Learning and Intelligent Systems

Sequential (time-series) models

Prof. Thomas Hofmann
Data Analytics Laboratory (da.inf.ethz.ch)
Slides: Andreas Krause
Time-series modeling

- So far, we have assumed

\[(x_i, y_i) \sim P(X, Y)\]

i.e., data points are generated independently, and identically distributed.

- Poor assumption, if data are generated over time, and dependent over time
Motivation for time-series models

- Data from many domains naturally exhibits temporal dependencies.

**Examples:**
- Biological data
- Medical data
- Weather forecasting
- Environmental monitoring
- Language modeling
- Sales trends
- Financial data
- ...

Most basic task:

- You observe a sequence of observations, and would like to predict the next one.
- I.e., you observe \( y_1, y_2, \ldots, y_t \)
  and your goal is to predict \( y_{t+1} \)
Dynamical models

- Static model

- Dynamic model

Typically assume \textit{discrete, unit-length} time steps

How should we model dependence over time?

\begin{itemize}
  \item \textbf{Weather}
  \item \textbf{Weather}_1 \hspace{1cm} \textbf{Weather}_2 \hspace{1cm} \textbf{Weather}_3 \hspace{1cm} \ldots
\end{itemize}
Markov Chains: a generative model

- A Markov Chain is a generative model:
  - Sample the starting state $y_1$ from a categorical distribution with $c$ states
    \[ P(Y_1 = y) = p_y \]
  - For $t=2:T$, given $Y_{t-1} = y'$, sample $Y_t$
    \[ P(Y_t = y \mid Y_{t-1} = y') = \theta_{y \mid y'} \]

- Key consequence: Next state only depends on current state, not on the past
  ➔ Markov chains have no memory (memoryless)
Markov models / Markov chains

Markov assumption:
\[ \forall t : P(Y_t \mid Y_1, \ldots, Y_{t-1}) = P(Y_t \mid Y_{t-1}) \]
\[ \equiv \forall y_1, \ldots, y_t : P(Y_t = y_t \mid Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}) = P(Y_t = y_t \mid Y_{t-1} = y_{t-1}) \]

Stationarity assumption:
\[ \forall t, y, y' : P(Y_{t+1} = y \mid Y_t = y') = P(Y_t = y \mid Y_{t-1} = y') \]
Illustration: Transition diagram for Markov chain

\[ Y_t \in \{\text{Sunny, Cloudy, Rainy}\} \]

\[ c = 3 \text{ states} \]
What if we want memory?

- Suppose we want to allow state transitions to depend on the past as well

\[ P(Y_{t+1} \mid Y_{t-k+1}, \ldots, Y_t) \]

- I.e., the transitions only depend on the previous k states
- The resulting model „remembers“ the previous k-1 states
- It is called \textit{k-th order Markov Model}
Reduction: $k$-th order to 1st order

Can always express $k$-th order Markov models as 1st order Markov models. How?

In general, $Z_t = [Y_t, Y_{t-1}, \ldots, Y_{t-k+1}]$

Choose $P(Z_t|Z_{t-1})$ to ensure consistency among $Y_t$.
How should we choose $k$?

- **Number of parameters:**

$$P(y_{t+1} = y_{t+1} \mid y_t = y_t, \ldots, y_{t-k+1} = y_{t-k+1}) = \Theta(y_{t+1} \mid y_{t+1}; t)$$

$$l(c-1) \cdot c^k = \Theta(c^{k+1})$$

- In general complexity (memory, run time) will be **exponential** in $k$
- In practice, typically need to choose $k$ very small
- Limited memory is a downside of Markov models
- We’ll see how to (partially) fix this later
Prediction in Markov Chains

E.g.: Given that it rains now, how likely is it to rain tomorrow?

\[
P(Y_{t+1} = y | Y_1 = y_1, \ldots, Y_t = y_t) = P(Y_{t+1} = y | Y_t = y_t) = \Theta_{Y_{t+1}Y_t} \quad \text{given as a parameter of the model}
\]
E.g.: Given that it rains now, how likely is it to rain two days from now?

\[
P(Y_{t+2} = y | Y_{1:t}) = \sum_{y_t} P(Y_{t+2} = y, Y_{t+1} = y_1 | Y_{1:t})
\]

\[
= \sum_{y_t} P(Y_{t+2} = y | Y_{1:t}, Y_{t+1} = y_1) \cdot P(Y_{t+1} = y_1 | Y_{1:t})
\]

\[
= \sum_{y_t} P(Y_{t+2} = y | Y_{t+1} = y_1) \cdot P(Y_{t+1} = y_1 | Y_{1:t})
\]

\[
= \sum_{y_t} \Theta_{y_1} \cdot \Theta_{y_1} y_t
\]
Matrix/vector notation

- Represent $P(Y_t)$ as probability vector
  \[ p^{(t)} = [p_1^{(t)}, p_2^{(t)}, \ldots, p_c^{(t)}] \in \mathbb{R}^c \]

