1 Time Series Basics

Stochastic process (SP): sequence of random variables indexed in time order: $Y_1, Y_2, \ldots, Y_t, \ldots$

Realization of an SP yields a time series: $\hat{Y}_1 = y_1, \hat{Y}_2 = y_2, \ldots, \hat{Y}_t = y_t, \ldots$

Statistical forecasting notation:

Past observations	$\mathcal{I} = \{\hat{Y}_1 = y_1, \hat{Y}_2 = y_2, \ldots, \hat{Y}_{t-1} = y_{t-1}\}$
Conditioning	$Y_t \mid \mathcal{I}$
Point forecast	$\text{mean}(Y_t \mid \mathcal{I})$
Forecast variance	$\text{var}(Y_t \mid \mathcal{I})$
Conditioning on time series	$Y_{t\mid t-1} = Y_t \mid \{y_1, y_2, \ldots, y_{t-1}\}$
h-step forecast	$Y_{T+h\mid t-1} = \text{E}[Y_{T+h} \mid \{y_1, y_2, \ldots, y_{t-1}\}]$

Time series expressions:

Trend	long-term increase/decrease
Seasonal	predictable, seasonal changes
Cyclic	changes that are not of a fixed period

Autocorrelation function (aka Autocorrelogram):

Seasonality	High positive values at multiples of seasonality
Trend	Higher absolute values for small lags
White Noise	All values below critical threshold of $\pm 1.96/\sqrt{T}$

Statistical Hypothesis Testing (SHT):
- Test time series against null hypothesis (e.g. white noise)
- Statistical test (e.g. Ljung-Box) yields p-Value
- Very small p-Value ⇔ reject null hypothesis

Residuals: difference between observed value and its fitted value: $e_t = y_t - \hat{y}_{t\mid t-1}$
- Residuals have non-zero mean ⇔ model is biased.
- Residuals are correlated ⇔ they contain information not captured by the model.
- Assumption: Residuals have constant variance and are normally distributed.

⇒Residuals should be indistinguishable from white noise → test using SHT (small p-value ⇔ residuals are correlated).

Cross-validation for time series: test set comes after its training set (in time)

Prediction intervals: Confidence region the forecast value is expected to be in. The 95% prediction interval for an h-step forecast can be computed as $\hat{y}_{T+h\mid T} \pm 1.96\hat{\sigma}_h$, with $\hat{\sigma}_h$ as the stddev of the h-step distribution. Assuming the *residuals are normally distributed*, $\hat{\sigma}_h$ can be estimated for different forecast methods by the formulas below.

The *error metrics* MAE, MSE, RMSE and SSE are all scale dependent. MAPE is scale independent but is only sensible if $y_t \gg 0$ for all t, and y has a natural zero.

Error metrics (for n values):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \cdot 100$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

$$\text{MASE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{\frac{1}{n-1}\sum_{i=2}^{n}|y_i - y_{i-1}|}$$

$$\text{SSE} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

2 Forecasting

- forecast $\hat{y}_{t\mid t-1}$ ⇔ fitted value of y at time t given past values up to time $t-1$
- We can perform h step forecasts by iteratively applying one step forecasts.
- There are different models to perform forecasts.

2.1 Simple Forecast Methods

Method	Description	Formula for h step forecast	Confidence $\hat{\sigma}_h$
Average	Mean of all past observations	$\hat{y}_{T+h\mid T} = \bar{y} = (y_1 + \cdots + y_T)/T$	$\hat{\sigma} \cdot \sqrt{1 + 1/T}$
Naïve	Value of last observation	$\hat{y}_{T+h\mid T} = y_T$	$\hat{\sigma} \cdot \sqrt{h}$
Seasonal Naïve	Last observed value from the same season	$\hat{y}_{T+h\mid T} = y_{T+h-m(k+1)}$	$\hat{\sigma} \cdot \sqrt{k+1}$
Drift	Last value plus average historical change	$\hat{y}_{T+h\mid T} = y_T + \frac{h}{T-1}(y_T - y_1)$	$\hat{\sigma} \cdot \sqrt{h \cdot (1 + h/T)}$

In the table above, T is the number of historical values, $\hat{\sigma}$ is the standard deviation of the residuals, m is the seasonal period (e.g. 12 for monthly data), and $k = \text{floor}[(h-1)/m]$ is the number of whole seasons.

