}_{k-j} = \sum_{i=1}^{n} \lambda_i y_{k-i-j}$$

$$\boldsymbol{e}_{k-j} = \hat{\boldsymbol{X}}_{k-j} - \boldsymbol{X}_{k-j} = \sum_{i=1}^{n} \lambda_i \boldsymbol{Q}_i \boldsymbol{X}_{k-i-j} - \boldsymbol{X}_{k-j}$$

$$e_{k-j} = \hat{y}_{k-j} - y_{k-j} = \sum_{i=1}^{n} \lambda_i y_{k-i-j} - y_{k-j}$$

Prediction error

$$\min_{\lambda_i \geq 0} \sum_{j=0}^{N} \|\boldsymbol{e}_{k-j}\|^2, \sum_{i=1}^{n} \lambda_i = 1$$

$$\min_{\lambda_i} \sum_{j=0}^{N} e_{k-j}^2$$

$$\boldsymbol{\lambda} = \mathrm{argmin}_{\lambda_i \geq 0} \|\boldsymbol{A}\boldsymbol{\lambda} - \boldsymbol{b}\|^2, \sum_{i=1}^{n} \lambda_i = 1$$

$$\boldsymbol{\lambda} = \boldsymbol{A}^{\dagger}\boldsymbol{b}$$

Discrete prediction problems - activity prediction

Problem formulation:

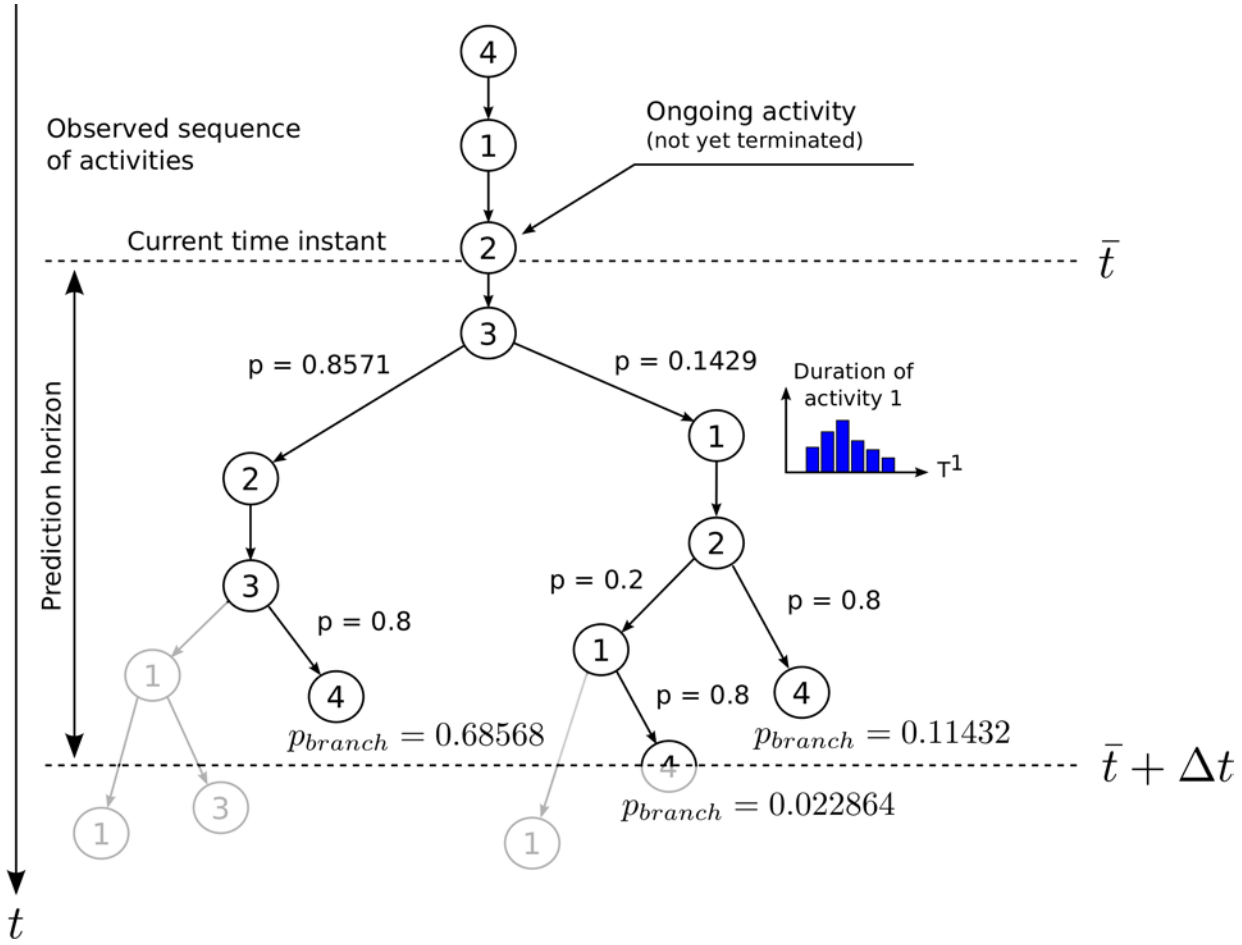
from the human perspective, a manufacturing process consists of two jobs, one can be performed autonomously, the other requires the prompt assistance of the robot. Having observed all the previous activities (portions of a job), we want to infer when the human will require assistance from the robot.

