AISE 4010

Deep Learning For Time Series Data

Arnav Goyal

2025-2026

Contents

1 Basic Concepts of Time Series Data

1.1 What is Time Series Data?

Time Series Data is sequential data, essentially data taken over some span of time. They key characteristic is that the data-points are *depdendent* on time, which makes it unique from other types of data. This time-dependency often means the past data influences the present, so with DL models we are basically trying to predict the future.

Some examples of time series data:

- Stock Prices
- Weather History
- Sensor Readings
- Videos (sequence of frames over time)

1.2 Time-Series Analysis

Time-Series Analysis deals wiht what type of information we are trying to extract from a big time-series dataset. There are a few types of analysis.

- Forecasting: aims to predict future values based on past data
  - ex: Getting rich with stock predictions
- Classification & Clustering: involves categorizing different time-series data based on their characteristic
  - ex: grouping together ECG patterns of patients with heart-conditions
- Anomaly Detection: aims to indentify unusual or unexpected patterns in the data
  - ex: Threat detection on IoT network

1.3 Time-Series Modelling

In order to perform the analyses above, we have a few modelling techniques available for modelling time-series-data:

- Classical Statistical Modelling
  - Parametric ARIMA/ARMA/SARIMA/etc models with interpretable coeffs.
- State-Space & Bayesian

  - Kalman filtering/smoothing, missing data handling
- Feature Based ML
  - Turn a series intoo a supervised table and fit a regressor/classifier (Random Forest, XGBoost, etc.)
- **Deep Learning**
  - Learn representations directly from sequences, capturing nonlinear/long-range structures. Best to use DL with Large data , multivariate sensors and complex patterns
- Hybrid & Ensembles
  - Combining different models from the above to use all of their strengths, hybrid models are usually more robust

Using these techniques we can basically perform Forecasting, Classification, and Anomaly Detection

1.4 Why Deep Learning for Time-Series?

Time-Series-Data is usually very complex, large, high-dimensional, and has intricate time dependencies that traditional methods struggle with. It has been proven to be effective in capturing complex patterns in data, in fields such as image recognition, OCR, autonomous driving, TTS, TTImg, etc.

DL Also has the benefit of removing the most painful part of using traditional ML techniques, it doesnt need any human intervention It learns from raw-data alone, which means feature engineering is not required, only dataset cleaning is required. Furthermore DL can also handle non-linearity and can capture super long-range dependencies

1.5 Types of Time-Series-Data

Time-Series data can be categorized according to its characteristics, we can group the data by

- Number of variables: Univariate or Multivariate?
  - Univariate means one variable changes over time, Multivariate means several variables are changing w.r.t time
- Nature of the Data: Continuous or Discrete?
  - This ones pretty self-explanatory
- Statistical Properties: Stationary or Non-Stationary?
  - Stationary means that the data's mean variance and autocovariance remain relatively constant over time, most real-world datasets are Non-Stationary, and must be transformed into Stationary sets for accurate modelling

1.6 Components of Time-Series-Data

Time series data might exhibit some of the following components:

- Trends
  - a long term movement upwards or downwards
- Seasonality
  - a repeating pattern at *regular* intervals (months, years, etc.)
- Cyclic Patterns
  - patterns that occur over *irregular* intervals (over long periods, beyond seasonality)
- Irregularity (Noise)
  - short-term unpredictable variations or random fluctuations in the data that cannot be easily explained

Each type of pattern requires specific analytical approaches, and finding the patterns in your data is the key to a good analysis.

1.7 Estimating Trends

There are a few ways to estimate trends:

- Moving Averages are robust & simple
- Exponential Moving Average: reacts faster than normal Moving Average
- LOESS (Locally Weighted Smoothing): flexible and robust
- Regression (Linear, Poly, Spline): interpretable
- Decomposition (classical or STL) can isolate a smooth "trend" component

2 Time Series Forecasting (Statistical Models)

2.1 Forecasting,

In practice most of the time series represent **stochastic** characteristics, and clearly the future values of the time-series cannot be precisely predicted, In other words there will always be some error with forecasting.