2.2 Exponential Smoothing

→ Weighted average of historical data with exponentially decreasing weights. Parameters can be found by minimizing the SSE.

Simple Exponential Smoothing (SES) (N, N): no trend or cyclic components → flat forecasts for all time horizons
$\hat{y}_{T+1\mid T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \ldots$, where $0 \le \alpha \le 1$. The weights sum up to one as a geometric series.
Forecast Equation: $\hat{y}_{t+h\mid t} = \ell_t$
Smoothing Equation: $\ell_t = \alpha y_t + (1-\alpha)\ell_{t-1}$

Holt's linear trend (A, N)
Forecast Equation: $\hat{y}_{t+h\mid t} = \ell_t + hb_t$
Smoothing Equation: $\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + b_{t-1})$
Trend Equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$

Holt's damped trend A_d, N
Forecast Equation: $\hat{y}_{t+h\mid t} = \ell_t + (\phi + \phi^2 + \cdots + \phi^h)b_t$
Smoothing Equation: $\ell_t = \alpha y_t + (1-\alpha)(\ell_{t-1} + \phi b_{t-1})$
Trend Equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)\phi b_{t-1}$

Holt-Winters additive seasonality (A,A)
Forecast Equation: $\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)}$
Smoothing Equation: $\ell_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})$
Trend Equation: $b_t = \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}$
Seasonality Equation: $s_t = \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}$

Holt-Winters multiplicative seasonality (A,M)
Forecast Equation: $\hat{y}_{t+h|t} = (\ell_t + hb_{t-1}) \cdot s_{t-m+h-m(q+1)}$
Smoothing Equation: $\ell_t = \alpha\frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1})$
Trend Equation: $b_t = \beta(\ell_t - \ell_{t-1}) + (1-\beta)b_{t-1}$
Seasonality Equation: $s_t = \gamma\frac{y_t}{\ell_{t-1}+b_{t-1}} + (1-\gamma)s_{t-m}$

Exponential Smoothing State Space Models (ETS)
Contrary to pure forecasting methods, State Space Models can provide point forecasts as well as prediction intervals.
ETS models have three parameters:
Error $\in \{A, M\}$ for {additive, multiplicative}
Trend $\in \{N, A, A_d\}$ for {none, additive, additive damped}
Seasonal $\in \{N, A, M\}$ for {none, additive, multiplicative}

Parameter ranges: $0 \leq \alpha, \beta^*, \phi \leq 1, \quad 0 \leq \gamma \leq 1-\alpha$

Automatic Forecasting aims to find the best performing model out of all possible permutations above by choosing the model with the lowest $AIC = -2\ln(\mathcal{L}) + 2k$ or $AICc = AIC + \frac{2k(k+1)}{n-k-1}$, where $\mathcal{L}$ is the likelihood of the model, k is the number of estimated parameters in the model and n is the number of observations.

2.3 Autoregressive Integrated Moving Average (ARIMA)

Combines autoregressive (AR) and moving average (MA) components along with differencing (I).

Random Walk (with drift): $y_t = c + y_{t-1} + \epsilon_t$
Stationarity: Statistical properties remain constant over time. The ACF of stationary processes quickly drops to zero and there is no seasonality visible.

Differencing can be used (repeatedly) to obtain a stationary time series.
Backshift notation: $By_t = y_{t-1}$; $y'_t = y_t - y_{t-1} = (1-B)\cdot y_t$; $y''_t = (1-B)^2 \cdot y_t$; Seasonal: $y'_t = y_t - y_{t-m} = (1-B^m)y_t$

Autoregressive (AR) Models use multiple regression with lagged values of y_t as predictors: $y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$, denoted as AR($p$).
They work only on stationary data. General condition for stationarity: Complex roots of $1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ lie outside the unit circle. For $q = 1$: $|\phi_1| < 1$.