- Represent $P(Y_{t+1}=y \mid Y_t = y')$ as matrix $T \in \mathbb{R}^{c \times c}$
  \[ T_{y', y} = P(Y_{t+1} = y \mid Y_t = y') \]

- Hereby, $T$ is called the transition matrix

- Then it holds that
  \[ p^{(t+1)} = p^{(t)}T \]
Example

\[
\begin{pmatrix}
0.4 & 0.4 & 0.2 \\
0.1 & 0.7 & 0.2 \\
0.2 & 0.2 & 0.6
\end{pmatrix}
\]

\[\rho^{(t)} = \begin{pmatrix} 0.5 & 0.5 & 0 \end{pmatrix}\]

\[\rho^{(t+1)} = \rho^{(t)} \cdot T\]
Prediction in Markov Chains

E.g.: Given that it rains now, how likely is it to rain a week from now?

\[ P(Y_{t+1}) = P(Y_t) \cdot T \]
\[ P(Y_{t+2}) = P(Y_{t+1}) \cdot T = P(Y_t) \cdot T^2 \]
\[ P(Y_{t+3}) = P(Y_{t+2}) \cdot T = P(Y_t) \cdot T^3 \]
Side note: Stationary distributions

- We can ask what happens as \( t \to \infty \)
- Given prior \( p^{(1)} = P(Y_1 = \cdot) \)

what can we say about the limit

\[
p = \lim_{t \to \infty} p^{(t)} = \lim_{t \to \infty} p^{(1)} T^{t-1}
\]

- In general, it may not exist.
- Even if it exists, it may not be unique.
- But under some conditions, the limit exists and is unique.
  - In this case, \( p \) is called the **stationary distribution**
- A sufficient condition is that \( T_{y', y} > 0 \)
BREAK
Learning a Markov Chain

- Given data, how do we estimate the parameters of the Markov Chain?
- What form does the data take?

Data contains at least one sequence of labels
  - May contain multiple \(m\) sequences (often \(m=1\))
  - Each time series might be of different length

\[
D = \{ (y_1^{(1)}, y_2^{(1)}, \ldots, y_{n_1}^{(1)}), \ldots, (y_1^{(m)}, y_2^{(m)}, \ldots, y_{n_m}^{(m)}) \}\]
Estimating the parameters of a Markov Chain

- **Given**: a data set
  \[ D = \left\{ (y_1^{(1)}, y_2^{(1)}, \ldots, y_{n_1}^{(1)}), \ldots, (y_1^{(m)}, y_2^{(m)}, \ldots, y_{n_m}^{(m)}) \right\} \]

- **Goal**: Estimate parameters
  \[ P(Y_1 = y) = p_y \quad P(Y_t = y \mid Y_{t-1} = y') = \theta_{y \mid y'} \]

- **How**?

- **Key insight**: Even though the \( Y_t \) are dependent, the transitions are independent (Markov property)!
Estimating transition probabilities

\[
D = \{ (y_{1:n_1}^{(1)}), \ldots, (y_{1:n_m}^{(m)}) \}
\]

\[
P(D | \Theta) = \prod_{i=1}^{m} P(y_{1:n_i}^{(i)} | \Theta)
\]

\[
\Theta = [\theta_1, \ldots, \theta_c]
\]

\[
\theta_1, \ldots, \theta_{cil}
\]

\[
\log P(D | \Theta) = \sum_{i=1}^{c} \text{Count}(y_i = j) \log p_{ij} + \sum_{i=1}^{c} \sum_{j=1}^{c} \text{Count}(x_{t+1} = j | x_t = i) \log \theta_{ij}
\]
Maximum likelihood estimation

Given data

\[ D = \{(y_1^{(1)}, y_2^{(1)}, \ldots, y_{n_1}^{(1)}), \ldots, (y_1^{(m)}, y_2^{(m)}, \ldots, y_{n_m}^{(m)})\} \]

The MLE of the parameters is given by:

\[ \hat{p}_y = \frac{\text{Count}(Y_1 = y)}{m} \]

\[ \hat{\theta}_{y|y'} = \frac{\text{Count}(Y_t = y, Y_{t-1} = y')}{\text{Count}(Y_{t-1} = y')} \]
Regularization & Smoothing

- Might want to regularize counts (use pseudo-counts, as in GBC) = smoothing
- Especially important if some state transitions are rare
- Also, if we have few time series ($m$ is small), then we will not be able to estimate the prior distribution very accurately
Beyond categorical variables

- So far: considered Markov models where each time step is represented as a categorical variable.