In order to forecast there are two fundamentally different strategies

- ML based (learn variations)
- Statistical based (system identification using a linear/nonlinear model such as ARIMA and SARIMA)

2.2 Cleaning the Data

For stats based forecasting, having clean data is *very* important. Cleaning data involves:

- Handling missing data and outliers
- Changing frequency
- Removing non-stationary effects, as some stats based models cannot handle non-stationary data

2.3 Handling Missing Time-Series Data

this is *very* common, and it is necessary to find the time values that are missing from the time-series, it is very frequent in time-seres due to the burden of long-term sampling. Some methods allow for missing values, but others do not.

The challenging part is that we cannot just put in some random data, as we want to ensure that all repeating patterns are still present, breaking this rule may effect the quality of the forecasting from the model.

There are two types of missing values:

- **MAR (Missing at Random)**
  - Variable has missing values that are randomly distributed, however they are related to some other variables values
- **MNAR (Missing not at Random/Systematic Missing Data)**
  - TODO:

The most common methods to address missing data in time-series are:

- **Imputation**: fill in missing data based on observations on the entire dataset
- **Interpolation**: when we use neighbouring data-points to estimate the missing value. Interpolation can also be a form of Imputation
- **Deletion** of affected time periods: when we choose not to use time-periods that have missing data at all. This risks losing information, and it will result in less data/patterns for your model to learn!

2.3.1 Deletion

This is the easiest way of handling missing data, but might result in some information loss, this is *not* suitable for large amounts of missing data

2.3.2 Statistical Imputation

Here we replace the missing data with the mean, median or mode of the variable, it is one of the most common methods. Its best to use this when the data is missing at random (MAR) and when the % missing values is very low.

2.3.3 Forward & Backward Fill

This is another type of way to handle missing time-series data, and its also very simple. Essentially when there is a trend/pattern that continues from one point to the next we will fill missing values using the most recent (succeeding, or preceeding) value.

In this method we dont consider any relationships between data values, and its bst when dealing with time-series that show a relatively stable trend, or when missing values occur in consecutive intervals Note that backwards filling should only be done when not trying to forecast.

2.3.4 Linear Interpolation

Filling a missing data point by assuming a line between the nearest neighbours, this is a simple and fast approach.

We can use this if there is only linear relationships in the dataset, if there are non-linearities (seasonality, etc) this is not the best type of method as it will disrupt the seasonal pattern

2.3.5 Regression Imputation

A method of imputing missing values by using the relationships between the data. In this method a regression model is used to predict the missing values based on the dataset. This can be used when the data is numeric, and there is a strong correlation between the variable with missing values and other variables.

2.3.6 KNN Imputation

K-Nearest-Neighbors is finding the K-nearest neighbors to the observation with missing data, and imputing them based on the non-missing-values in the neighbors Its best to use this when the data are MAR, its easy to implement, but it should not be used for high-dimensional data due to computational overhead.

2.3.7 Seasonal-Trend Decomposition

This method can capture complex-seasonal patterns that other imputation methods might miss. It can handle any type of seasonality, but its not appropriate to use if the dataset has a high level of noise due to the magnitude of the residual component compared to the trend and seasonality. If there is a high level of noise its best to use robust STL.

2.4 Upsampling and Downsampling

Resampling refers to the change of the freuqency in time-series observations. But why would we resample?

- Matching frequencies: Aligning time-series data from different sources or for different analyses that require a consistent frequency.
- Data Exploration: examining data behaviour at different resolutions to gain insights
- Computational Efficiency: Downsampling can reduce dataset size and the computational load for analysis or model training.
- Modelling Requirements: Some models may require data at a specific frequency

Upsampling means increasing the timestamp frequency, Downsampling is the opposite. Often with Upsampling we use Interpolation techniques to produce new data Commonly we use polynomial interpolation to carry this out.

Downsampling would be carried out by aggregating multiple observations. You can take the mean of these points, etc.

2.5 Removing Non-Stationary Effect

When a time series is **stationary** is can be easier to model. Many stats based modelling techniques assume the data is stationary to be effective. This means that we will have to make non-stationary data into stationary data.

Time series data are stationary if they do not have a trend or seasonal effects. Summary statisitcs calculation on the time series are consistent over time, like the mean the variance or the autocorrelation of the observations.

So how can we tell if our data is stationary?

