AI/ML in Networking

Lecture 4: Time Series & Wi-Fi Traffic Prediction

Francesc Wilhelmi (francisco.wilhelmi@upf.edu)

Università degli Studi di Cagliari

Overview

1. Introduction to time series forecasting

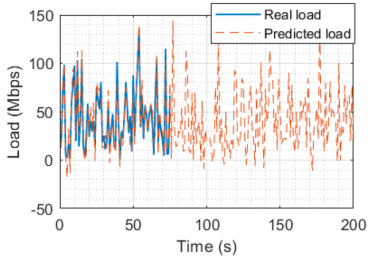
2. ML models for time series forecasting

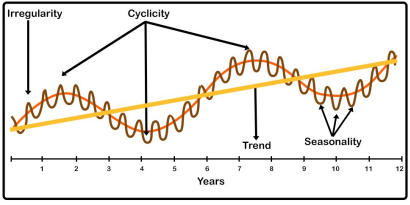
3. Time series forecasting applications

Outline

Time series

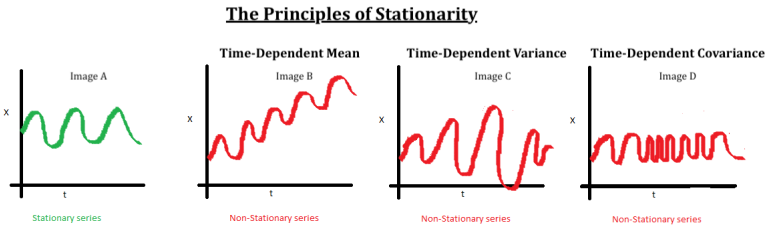
- A **time series** is a sequence of data points indexed in time order.
- **Time series forecasting** is the process of predicting future values based on historical data.
- Characteristics of a time series:
  - **Trend:** Long-term increase/decreases.
  - **Seasonality:** Repeating patterns at fixed intervals (e.g., daily, weekly, monthly).
  - **Cyclicity:** Repeating patterns at irregular intervals.
  - **Irregularity/Noise:** Random fluctuations.





Stationarity in Time Series

- A stationary time series is one whose statistical properties (mean, variance, and auto-correlation) are constant over time.
- Stationary time series are easier to forecast because their behavior is more predictable.
- Stationarity is needed for some time series forecasting models (e.g., ARIMA), as they assume that the underlying process generating the data is stationary.
- Non-stationary series often exhibit trends, seasonality, or changing volatility.



Types of Stationarity

- **Strict Stationarity:** A time series $\{X_t\}$ is strictly stationary if the joint distribution of $(X_{t_1}, \ldots, X_{t_k})$ is the same as $(X_{t_1+h}, \ldots, X_{t_k+h})$ for all integers $k, t_1, \ldots, t_k, h$. This is a very strong condition.
- **Weak (or Covariance) Stationarity:** A time series $\{X_t\}$ is weakly stationary if:
  1. The mean is constant: $\mathbb{E}[X_t] = \mu$ for all t.
  2. The variance is constant: $\text{Var}(X_t) = \sigma^2 < \infty$ for all t.
  3. The autocovariance depends only on the lag: $\text{Cov}(X_t, X_{t-h}) = \gamma(h)$ for all t and h.

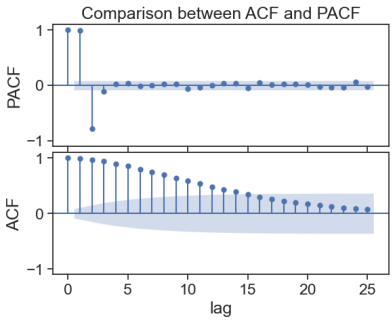
Assessing Stationarity

Visually:

- Temporal plot: Look for trends, seasonality, or changing variance over time.
- Autocorrelation Function (ACF) plot: ACF typically drops off relatively quickly to zero if there's stationarity.
- Partial Autocorrelation Function (PACF) plot

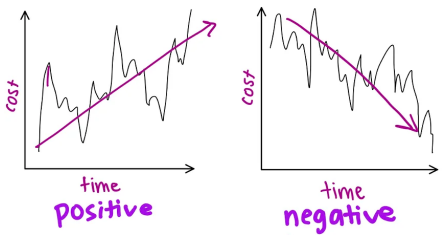
Statistical Tests:

- Augmented Dickey-Fuller (ADF) test
- Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test



Trends in Time Series

- A trend is a long-term increase or decrease in the data over time.
- Regardless of short-term fluctuations or seasonal patterns.
- The presence of trends implies non-stationarity, as the mean of the series is changing over time.
- Identifying trends helps in understanding the behavior of the time series (e.g., economic growth, population change), which is useful to make better predictions.