ARMA: $y_t = c + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q}$
ARIMA(p, d, q): Combine ARMA with differencing
p = order of the AR part
d = degree of first differencing involved
q = order of the MA part
ARIMA = ARMA followed by $(1-B)^d y_t$
ARIMA(1,1,1) model: $\underbrace{(1-\phi_1 B)}_{\text{AR(1)}}\underbrace{(1-B)}_{\text{1st diff}} y_t = c + \underbrace{(1+\theta_1 B)\varepsilon_t}_{\text{MA(1)}}$
ARIMA(1,1,1): $y_t = c + y_{t-1} + \phi_1(y_{t-1} - y_{t-2}) + \theta_1 \varepsilon_{t-1} + \varepsilon_t$

Moving Average (MA) Models use multiple regression with past noise as predictors: $y_t = c + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q}$, denoted as MA($q$).
Any MA(q) process can be written as an AR(∞) when the roots of the characteristic polynomial lie outside the unit circle. General condition for invertibility: Complex roots of $1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q$ lie outside the unit circle. For $q = 1$: $|\theta_1| < 1$. For $q = 2$: $|\theta_2| < 1, \quad \theta_1 + \theta_2 > -1, \quad \theta_1 - \theta_2 < 1$

ARIMA model examples:

ARIMA$(0,0,0)$	white noise model
ARIMA$(0,1,0), c = 0$	random walk
ARIMA$(0,1,0), c \neq 0$	random walk with drift
ARIMA$(p,0,0)$	AR(p)
ARIMA$(0,0,q)$	MA(q)

3 Digital Signals

Complex exponential sequence: $x(n) = Ae^{j(\omega_0 n + \varphi)} = A\cos(\omega_0 n + \varphi) + jA\sin(\omega_0 n + \varphi)$, with $\omega_0 = \frac{2\pi}{T_0} = 2\pi f_0 = 2\pi\frac{f}{f_s}$

Discrete Fourier Series: $x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k) e^{j\overbrace{(2\pi/N)k}^{\omega_0(k)} n} \Leftrightarrow X(k) = \sum_{k=0}^{N-1} x(n) e^{-j(2\pi/N)kn} = X(f{=}\frac{f_s k}{N}) \quad (0 \leq k, n < N)$

DFT properties

Property	Time Domain	Frequency Domain
Periodicity	$x(n) = x(n+N)$	$X(k) = X((k))_N$
Linearity	$ax_1(n) + bx_2(n)$	$aX_1(k) + bX_2(k)$
Convolution	$x_1(n) * x_2(n)$	$X_1(k)X_2(k)$
Multiplication	$x_1(n)x_2(n)$	$\frac{1}{N}(X_1(k) * X_2(k))$

LTI systems
Output: $y(n) = \sum_{k=-\infty}^{\infty} x(n)h(n-k) = x(n) * h(n)$
Frequency response $H(\omega)$: $Y(\omega) = X(\omega)H(\omega)$
FIR Filter: $y(n) = a_0 x(n) + a_1 x(n-1) + \cdots + a_p x(n-p)$