- What if we want to model
  - Multiple categorical variables (e.g., to implement memory)?
  - Continuous variables?
Sequence prediction

Let’s reconsider the basic problem of sequence prediction: Predict $y_{t+1}$ from $y_1, y_2, \ldots, y_t$

Basic challenge: estimate conditional distribution

$$P(Y_{t+1} \mid y_1, \ldots, y_t)$$

We can view this as a supervised learning problem!

$$\approx \hat{P}(y_{t+1} \mid y_t, y_{t-k+1}, \ldots, y_t, \theta)$$

Key challenge:

Number of conditioning variables increases with $t$
Potential solution

- Restrict conditional distribution to depend only on some finite set of features of $y_{1:t}$

**Typical example:** Restrict dependence to depend only on last $k$ values

$$P(Y_{t+1} \mid y_1, \ldots, y_t) = P(Y_{t+1} \mid y_{t-k+1}, \ldots, y_t)$$

This results in a $k$-th order Markov model. How can we control its complexity?

**Key idea:** don‘t allow arbitrary dependence on previous $k$ values.
Gaussian linear time series

- For example, assume

\[ P(Y_{t+1} = y \mid y_{t-k+1}, \ldots, y_t) = \mathcal{N}(y; w_0 + \sum_{i=1}^{k} w_i y_{t-k+i}, \sigma^2) \]

- This is called a (Gaussian) autoregressive model of order \( k \)
- It is a special case of a Gaussian process (not discussed here)
- How can we fit the parameters?

- This is simply linear (least-squares) regression!
Reduction to linear regression

\[ P(Y_{t+1} = y \mid y_{t-k+1}, \ldots, y_t) = \mathcal{N}\left(y; w_0 + \sum_{i=1}^{k} w_i y_{t-k+i}, \sigma^2\right) \]

For \( k = 2 \)

\[ D' = \{(x_1', y_1'), \ldots, (x_n', y_n')\} \quad n = t-2 \]

\[ P(y_{3:t} \mid y_{1:2}, \theta) = \prod_{i=3}^{t} P(y_i \mid y_{i-1}, y_{i-2}, \theta) \]

\[ = \prod_{i=1}^{n} P(y_i' \mid x_i', \theta) = \prod_{i=1}^{n} \mathcal{N}(y_i' \mid w^T x_i', \sigma^2) \]
Reduction: Order-k time series to iid supervised learning

\[
\begin{array}{cccccc}
    y_1 & y_2 & y_3 & y_4 & \cdots & y_T \\
\end{array}
\]

\[
\begin{array}{cccc}
    y_1 & y_2 & y_k & y_{k+1} \\
    y_2 & y_3 & y_{k+1} & y_{k+2} \\
    y_3 & y_4 & y_{k+2} & y_{k+3} \\
    \vdots & \vdots & \vdots & \vdots \\
    y_{T-k+1} & y_{T-k+2} & y_{T-1} & y_T \\
\end{array}
\]

Rows iid
Input (X)  Output (Y)
Gaussian non-linear time series

We might more generally assume

\[ P(Y_{t+1} = y \mid y_{t-k+1}, \ldots, y_t) = \mathcal{N}(y; f(y_{t-k+1}, \ldots, y_t; \theta), \sigma^2) \]

for some (possibly nonlinear, multivariate) function \( f \)

This is equivalent to assuming

\[ y_{t+1} = f(y_{t-k+1}, \ldots, y_t; \theta) + \epsilon_t \]

where

\[ \epsilon_t \sim \mathcal{N}(0, \sigma^2) \]

For example, might assume \( f(\cdot; \theta) \) is specified as a neural network with weights given by \( \theta \)

Can train via backpropagation / SGD
Non-gaussian timeseries models

- Can replace the Gaussian likelihood by a different one.
- E.g., if all $Y_t$ are binary, might use Bernoulli likelihood

$$P(Y_{t+1} = 1 \mid y_{t-k+1}, \ldots, y_t) = \frac{1}{1 + \exp\left(-f(y_{t-k+1}, \ldots, y_t; \theta)\right)}$$

- Can train using stochastic gradient descent
Predicting multiple time steps

• Suppose we fit a model

\[ P(Y_{t+1} = y \mid y_{t-k+1}, \ldots, y_t) = \mathcal{N}(y; f(y_{t-k+1}, \ldots, y_t; \theta), \sigma^2) \]