Types of Trends

- **Linear trend:** A constant rate of increase or decrease over time.

$$Y_t = \beta_0 + \beta_1 t + \epsilon_t$$

  where Y_t is the series at time t, β_0 is the intercept, β_1 is the slope, and ϵ_t is the error term.

- **Quadratic trend:** A non-linear trend where the rate of change is not constant.

$$Y_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \epsilon_t$$

- **Exponential trend:** The series increases or decreases at a constant percentage rate. Often linearized by taking logarithms.

$$Y_t = \beta_0 e^{\beta_1 t} \cdot \epsilon_t \quad \Rightarrow \quad \ln(Y_t) = \ln(\beta_0) + \beta_1 t + \ln(\epsilon_t)$$

Trend Identification and Removal

- **Moving average analysis:** Reduce short-term fluctuations and to highlight the trend.

$$\bar{Y}_t = \frac{1}{k} \sum_{i=0}^{k-1} Y_{t-i}$$

- **Regression analysis:** Fitting a regression model ($Y_t = \beta_0 + \beta_1 t + \epsilon_t$) to the time series data to find the trend.
- **Differencing:** Taking the difference between consecutive observations to remove the trend.

$$\Delta Y_t = Y_t - Y_{t-1}$$

If the first difference still shows a trend, a second difference can be taken:

$$\Delta^2 Y_t = \Delta Y_t - \Delta Y_{t-1} = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2})$$

Outline

1. Introduction to time series forecasting

2. ML models for time series forecasting

3. Time series forecasting applications

Summary of Approaches

- **ARIMA:** Statistical model for stationary time series.
- **RNNs (GRU & LSTM):** Neural networks with explicit memory tracking, which allows capturing sequential dependencies of data.
- **CNNs:** Neural networks based on ‘convolutions’. Effective for local feature extraction and parallelization.
- **Transformers:** Attention-based type of neural networks, powerful for long-range dependencies.

ARIMA (AutoRegressive Integrated Moving Average)

- A statistical model for time series forecasting.
- Consists of three components:
  - **AR (AutoRegressive):** Combine past observations linearly.
  - **I (Integrated):** Apply differencing to make the time series stationary (remove trend and seasonality).
  - **MA (Moving Average):** Combine the previous forecast errors.
- Denoted as ARIMA(p, d, q), where p, d, q are the orders of AR, I, and MA components.
- It makes sense to use ARIMA when:
  - The time series data is stationary or can be made stationary through differencing.
  - For relatively short-term forecasting.
  - When the interpretability of the model is important.

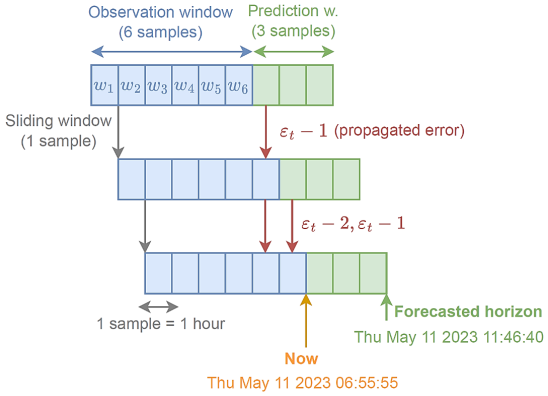
ARIMA

- Decompose the stream of data into time series

$$\hat{y}_t = \sum_{p=1}^{P} \omega_p y_{t-p} + \sum_{q=1}^{Q} \omega_q \varepsilon_{t-q}$$

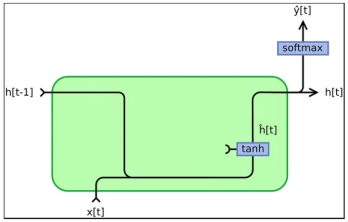
Predicted sample — Weighted past values — Importance of past errors