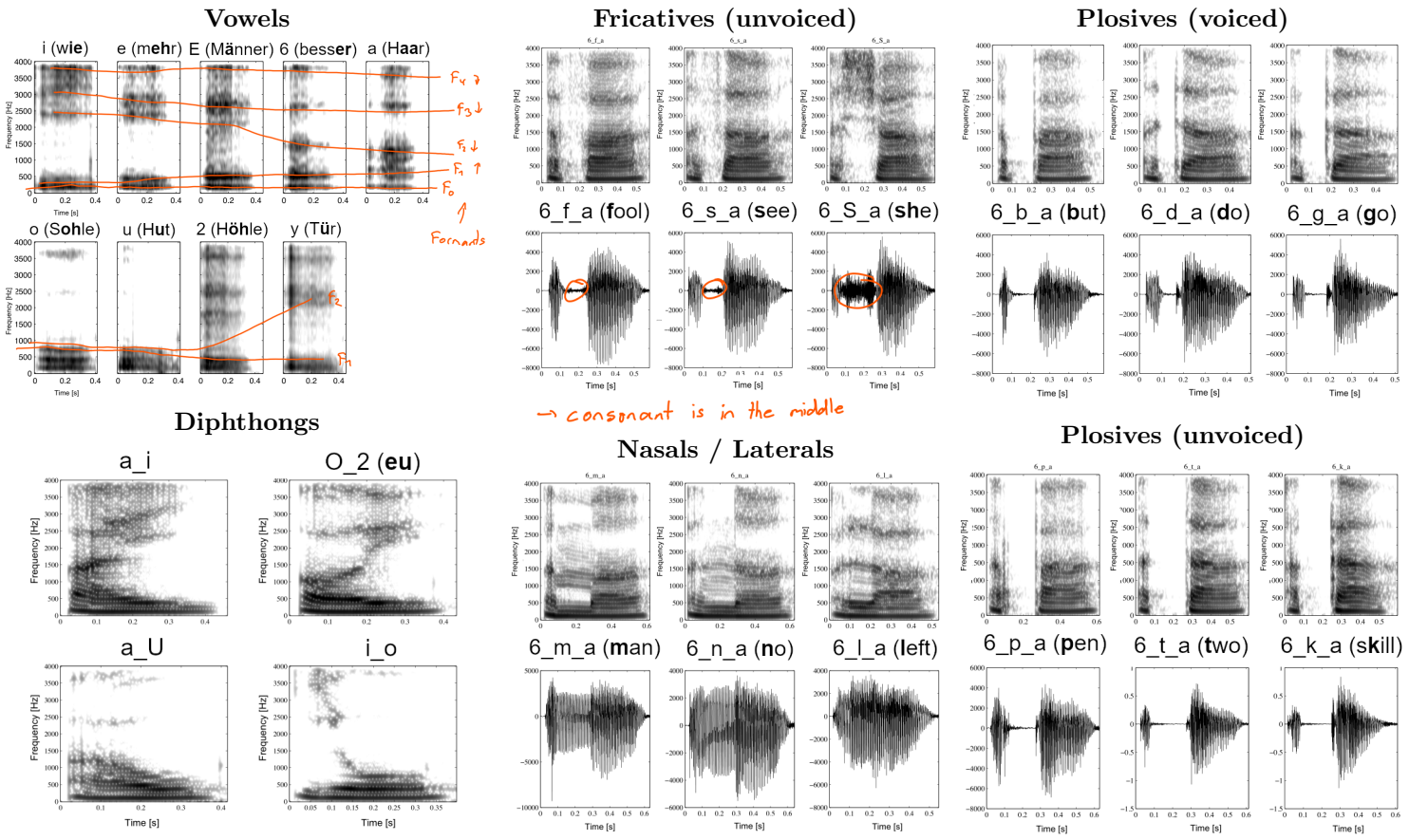
4 Speech Signals

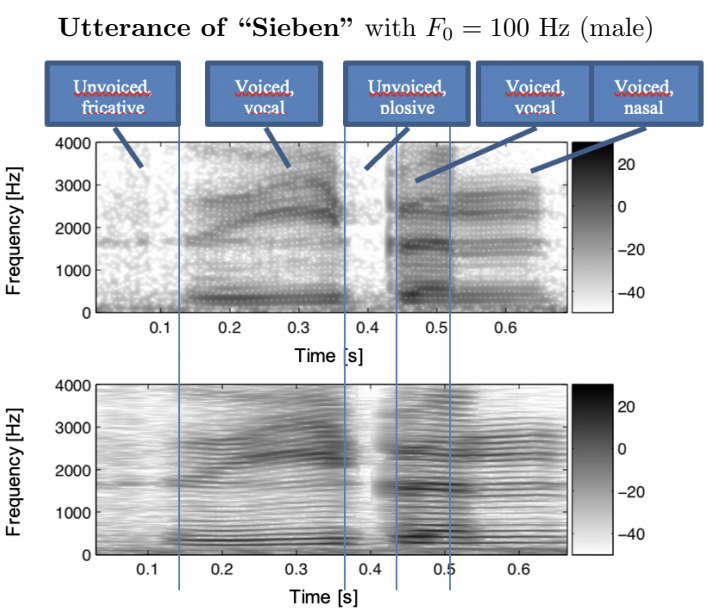
Information content: What?, Who?, How?, speaking environment, transmission channel, background noise

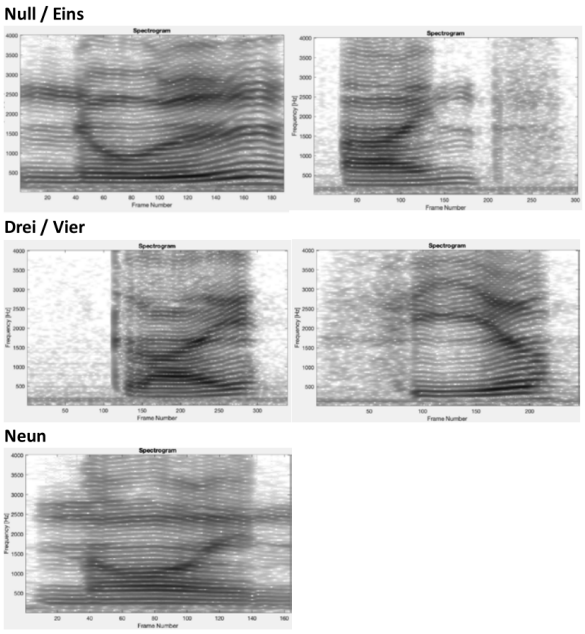
Phonemes: Smallest sound unit, speaker-independent. Western languages have 20 to 60 phonemes (German: 48).
Phones: Acoustic representation of the phoneme, speaker-dependent.

Short-time spectral analysis: Calculate the spectrum of a sliding window. Idea: speech signal is quasi-stationary inside the window. Spectrum of the windowed signal: $\bar{X}(\omega) = X(\omega) * W(\omega)$. Smoother window $\leftrightarrow$ lower side lobes. Longer window $\leftrightarrow$ higher spectral resolution. The **Spectrogram** shows the temporal changes in the signal spectrum.

Formants are high energy areas in the spectrogram (usually dark). The fundamental frequency F_0 ranges from 50 Hz (deep mans voice) to 400 Hz (child), depending on the speaker. The formants F_1 to F_4 convey information about the phone sequence. F_1 and F_2 are speaker dependent.







4.1 Speech Recognition

Speech recognition is the technology that converts spoken language into written text, allowing computers to understand and process verbal commands or transcribe spoken words. Very difficult because human voices can be very different, humans make mistakes, background noises and different acoustics for different environments. Also, there are no word boundaries in human speech.

4.1.1 Feature Extraction

For speech recognition, spectrograms and the formants are important features. Unimportant information can be filtered out by many means, e.g. DFT-Cepstrum (IDFT of the Log of the DFT-Spectrum) to extract the source (vocal excitation) from the filter (vocal tract) components. A Mel-Spectrum is a spectrum that more closely resembles the characteristics of the human ear by having logarithmic sensitivity for frequencies above 1 kHz (lower sensitivity for higher frequencies). Mel-Cepstrum: IDFT of Log(Mel-Spectrum). The Fast Cochlea Transform (FCT) is a signal processing technique inspired by the auditory processing in the human cochlea.

4.1.2 Classical Approaches: Rule- based and Pattern Matching

Both require feature extraction from the speech signal.