- Visual inspection
  - can be unreliable

- Summary statistics
- Statistical tests (KPSS, ADF)
  - this is the most accurate way
- Autocorrelation Function (ACF)

2.5.1 Summary Statistics

This is a "quick and dirty" check, but it can be wrong. Essentially this works by splitting the data into two/more partitions and comparing the mean/variance of each group If they differ significantly, the data is likely to be non-stationary

2.5.2 Statistical Tests

Statistical tests (unit root tests) always gives you the right answer. it makes strong assumptions about your data. They can only be used to inform the degree to which a null hypothesis can be rejected, or not be rejected.

- Augmented Dickey-Fuller (ADF)
- Kwaitkowski-Philips-Schmidt-Shin (KPSS)

In ADF:

- Null hypothesis H_0 if not rejected, the suggests the time series have a unit-root, meaning it is non-stationary, so it has some time-dependent structure
- Altnerate hypothesis H_1, is H_0 is rejected, suggests the opposite of H_0
- Interpreting the p-value
  - if $p \leq 0.05$ reject H_0(stationary)
  - if $p > 0.05$ accept H_0 (non-stationary)

Note that KPSS is the inverse of ADF, so H_0 and H_1 are swapped.

2.5.3 Autocorrelation & Partial Autocorrelation Function

Autocorrelation analysis is an important step in the Exploratory Data Analaysis (EDA) of Time-series. It helps to detect hidden patterns and seasonality, it also helps in checking for randomness.

It is very important when using an ARIMA model for forecasting because the autocorrelation helps to indentify the AR and MA parameters for the model.

The autocorrelation function (ACF):: is a correlation between a time-series and its own past values at different time lags (ACF at lag k is the correlation betwen time series values at time t and time $t-k$)

the partial ACF (PACF): TODO

We can use ACF/PACF to determine if data is stationary or non-stationary. In stationary data the ACF/PACF might have a few spikes but WILL quickly decay to 0 Non-stationary data has the ACF/PACF shows no gradual decay, or very slow decay in addtion to spikes at the start. However this is not always the best approach.

2.5.4 Making Time-Series Stationary

We can do this by eliminating the trend, and reducing the seasonality.

- Transforms such as log can help stabilize the variance of a time-series
- Differencing cann help stabilize the mean of time-series by removing changes in the level of a time-series $y_t = y_t - y_{t-1}$

We can apply a lag-2 or second order differenced, occasionally the differenced data will not appear stationary, and may be necessary to difference it again. Going beyond second orders are typically not needed

$$y_t - y_{t-1} - (y_{t-1} - y_{t-2}) = y_t - 2y_{t-1} + y_{t-2}$$

Lag-m or seasonal differincing, it removes the seasonal effect between an observation and a previous observation during the same season. Here m is the number of seasons

$$y_t - y_{t-m}$$

2.6 ARIMA Model Based Prediction

ARIMA stands for "auto-regressive integrated moving average", it features three parts

- The Autoregressor
- The Integrator
- The Moving Average

This model is a class of linear model, that uses past data-values to forecast future values. It can handle non-stationary data.

2.6.1 The Autoregressor

This is essentially predicting a variable with a linear combination of predictors. The AR model assumes that the current point y_t is dependent on previous values due to this assumption we can build a linear-regression model.

The term autoregression indicates that it is a regression of the variable against itself. These models are quite flexibile at handling a wide range of different time-series patterns.

We define an auto-regressive model of order p as AR (p) as below

$$\text{AR}\left(p\right) = c + a_1 y_{t-1} + a_2 y_{t-2} + \dots + a_p y_{t-p} + \epsilon_{\text{whitenoise}}$$

The ACF/PACF can be used to estimate the parameters for our AR. TODO

2.6.2 The Moving Average

Rather than using past-values of the forecasr variable in a regression, the moving average model uses past forecast errors in a regression-like model The MA model assumes that the current value y_t is dependent on the error terms, including the current error

We define a moving-avergae model of order q as MA (q) as below

$$\text{MA}\left(q\right) = c + \epsilon_{\text{whitenoise}} + a_1 \epsilon_1 + a_2 \epsilon_2 + \dots + a_q \epsilon_q$$

2.6.3 The ARIMA

If we combine differencing with the ARMA model, we can the ARIMA model. We define an ARIMA model with order p d and q as below

$$\text{ARIMA}\left(p, d, q\right) :$$

$$y'_t = c + \Sigma_{n=1}^{p} a_n y'_{t-n} + \Sigma_{n=1}^{q} b_n \epsilon_{t-n}$$