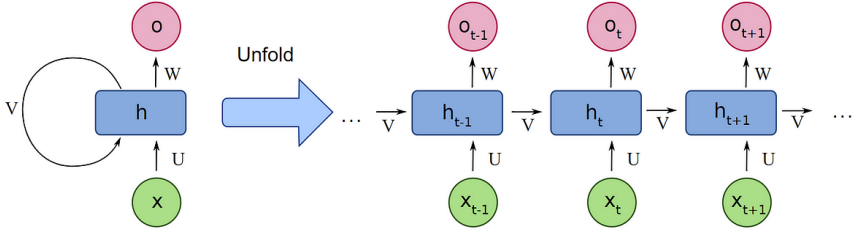
- "Rolling forecast origin": Use a sliding window where N slots (e.g., last 24h) are used to predict the next M slots (e.g., next hour)
  - One-step forecast: $M = 1$
  - Multi-step forecast: $M > 1$
  - Multi-step forecast with estimations: $M >> 1$ (predictions are used to make further predictions in the future)



Recurrent Neural Networks (RNNs)

- A type of neural network designed to process sequential data (e.g., text, speech, video, traffic).
- RNNs use information from previous steps in a time series.
- RNN blocks combine current (t) and past samples (t-1) using a given function (e.g., hyperbolic tangent).





Gated Recurrent Unit (GRU)

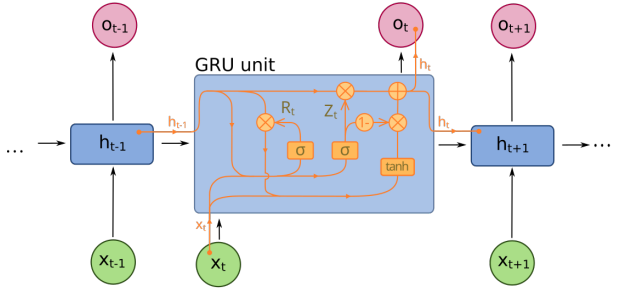
- A simple type of RNN based on *update* and *reset* gates.
  1. **Update Gate (z_t):** Controls how much new information should be incorporated.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{1}$$

  2. **Reset Gate (r_t):** Controls the importance of the past information.

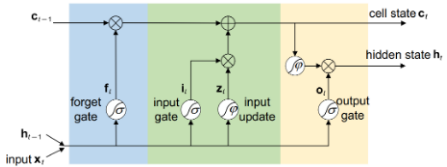
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{2}$$

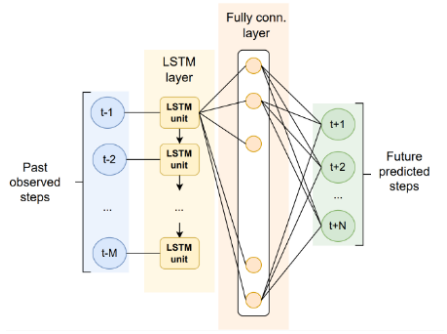
- Fast training and low complexity.



Long Short-Term Memory (LSTM)

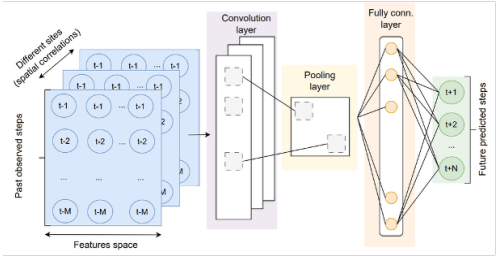
- LSTM is a parametric type of model that extends RNNs
- LSTM Networks are formed by layers of connected LSTM units, with input, forget, and output gates, which control the flow of information.
  - A feed-forward fully connected layer is added to generate the predictions (N future slots)
- Pros and cons:
  - Good to avoid learning long-term dependencies ("vanishing gradient" problem)
  - Scalability issues $\rightarrow$ slow training on large data





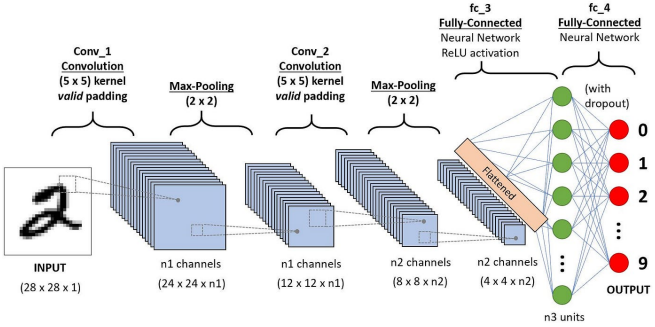
Convolutional Neural Network (CNN)

- A CNN is a parametric model that uses **convolutions** to focus the attention on the most important patterns in data.
- It includes **pooling** layers to reduce the complexity of the introduced features and the output of convolutions.
- CNNs are good for problems with multiple variables and for automatic detection of relevant features through convolutions.
- Can be used as feature extractors for other models (e.g., followed by LSTMs).
- However, they are computationally expensive (may require GPUs) and require big datasets to offer good results.