What is an observation?

an observation is the sequence of activities up to current time instant, i.e.

$$\boldsymbol{X}_k, \boldsymbol{X}_{k-1}, \ldots, \boldsymbol{X}_{k-n}$$

Discrete prediction problems - activity prediction

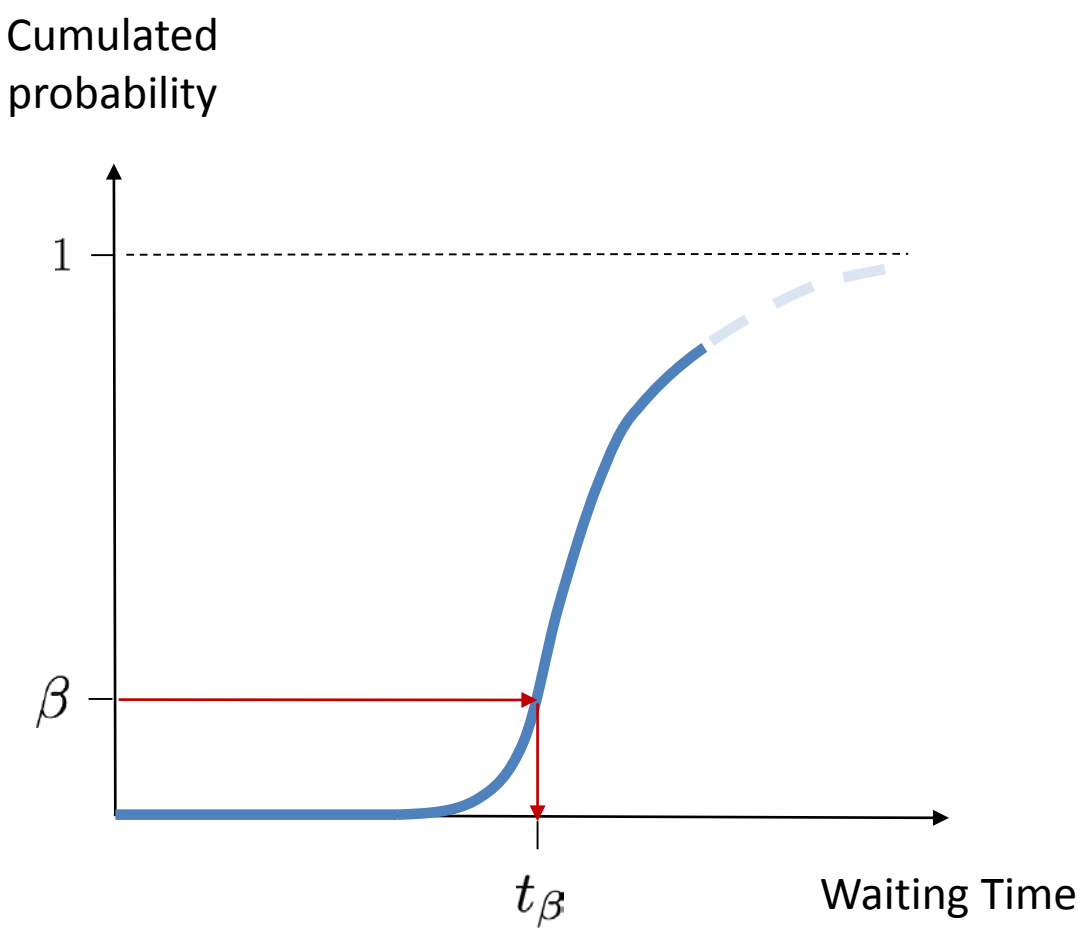


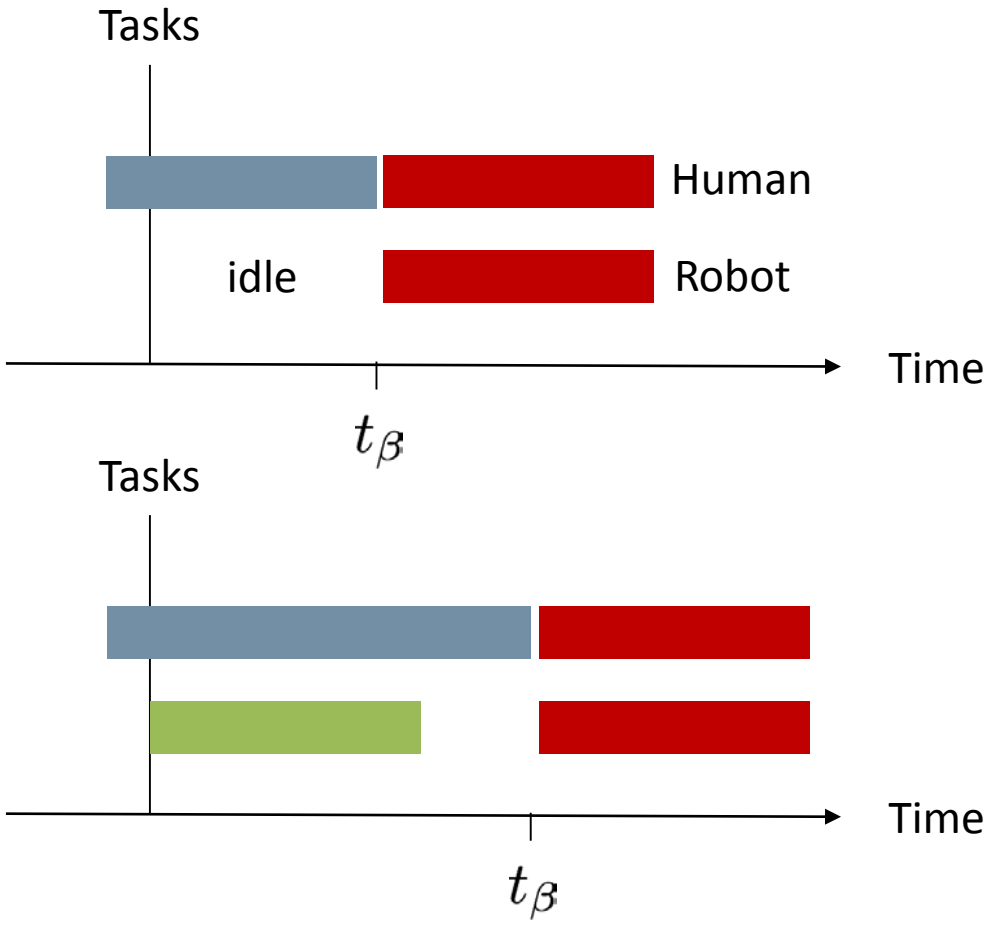
We can use the prediction model to recursively build the reachability tree.

After having collected enough data, we are also able to extract particles regarding the duration of each activity.

Given a prediction horizon we can also prune path along the trees that exceed the horizon.

Discrete prediction problems - activity prediction





Cividini, F., Casalino, A., Zanchettin, A.M., Piroddi, L. & Rocco, P. INCOM, 2018.

Optimal scheduling

Round-robin scheduling

https://www.youtube.com/watch?v=KG7WLNdi8Uw

POLITECNICO
MILANO 1863