• How about predicting multiple time steps ahead?

\[ P(Y_{t+\tau} = y \mid y_{t-k+1}, \ldots, y_t) = \mathcal{N}(y; f(y_{t-k+1}, \ldots, y_t; \theta), \sigma^2) \]

• What if variance is 0?

\[ \implies y_{t+1} = f(y_t) = f(y_t; \theta) \]

\[ y_{t+2} = f(y_{t+1}) = f(f(y_t)) \]
Uncertainty in prediction

\[ P(Y_{t+1} = y \mid y_t) = \mathcal{N}(y; f(y_t), \sigma^2) \]

\[ P(Y_{t+2} \mid y_t) = \int P(Y_{t+2} \mid y_{t+1}) P(Y_{t+1} \mid y_t) \, dy_{t+1} \]

*In general, integral intractable;*

*P(Y_{t+2} \mid y_t) is not Gaussian*
Predicting multiple timesteps

- In general, computing exact predictions (as possible in Markov models) is intractable.
- However, can approximate it, e.g., via sampling from the model.

\[
\text{Draw samples: } y^{(1)}_{t+1:t+2}, \ldots, y^{(N)}_{t+1:t+2} \\
\text{Then compute, e.g., } \mathbb{E}[Y_{t+2}] \approx \frac{1}{N} \sum_{i=1}^{N} y^{(i)}_{t+2}
\]
Sample approximations of expectations

- $x_1, \ldots, x_N, \ldots$ independent samples from $P(X)$

(Strong) Law of large numbers:

Hereby, the convergence is with probability 1 (almost sure convergence)

Suggests approximation using finite samples:

$$
\mathbb{E}_P[f(X)] = \lim_{N \to \infty} \frac{1}{N} \sum_{i=1}^{N} f(x_i)
$$

$$
\mathbb{E}_P[f(X)] \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i)
$$
How many samples do we need?

- Hoeffding’s inequality

Suppose $f$ is bounded in $[0, C]$. Then

$$P \left( \left| \mathbb{E}_P[f(X)] - \frac{1}{N} \sum_{i=1}^{N} f(x_i) \right| > \varepsilon \right) \leq 2 \exp\left(-2N \varepsilon^2 / C^2\right)$$

- Thus, probability of error decreases exponentially in $N!$
How to obtain samples (k=1)?

\[ P(Y_{t+1} = y \mid y_t) = \mathcal{N}(y; f(y_t), \sigma^2) \]
Forward sampling algorithm

Input: model \( P(Y_{t+1} \mid Y_{t-k+1}, \ldots, Y_t) \), data \( y_{1:T} \)
prediction horizon \( \tau \in \mathbb{N} \), #samples \( N \)

For \( i=1:N \)

- Set \( y_{1:T}^{(i)} = y_{1:T} \)
- For \( t = T : T + \tau - 1 \) do

  Sample \( y_{t+1}^{(i)} \sim P(Y_{t+1} \mid y_{t-k+1}^{(i)}, \ldots, y_{t}^{(i)}) \)

Then, e.g., predict mean (for continuous \( Y \)) by

\[
\mathbb{E}[Y_{T+\tau}] \approx \frac{1}{N} \sum_{i=1}^{N} y_{T+\tau}^{(i)}
\]
Model selection

Wait.. If time-series modeling can be reduced to standard (iid) supervised learning, what about model selection?

In particular, how can we implement cross-validation?
Temporal cross-validation

Partition data into $r+1$ fold, respecting time order.
I.e., $D_1$ contains the first $T/(r+1)$ data points, $D_2$ the second etc.

For $i=2:k+1$ do
- Train on folds $D_1,...,D_{i-1}$, and test on fold $D_i$ only
- Report average over the $k$ performance estimates
Summary so far

- Sequence prediction (with bounded memory) can be reduced to supervised learning
  
  ➔ Can use all the standard tools we learned in this class!

- For the special case of Markov models, prediction can be done recursively (via matrix multiplication)

- For the general case, can try to fit a parametric model, and then produce samples from the model

- Can do model selection via temporal cross-validation
Outlook

- Can extend time-series models to allow for hidden variables (very useful to capture memory)

- Examples:
  - Hidden Markov Models (generalizes Gaussian Mixtures)
  - Recurrent Neural Networks (RNNs)

- These models also allow sequence-to-sequence prediction. Many applications:
  - Speech recognition
  - Handwriting recognition
  - Statistical translation
  - ...