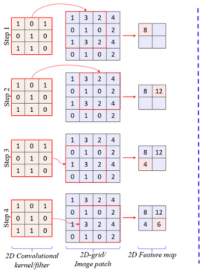
CNN Elements

1. **Convolutional layer:** Applies learnable filters (kernels) to the input. Produce feature maps.
2. **Activations (e.g., ReLU)**: Applies a non-linear activation function element-wise to the feature maps.
3. **Pooling layer:** Down-samples the feature maps, reducing dimensionality.
4. **Fully-connected layer:** Takes the flattened output from preceding layers and performs classification or regression.

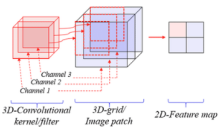


The Convolution Operation

- A filter (kernel) slides over the input (e.g., image) to identify local patterns.
- At each position, it performs element-wise multiplication between the filter and the corresponding input patch, then sums the results.
- Parameters:
  - **Filter Size:** Dimensions of the kernel (e.g., 3×3, 5×5).
  - **Stride:** The number of elements that the filter shifts in each step.
  - **Padding:** Adding zeros around the input border to preserve spatial dimensions or prevent loss of information.



Convolution on 2D-grids



Convolution (Step 1 only) on 3D grids

Altaf, Fouzia, et al. "Going deep in medical image analysis: concepts, methods, challenges, and future directions." IEEE access 7 (2019).

Sample filter for cat detection in CNN

https://sefiks.com/2017/11/03/a-gentle-introduction-to-convolutional-neural-networks/

*A nice video to understand convolutions: https://www.youtube.com/watch?v=KuXjwB4LzSA

Pooling Layers

Max Pooling:

- Selects the maximum value from the window (receptive field).
- Emphasizes most prominent features.
- Example (2×2 maxpool with stride 2):

$$\text{Input: } \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \xrightarrow{\text{Max Pool}} 4$$

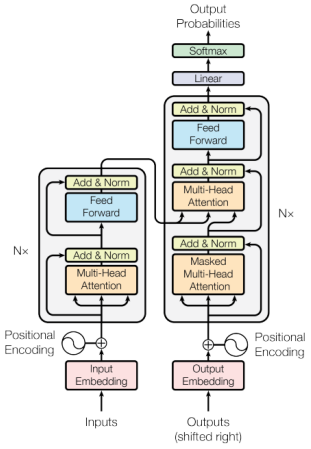
Average Pooling:

- Calculates the average value from the window.
- Smooths out the features.
- Example (2×2 avgpool with stride 2):

$$\text{Input: } \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \xrightarrow{\text{Avg Pool}} 2.5$$

Transformer

- Relies solely on **attention mechanisms**, discarding recurrence and convolutions.
- Excellent at capturing long-range dependencies efficiently.
- Components
  - **Self-Attention:** Allows the model to weigh the importance of different parts of the input sequence when processing each element.
  - **Multi-Head Attention:** Multiple attention mechanisms run in parallel, capturing different aspects of relationships.
  - **Positional Encoding:** Adds information about the position of tokens in the sequence, as attention alone is permutation invariant.
  - **Feed-Forward Networks:** Applied independently to each position.
- Can model complex, non-linear relationships and long-term dependencies, but they are very computationally expensive.



Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Outline

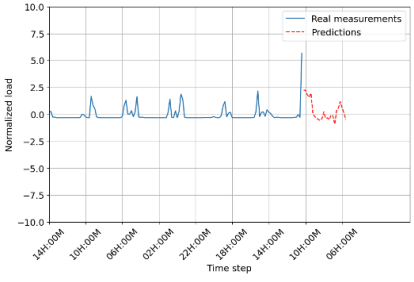
Wi-Fi AP Load Prediction (I)

Goal

- Predict future values of a stream of data (e.g., AP load)
- Different time horizons (e.g., next 1h, next 24h)
- AP load is predicted on a per-AP basis

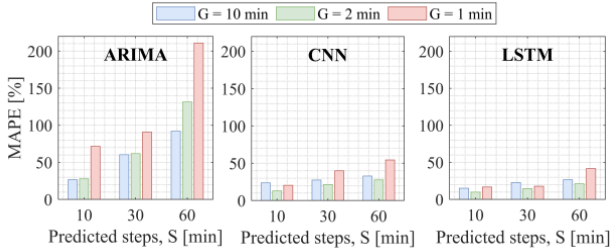
Applications

- Proactively identify hot/cold spots
- Optimize AP configurations
- Anticipate/reduce troubleshooting
- Dynamic power saving
- Smart resource allocation
- Load-aware roaming



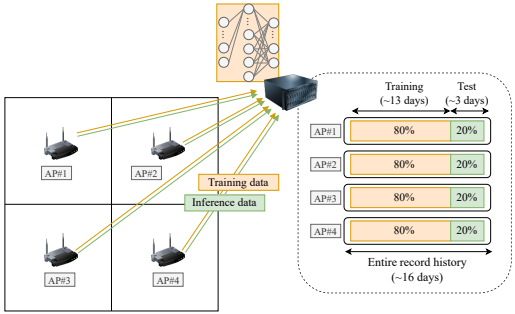
Wi-Fi AP Load Prediction (II)

- Data granularity has an impact on both model's accuracy and cost
- Possible granularities: 1 hour, 10 min, 5 min, 2 min, 1 min
- Training dataset size (data from 16 APs):
  - 1 hour: 6.078 samples
  - 10 min: 36.788 samples
  - 5 min: 66.967 samples
  - 2 min: 167.516 samples
  - 1 min: 368.465 samples



Wilhelmi, Francesc, et al. "AI/ML-based Load Prediction in IEEE 802.11 Enterprise Networks." 2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN). IEEE, 2024.

Wi-Fi AP Load Prediction (III)



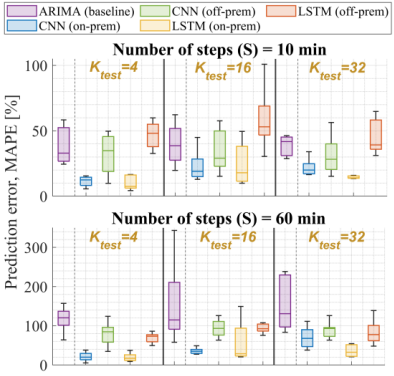


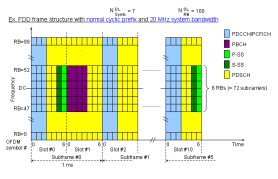
Fig. 2: MAPE achieved by the models on test sets corresponding to $K_{\text{test}} = \{4, 16, 32\}$ APs, for $S \in \{10\,\text{min}, 60\,\text{min}\}$.

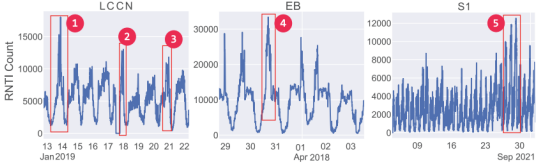
Wilhelmi, Francesc, et al. "AI/ML-based Load Prediction in IEEE 802.11 Enterprise Networks." 2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN). IEEE, 2024.

Cellular Traffic Prediction

- Goal: capture LTE PDCCH (Physical Downlink Control Channel) data every TTI (i.e., 1 ms), which is unencrypted and derive useful information (e.g., load, anomalies)
- Available information:
  - C-RNTI: temporary user identifier
  - MCS: modulation and coding scheme
  - RBs: no. of resource blocks assigned
  - HARQ ID: HARQ process identifier
  - TBS: Transport Block Size (derived)
- Setup:
  - HW: SDR plus mini-PC
  - SW (OWL): open-source software to decode DCI messages

More info at `https://supercom.cttc.es/index.php/supercom-solutions/collect-data/lte-data`





Pavlidis, Nikolaos, Vasileios Perifanis, Selim F. Yilmaz, Francesc Wilhelmi, Marco Miozzo, Pavlos S. Efraimidis, Remous-Aris Koutsiamanis, Pavol Mulinka, and Paolo Dini. "Federated learning in mobile networks: A comprehensive case study on traffic forecasting." IEEE Transactions on Sustainable Computing (2024).