Rule- based: Classification is based on rules for each class derived from human knowledge/observation. Very difficult because not generalizable for different speakers.

Pattern Matching: Classification based on a distortion measure between a feature pattern and given reference patterns for each class (kNN, SVMs). Main problem is that the patterns can also vary in the temporal structure. We can use Dynamic Time Warping (DTW) to align the extracted features to the reference features by locally stretching or squeezing pattern by duplicating or dropping feature vectors.

4.1.3 Statistical Classification

To calculate joined probabilities, we can use bayes rule: $P(X|W) = \frac{P(W|X) \cdot P(W)}{P(X)}$

Given extracted feature X from the speech recognizer, the MAP classifier must choose the word sequence W that has the highest a-posteriori probability of all possible word sequences.

4.1.4 Hidden Markov Model (HMM)

Defined by N states, state transition probabilities as well observation probability distribution in each emitting state.

Forward Algorithm: Given a HMM λ, a sequence of emitted observations $\mathbf{X} = x_1, x_2, \ldots, x_T$, we want to efficiently calculate $P(\mathbf{X}|\lambda)$. Known as Evaluation problem. Brute force approach would be to simply try out all state combinations and see which one has the highest likelihood of giving the seen sequence. Forward algorithm takes advantage of independent states and instead keeps the probabilities for emitting the observations for each state. Forward: Addition.

Viterbi Algorithm: Given a HMM λ, a sequence of emitted observations $\mathbf{X} = x_1, x_2, \ldots, x_T$, we want to efficiently calculate the most likely state sequence $Q* = \max_Q P(Q|\mathbf{X}, \lambda)$. Known as Decoding Problem. Same idea as Forward, but instead of adding all possibilities, we always take the max. N-best Viterbi outputs the N-best state sequences instead of only the best.

Forward-Backward algorithm/Baum-Welch algorithm: Given a sequence of emitted observations $\mathbf{X} = x_1, x_2, \ldots, x_T$ and a model structure, find parameters for HMM ω such that $\omega* = \max_\lambda P(\mathbf{X}|\lambda)$. Known as Estimation Problem. How HMMs are trained. Forward-Backward estimates the probabilities of the sequence given the current models while Baum-Welch applies changes to optimize the parameters, kind of like backpropagation.

4.1.5 HMM Speech Recognizer

Speech recognition is done with HMMs of sub-units that are concatenated to word and sentence recognition networks. Viterbi then finds the most likely path through the network. State sequence → subunit sequence → word sequence → sentence

Lexicon: Describes pronounciation of all words allowed; Language model: Describes all utterances allowed and their probabilities. Typically, Mel-Cepstrum, Delta-Cepstrum or other are used as features.

Disadvantages: HMM assumptions never totally met in practice: Conditional independence of states and observations; Training HMMs is not inherently discriminative but rather likelihood maximizing.

4.1.6 Deep Learning Speech Recognizer

Datasets: Split dataset into training and test/evaluation sets. **Word Error Rate**: Count substitutions (S), insertions (I), deletions (D) and divide by number of words in ground truth. **Data Preprocessing** Scale input data to have zero mean and unit variance. **Initialization**: use random gaussian distributed weights. **Overfitting**: Memorizing the training set. Use dropout to mitigate. **Batch Size**: Compromise between faster and more optimal training. **Batch Normalization**: Scale the activations to have zero mean and unit variance (only during training).

Normal Depp Neural Networks (DNN) classify only input patterns of constant size (e.g. image), so other architectures are used.

DNN-HMM: Use DNN as feature extractor for HMM; **CNN**: Use CNN on spectrogram; **LSTM vs. GRU**: both types of recurrent neural network (RNN) architectures designed to address the vanishing gradient problem in traditional RNNs. LSTMs have a more complex memory cell that consists of a cell state and three gates - input gate, forget gate, and output gate. GRUs have a simpler memory cell with two gates - reset gate and update gate.